Policy optimization of language models to align honesty and efficiency of reasoning agents in multi-turn conversations

Anonymous Author(s)
Affiliation
Address
email

Abstract

Reinforcement learning from human preferences can fine tune language models for helpfulness and safety, but does not directly address the fidelity and efficiency of reasoning agents in multi-turn conversations. We propose a method to improve the validity, coherence and efficiency of reasoning agents by defining a reward model as a mapping between predefined queries and tools which can be applied to any custom orchestration environment. The reward model is used for policy optimization to fine tune the clarification fallback behavior and help the agent learn when best to ask for clarifications in multi-turn conversations. A separate, adversarial agent generates prompt variation around each query, which increases robustness of the learned policies and enhances generalization. This is demonstrated in several orchestration environments where after fine tuning with either proximal policy optimization or verbal reinforcement, the new policy systematically identifies the correct intents and tools in < 2 steps in over 99% of all sampled dialogues.

1 Introduction

Combining large languages models (LM) with policy optimization based on human preferences has recently led to new generations of chatbots showing human-level capabilities in helpfulness and safety, with each new release often massively better than previous LM generations. As shown in Bai et al. [2022], Touvron et al. [2023], Lee et al. [2023], by using different reward models for helpfulness and safety, reinforcement learning from human feedback (RLHF) and AI feedback (RLAIF) can be scaled to help find a better solution to the tradeoff between helpfulness and harmlessness. Direct policy optimization (DPO, Rafailov et al. [2024]) leads to similar improvements by tuning the policy directly from a preference dataset instead of encoding the dataset into a reward model for reinforcement learning. In all cases, the improved Pareto frontier between these two capabilities has been shown to result in more honest LM by reducing hallucination (Bai et al. [2022], Touvron et al. [2023]).

RLHF/RLAIF and DPO can fine tune general LM capabilities such as helpfulness and safety, but cannot directly fine tune the fidelity and efficiency of LM-based reasoning agents, which is essential in the context of customer assistants orchestrating predefined sets of intents and possible actions to take. In this context, the agent must often navigate multi-turn conversations that end when the original intent of the user is fulfilled. With new LM generations reaching human-level capabilities, it has become essential for customer assistants (e.g., Alexa, Siri) not just to increase their semantic capabilities, but to also behave logically and efficiently when the original intent of the user is ambiguous, in particular in cases where the intent is ambiguous *at first*, but simple enough that it can be disambiguated and solved with a few simple actions. For example, finding a good doctor in your area may require some clarification at first, but can be done nearly-perfectly after a few clarifications and a few clicks on the web, just as a human can do.

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

In this paper, we propose a framework to directly optimize the validity, coherence and efficiency of a LM policy specifically for a predefined orchestration environment. The proposed method uses a general reinforcement learning formulation for sequential learning to fine tune LMs in multi-turn conversations. This is in contrast to all RLHF, RLAIF, and DPO methods used in seminal papers from Stiennon et al. [2020], Ouyang et al. [2022], Bai et al. [2022], Touvron et al. [2023], Lee et al. [2023], Rafailov et al. [2024], which treat each turn in a conversation as independent of each other and thus do not directly fine tune the strategic behavior of an agent in a sequential multi-turn conversation. To directly fine tune the strategic reasoning capabilities of LMs in a conversation, we define a reward model as a global (i.e., non-token based) mapping between some predefined queries and actions. This reward model is then used for policy optimization to fine tune the clarification fallback behavior in simulated multi-turn conversations so the agent can learn when best to ask for clarifications. A separate LM generates prompt variation around each query, which increases robustness of the learned policies and enhances generalization. Despite the main limitation of this method which is the need to operate in a predefined environment, this method is very flexible as it can be applied to any customer assistant orchestrating a finite set of intents and possible actions i.e., most cases of practical interest.

To demonstrate the efficiency and flexibility of this method, we first apply it on an orchestration environment typical of customer assistants such as Alexa and Siri, where a custom set of 6 user-intents, each associated to 5 different prompts and a variable number of required slots (a.k.a. context data), is used to simulate multi-turn conversations with a user and define rewards for validity (identify the correct intent), coherence (identify the correct slots), and efficiency. In this experiment, the agent is fine-tuned by proximal policy optimization (PPO, Schulman et al. [2017]) and learn selection policies across user intents and the clarification fallback. Some prompts and slots are voluntarily ambiguous and so the agent may learn to fall back on asking for clarification of intent.

In a second experiment, another LM agent interacts with the LM assistant and emulates the user. This second agent is also fine tuned by PPO and learns to select prompts that lead to higher reward, creating a collaborative environment. During PPO, a cooperation between the two agents emerges and ultimately reaches a Nash equilibrium Silver et al. [2018], as shown in the result section.

In a third and final experiment, we apply our method to a reasoning-acting agent (ReAct, Yao et al. [2022]), again using 6 queries (create *vs.* analyze a picture, compute *vs.* execute an expression, retrieve one of two related but different publications), 6 corresponding tools, and a separate LM to generate prompt variation around each query. In this orchestration environment we apply "verbal" reinforcement learning (VRL) where the parameters of the LM are frozen and a simple look-up table of the reward accumulated for each prompt is appended to the agent's system prompt, and refreshed at every step during training. This verbal reinforcement learning was recently proposed in Shinn et al. [2024] and enables an LM to "reflect" on and learn from its past behavior, without ever changing its parameters. VRL was chosen in this third experiment to showcase the flexibility of the proposed method and its ability to fine tune reasoning agents using a wide range of learning algorithms.

In all orchestration environments tested, with and without adversarial prompt variation, and with either PPO or VRL algorithm, after fine tuning the policy systematically identifies the correct intents and tools in < 2 steps in over 99% of sampled dialogues. This implies the new policy has learned to strategically fall back on asking for clarification of intent in ambiguous cases, instead of proceeding with a potentially wrong intent, and successfully completes dialogues after clarifying the user intent.

2 Policy optimization of reasoning agents in multi-turn conversations

The generalized reinforcement learning formulation for sequential learning to fine tune LMs in multi-turn conversations is presented in Fig. 1. An orchestration environment inherently defines a reward model by deterministically mapping a set of queries to a set of tools for a specific use case. The lack of determinism and need for fine tuning come from the nuances and potential ambiguities across the nearly infinite number of possible ways that a user, in a given natural language, may formulate each query. So it is assumed that a general pre-trained LM would not behave perfectly in such orchestration environment (as otherwise there would be no need for fine tuning). The imperfections are a direct consequence of potential ambiguity between prompts related to different intents and imperfect semantic parsing from the LM. In short, an orchestration setup is assumed to have inherent semantic pitfalls, which may lead to responses that are invalid, incoherent, or inefficient.

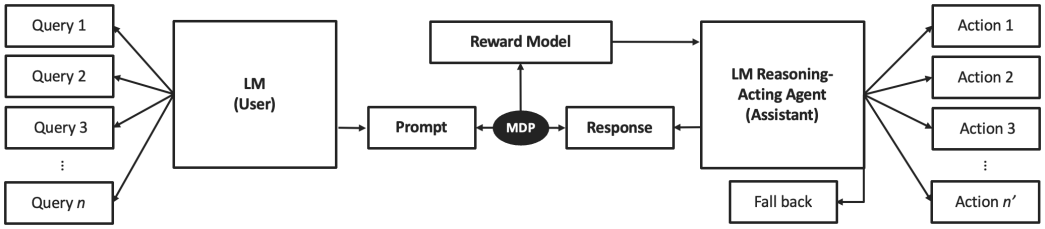


Figure 1: Fallback policy optimization of reasoning-acting orchestration agents. A custom mapping between n queries and n' actions defines the reward model used for fine tuning the agent.

The goal of the proposed method is to optimize the fallback of the agent so it systematically asks for clarifications when it is confused, instead of hallucinating or confidently making a wrong decision (e.g., invoking a wrong tool, proceeding with a wrong intent). Due to the deterministic nature of the query-to-action mapping in a predefined orchestration environment, an optimal fallback behavior can in principle be learned from an offline explore-exploit reinforcement learning algorithm.

The reward model shown in Fig 1 can be used as a human-preference model for policy optimization as in Christiano et al. [2017], Stiennon et al. [2020], Bai et al. [2022], Touvron et al. [2023], Lee et al. [2023], or any other reinforcement learning framework (Sutton and Barto [2018]). Note that in Fig.1, n does not need to be equal to n' as long as each query is unambiguaoulsy mapped to a tool. In this paper we explore a sequential PPO framework learning across sequences of multi-turn conversations (Li et al. [2016]), and verbal reinforcement learning (Shinn et al. [2024]). This method directly fine tunes the performance of a LM in term of validity, coherence, and efficiency of dialogues, by learning where there are semantic pitfalls in this specific orchestration setup i.e., when to ask for clarifications in multi-turn conversations. The details of the reward model, states, actions, and simulation setups, are described in the next section for three different orchestration environments.

3 Methodology

3.1 Proximal policy optimization of the clarification fallback

In a first experiment, a custom orchestration environment (summarized in Appendix A) was defined with a hierarchical modular organization. For example, the intents */pizza* and */dessert* are found within the intent */food*. These intents can be difficult to disambiguate because the prompts used by the user to invoke them may differ by only one or two tokens. Such prompts would benefit from asking for clarification of intent. Each intent is also associated with a variable number of required slots (a.k.a. context data) needed for fulfillment as in a typical Alexa-like customer assistant setup. The slots can be very ambiguous too as some slots are 100% identical between different intents, such as *Are you looking for pick up or delivery* for all intents within */food*. This ambiguity was preserved to evaluate whether the agent could ultimately learn tradeoffs between efficiency and coherence i.e., complex fallback mitigation policies that start fulfilling slot data while asking for clarifications to determine the exact nature of the user's intent. As reported in the result section, the PPO agent did learn such emergent strategies by offline reinforcement learning without the need for human feedback.

An optimal dialogue can be defined as one which is (i) *valid*: the agent identifies the correct user intent, (ii) *coherent*: the agent solicits appropriate context data as required for fulfilling the user intent i.e., it does not solicit data *not* required for fulfillment, and (iii) *efficient*: the agent minimizes the time it takes to fulfill the user intent. In the first experiement, these goals are encoded explicitly into the following reward function:

$$r_t = \lambda_1 \, r_t^1 \, + \, \lambda_2 \, r_t^2 \, + \, \lambda_3 \, r_t^3$$

where λ_1, λ_2, λ_3 are preset weights (hyperparameters) associated to each reward component.

The component r_t^1 rewards the agent depending on whether it identifies the correct intent. The agent can select either of the 6 user intents, or fall back on asking for clarification instead. If the agent guesses the user intent correctly, it receives a positive reward ($+5$). If the agent guesses the intent incorrectly, it receives a negative reward (-5). If the agent falls back on the clarification intent, it is neither rewarded nor penalized by this component i.e., $r_t^1 = 0$.

The component r_t^2 rewards the agent depending on whether it identifies the correct slots. The agent can prompt the user for either of the predefined slots (shown in Appendix A), or not prompt the user for any slot. If the agent solicits a valid slot, it receives a positive reward ($+5$). If the agent solicits slot-data not required for fulfillment, it receives a negative reward (-5). If the agent does not prompt the user for any slot, it is neither rewarded nor penalized by this component i.e., $r_t^2 = 0$.

The component r_t^3 systematically penalizes the agent by a negative value -1 at every step of the dialogue, so the agent is incentivized to close the dialogue quickly. By simultaneously mitigating all three goals of validity, coherence and efficiency, the agent tries to minimize the time it takes to identify the correct intent and solicit valid slots. Given some prompts are ambiguous, the agent may learn to fall back on asking for clarification at any time, for any prompt.

This explicit formulation of the three goals of validity, coherence and efficiency in the reward function is replaced by an implicit formulation in the third experiment (see section 3.3). The advantage of using an explicit formulation is to leverage the full potential of the reinforcement learning formalism for sequential learning with potentially delayed reward. In illustrate this, we decouple the parameters of the actor from the parameters of the LM. The LM is used exclusively for state representation (i.e., natural language embeding vector), and the actor is used exclusively to learn optimal sequential decision-making policies. Any LM encoder can be used to encode user prompts. We experimented with Amazon Titan and also a simpler Word2Vec model applied to the entire corpus of the environment defined by all queries, prompts, and slot requests (Appendix A). Policies learned were similar when using either encoder, probably due to the small size of the corpus. A separate binary vector tracked which of the candidate slots had been filled in the course of each dialogue, and was appended to the LM embedding vector to represent the state of the agent during a conversation.

In this orchestration environment the agent takes two actions simultaneously at every turn of a conversation: it selects an intent or falls back on asking for clarification of intent, and asks the user to fill a slot or does not ask the user to fill a slot. As shown in appendix A, there are 6 possible intents and 6 possible slots across all intents, so the action is defined as a vector of two integers (a, b), where a is the selected intent, b is the selected slot, and a and b can take values between 0 and 6. 0 corresponds to *clarification* and *no slot*, respectively.

To generate dialogues offline for PPO (Schulman et al. [2017]), intents are sampled uniformly at random and five possible prompts for each intent (pre-generated by Claude v3 Sonnet in Amazon Bedrock) are also sampled uniformly at random. Each dialogue (episode) is a multi-turn conversation between the user and the agent, given the user has chosen one intent and the agent needs to identify this intent (or ask for clarification of intent) and fulfil it by soliciting valid slots. During the PPO simulations, the quality of the learned policies is measured by tracking the evolution of accumulated reward and the number of turns in each dialogue, see result section.

3.2 Proximal policy optimization with Nash equilibrium

The above experiment was extended to a multi-agent setup, where two agents interact with each other: one agent emulates the assistant exactly as described in section 3.1, and one agent emulates the user. The key difference is when the assistant asks for clarification, or guesses the intent incorrectly, the user does not select a prompt randomly as in the experiment above, instead it learns to select prompts that lead to higher reward during offline PPO simulations.

Each agent becomes the *environment* from the perspective of the other agent. The behavior of each agent is learned directly by PPO from the sampled dialogues. Both agents aim to maximize the same reward function and thus a cooperation between the two agents is expected (Silver et al. [2018]), which can result in improved speed and coherence as confirmed in the result section. The multi-agent setup was created because in practice, a human user who regularly interacts with a chatbot tends to use words that the chatbot understands better. That is, a human user does not maintain a purely uniform and random choice of words as emulated in the first experiment. Results on the performance of learned policies in this multi-agent setup are reported for an identical orchestration environment as in experiment 1 (appendix A) and an identical number of sampled dialogues (30,000), to easily compare the two experiments, see result section.

3.3 Verbal reinforcement learning of the clarification fallback

Finally, we also applied our method to an LM agent prompted for reasoning-acting (ReAct) as proposed in Yao et al. [2022]). A ReAct agent engages in self-reflection before generating a response for the user: the LM first generates an internal thought, which is a decision to either use a tool amongst a finite set of predefined tools made avaialble to the agent Yao et al. [2022], or respond to the user. If it decides to use a tool, the tool is executed and returns some data which the agent then uses as additional context to generate another thought. The new thought is again a decision to either use a tool or respond to the user. This internal reasoning-acting loop continues until the agent decides it has enough context data to respond to the user. A key requirement in ReAct (Yao et al. [2022]) is to predefine a finite set of tools. This turns out to be a perfect use case for our method. For this experiment, a set of 6 queries and 6 corresponding tools define the reward model (Appendix C). Queries were voluntarily defined to be ambiguous: create *vs.* analyze a picture, compute an expression with a calculator *vs.* execute an expression with a Python interpreter, retrieve Anthropic paper on RLHF *vs.* retrieve Anthropic paper on RLAIF.

A separate LM (adversarial agent) generates prompt variation around each query: 5 prompts were selected per query, again voluntarily selecting some prompts that are very similar between queries. In particular, queries 3-4 and 5-6 share nearly identical prompts, see Appendix D for details.

In this ReAct orchestration environment, we apply "verbal" reinforcement learning where the parameters of the original LM are frozen, and a simple look-up table of the accumulated reward for each prompt variation is appended to the agent's system prompt, and refreshed at every step during training. This type of verbal reinforcement learning was recently proposed in Shinn et al. [2024] to align a LM by memorizing and reflecting on its past behavior instead of fine tuning its model parameters. VRL was chosen here to showcase the flexibility of the method we propose.

4 Results

4.1 Analysis of fallback proximal policy optimization in multi-turn conversations

4.1.1 Detailed PPO simulation setup

The agent emulating the assistant was trained by simulating interactions $(s_t, a_t, r_{t+1}, s_{t+1})$ with users where t is the number of turns in each dialogue. All simulated dialogues were generated using the OpenAI Gym and the PPO algorithm available in Amazon SageMaker from the Inter Coach RL library, for a total number of interactions varying from 130,000 to 150,000 and representing a total of 30,000 dialogues. Each simulation was repeated 3 times and run on 36 CPUs using C5 9XL Amazon EC2 instances, taking approximately 5h each. An efficient exploration of state-action space was ensured by applying an ϵ-greedy exploration schedule: a search over different ϵ-schedules showed that linearly switching ϵ from 10% to 1% over the first 60,000 interactions led to the best results.

4.1.2 Analysis of User-Chatbot Interactions in Dialogues Simulated with PPO

Fig. 2 shows the evolution of accumulated reward in each dialogue, which measures how good the policy followed was in each dialogue. Thus, when compared between dialogues (along the x-axis in Fig. 2) it measures the relative value of the policies learned. Three independent trials of 30,000 dialogues were produced to assess sensibility to the initial random exploration due to the ϵ-greedy schedule. Fig. 2 suggests the PPO agent first explores the orchestration environment, then identifies a policy that mitigates the multiple learning goals of validity (identify the correct intent), coherence (solicit valid slots), and efficiency. In all trials, a clear transition is observed from a phase of random exploration in the first 10,000 episodes, where the accumulated reward ranges from -120 to $+20$, to a phase where the accumulated reward is spread more narrowly and skewed toward higher values. In the final 5,000 dialogues the accumulated reward ranges between -40 and $+20$. This indicates that the PPO policy now systematically avoids certain behaviors when interacting with the user.

Table 1 shows the average proportion of *successful* dialogues (i.e., intent fulfilled within 10 steps) and the average number of steps in successful dialogues, per intent. Standard errors in parentheses were computed over the three independent trials shown in Fig. 2. In the first 5,000 dialogues, 68% are successful and take > 4 interactions to complete on average. In contrast, in the final 5,000 dialogues, 99% are successful and these take at most 2 steps on average to infer the right intent. Fig. 2 confirms

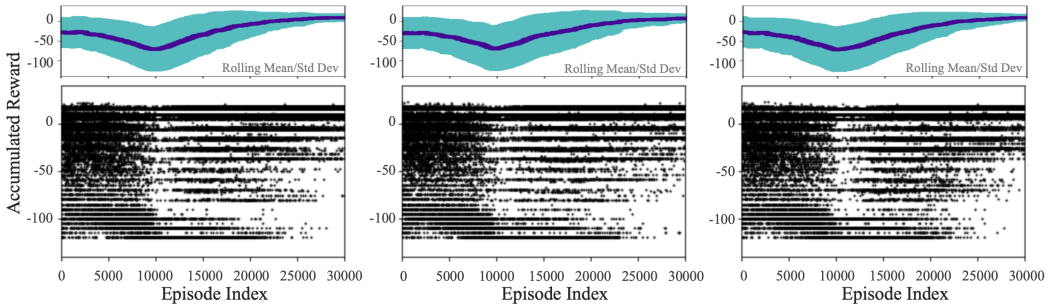


Figure 2: Accumulated rewards across 3×30,000 dialogues sampled with single-agent RL.

Table 1: Validity and efficiency of sampled dialogues.

	Single Agent		Multi Agent	
	0-5K	25-30K	0-5K	25-30K
% success	68 (.8)	99 (.1)	67 (1.4)	100 (0)
Number of steps in successful dialogues:				
Intent 1	4.1 (.2)	1.0 (.0)	4.1 (.1)	1.0 (.0)
Intent 2	4.2 (.1)	1.8 (.2)	4.1 (.1)	1.3 (.1)
Intent 3	4.2 (.0)	1.6 (.2)	4.1 (.2)	1.3 (.0)
Intent 4	4.1 (.1)	1.7 (.3)	4.3 (.0)	1.3 (.0)
Intent 5	4.3 (.2)	1.7 (.3)	4.2 (.1)	1.3 (.0)
Intent 6	4.4 (.1)	2.3 (.5)	4.3 (.1)	1.4 (.1)

sub-optimal policies are still occasionally followed. This indicates that the PPO policy has not yet *fully* converged by the end of these simulations.

Appendix B shows a few examples sampled at the beginning and at the end of the simulations. In some dialogues, the agent learned to solicit a valid slot even when it felt back on asking the user for clarification of intent. The agent has thus learned some original policies which correctly infer slots required for fulfillment *even when the exact intent cannot yet be precisely determined*. This strategy spontaneously emerged by PPO and allows the agent to be more efficient i.e., to complete a dialogue with a smaller number of steps without sacrificing validity and coherence.

4.1.3 Nash Equilibrium in Dialogues Simulated with Multi-Agent PPO

When the assistant and the user are both PPO agents, a Nash equilibrium Silver et al. [2018] is expected because both agents try to maximize the *same* reward function. That is, they both aim to make the dialogue more valid, coherent and efficient. The only difference compared to the single-agent experiment above is that the user is also a PPO policy whose action is to select a prompt within the subset of five prompts predefined for a given intent.

As can be seen in Fig. 3, the agent converges toward optimal policies (reward in range -20 to $+20$) faster than in the single-agent simulations (Fig. 2) where the user selects prompts randomly. A second result observed in multi-agent simulations is that 100% of the policies followed by the end of the PPO simulations have a reward ranging between -20 and $+20$. The standard deviation observed in the final 5,000 dialogues is significantly smaller in Fig. 3 compared to Fig. 2. Table 1 also reports 100% of dialogues complete successfully. Thus, the cooperation of the user has eliminated the sub-optimal policies that were still occasionally observed in the end of the single-agent PPO simulations.

Table 1 also indicates a systematic improvement in *efficiency*, for every intent. It takes 1.3 steps on average to infer the correct intent, compared to 1.8 steps on average in single-agent simulations. This makes sense because when the user sends prompts better understood by the agent, the agent less often needs to ask for clarifications. Appendix B shows some examples of dialogues sampled at the end of multi-agent simulations. These results indicate that the PPO agent has identified reproducible

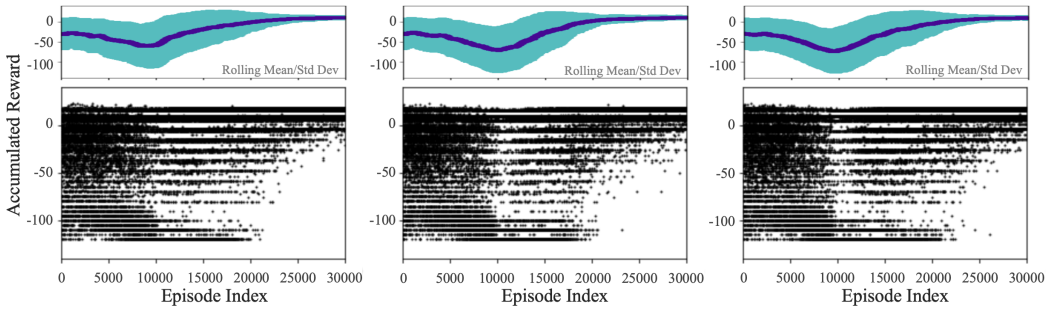


Figure 3: Accumulated rewards across 3×30,000 dialogues sampled with multi-agent RL.

policies to fall back on asking for clarification or infer the correct intent (validity), solicit valid slots (coherence), and minimize the time it takes to fulfill intents (efficiency).

4.2 Analysis of fallback policy optimization by verbal reinforcement learning

4.2.1 Detailed VRL simulation setup

In this third experiment, the ReAct agent and the agent used to generate prompt variations for each query both use Anthropic Claude v3 Sonnet available in Amazon Bedrock as backend LM. The ReAct agent was created using a custom ReAct system prompt and the LangChain react agent library. This agent was trained by simulating episodes of *reflections* which consist in a user prompt followed by reasoning-acting turns $(s_t, a_t, r_{t+1}, s_{t+1})$, where t is the total number of turns including the initial user prompt and every ensuing thought (s_t) and action (a_t) generated internally until the agent decides to respond to the user. Every time the agent selects a tool, which can happen more than once in an episode of reflection, a scalar numerical reward r_{t+1} ($+1$ for valid tool, -1 for invalid tool) is added to the current value of aggregated reward specifically for the original user prompt that initiated the agent's reflection. The latest value (aggregated reward) for every prompt is stored in a look up table and appended as context data to the agent system prompt. The agent is instructed to maximize reward. An ϵ-greedy exploration is reinforced by temporarily setting all values to 0 in 5% of user interactions. User queries and prompts are sampled uniformly at random for 10 epochs. The aggregated reward and number of clarification fallback are reported for each epoch for each prompt in the next section.

4.2.2 Analysis of Reasoning-Acting Generated in Reflections Simulated with VRL

Fig. 4 shows the evolution of accumulated reward and the number of times the ReAct agent decided to fall back on asking for a clarification instead of selecting a tool, in each loop of reflection initiated by a user prompt. Fig. 4 and Table 2 suggest the VRL agent first explores the orchestration environment, then identifies prompts which are too ambigous to identify the correct tool and are worth asking for clarification. For these ambiguous prompts, using only 10 epochs through all prompts and all queries, the reward has become sufficiently negative for the VRL agent to systematically fall back on asking for clarification instead of attempting to select a tool. In other words, the VRL agent learned to follow a policy which is honest, valid and efficient, because it never again attempts to select a tool when it receives a user prompt which it was never able to disambiguate in the past.

This learning by reinforcement after just a few epochs is a typical human behavior in multi-turn conversations, as we naturally behave strategically and efficiently when we are confused by our interlocutor's utterances, instead of nurturing and repeating the same confusion over and over again. In particular, as mentioned in the introduction and illustrated in this experiment, when the original intent of the user is ambiguous "*at first*", but simple enough that it can be disambiguated and perfectly solved after asking for clarifications in the conversation. Given the prompt variation is itself generated by Claude v3, this method can directly scale to a much larger number of prompt variations, queries and tools, in the context of superalignment.

Note that in the setup described in section 4.2.1, asking for a clarification leads to sample a different prompt for the same query, and as can be seen in Fig. 4 there are at most three ambiguous prompts out of five possible prompts for every query. On average, the correct tool is selected in < 2 steps across

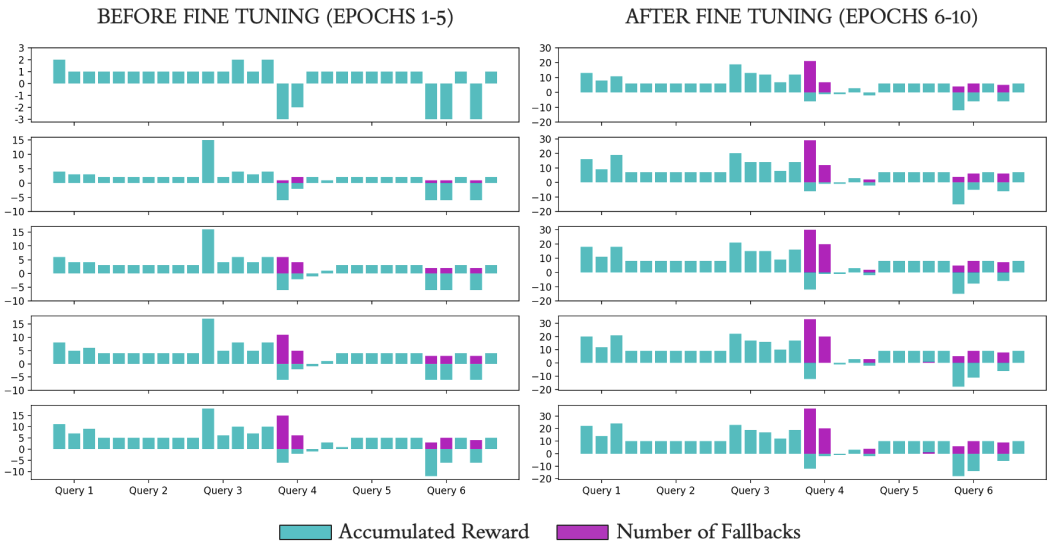


Figure 4: Accumulated reward and number of fallbacks across 10×30 episodes of reflections sampled with VRL. The x-axis corresponds to the five prompts generated by Claude for each of the six queries.

Table 2: Validity and efficiency of sampled reflections.

	Before Fine Tuning		After Fine Tuning	
	Reward	Fallback	Reward	Fallback
Query 1	1.2 (.1)	0.0 (.0)	6.4 (1.7)	0.0 (.0)
Query 2	1.0 (.0)	0.0 (.0)	5.0 (.0)	0.0 (.0)
Query 3	1.4 (.2)	0.0 (.0)	9.4 (3.5)	0.0 (.0)
Query 4	-0.4 (1.7)	0.0 (.0)	-1.6 (2.1)	4.2 (5.0)
Query 5	1.0 (.0)	0.0 (.0)	5.0 (.0)	0.1 (.1)
Query 6	-1.4 (1.9)	0.0 (.0)	-2.0 (5.6)	2.2 (1.8)

all sampled episodes of reflections. These results are quantitatively similar to the results obtained in section 4.1 for different orchestration environments using PPO.

5 Conclusion

Optimization of the clarification fallback policy was successfully applied in three different reasoning-acting orchestration tasks, using PPO to fine tune model parameters, or using VRL to improve tool retrieval by reflecting on past sampled rewards. In all experiments, after fine tuning the new policy systematically identifies the correct intents and tools in < 2 steps in over 99% of sampled dialogues. This implies the new policy has learned to strategically fall back on asking for clarification of intent in ambiguous cases, instead of proceeding with a potentially wrong intent, and successfully completes dialogues after clarifying the user intent.

The method proposed in this paper defines a reward model as a mapping between predefined queries and actions. As a result, it can directly fine tunes the validity, coherence and efficiency of LM policies and reasoning agents in multi-turn conversations, in contrast to RLHF, RLAIF and DPO methods in common usage. This method can be applied to any custom orchestration environment.

The method's main limitation is the need to operate in a predefined orchestration environment, where all possible queries and intents are known and well-defined. But human preferences have many facets including mixed and unclear intents, and intents changing over time. The method is unlikely to perform well in environments where user intents can be mixed or unclear. But we do not see this as prohibitively limiting because most customer assistants (e.g., Alexa, Siri) do operate in the confine of a finite set of queries and possible actions. And even in the longer-term where LM agents could reach

superhuman capabilities, aligning them will likely benefit from restraining their control to a finite set of actions chosen by humans, or at least by some form of AI constitutions (Bai et al. [2022]).

The VRL table lookup approach was extremely efficient in a small orchestration environment (Appendix C-D) and did not need LM parameter fine tuning, but is unlikely to scale to very large numbers of queries and tools. We plan to scale this method by replacing the table lookup approach by function approximation, as typically done in the context of deep reinforcement learning.

A separate adversarial agent was used to generate prompt variation around each query, which increases robustness of the learned policies and enhances generalization. When the adversarial agent is also a PPO policy incentivized to maximize the same reward as the assistant, the cooperation between the user and the assistant led to better dialogues. In practice, human users do not sample prompt uniformly at random so the cooperation may be more indicative of actual performance in online setups. Humans have been conditioned, through evolution and a lifetime of reinforcement, to adapt utterances (i.e., prompts) when it is self-beneficial to do so.

When the reasoning-acting agent needed to take two actions simultaneously (results reported in section 4.1), the agent learned an original policy that asks for some precise and valid context data even when the exact intent is not yet precisely known i.e., partially understood but too ambiguous to identify the exact intent. This strategy spontaneoulsy emerged during the PPO simulations, without human feedback, and so showed potential for superalignment (Bai et al. [2022], Lee et al. [2023]). The PPO policy found a way to increase efficiency without sacrificing quality and coherence.

These results demonstrate that policy optimization of language models, with or without parameter fine tuning, can be used to align the validity and efficiency of reasoning agents in multi-turn conversations by fine tuning LM's abilities to decide when it is best to ask the user for clarifications.

References

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*, 2023.

Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. Deep reinforcement learning for dialogue generation. *arXiv preprint arXiv:1606.01541*, 2016.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, and T. Lillicrap. A general reinforcement learning algorithm that masters chess, shogi and go through self-play. *Science*, 362 (6419):1140–1144, 2018.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021, 2020.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, and Petko Georgiev. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

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

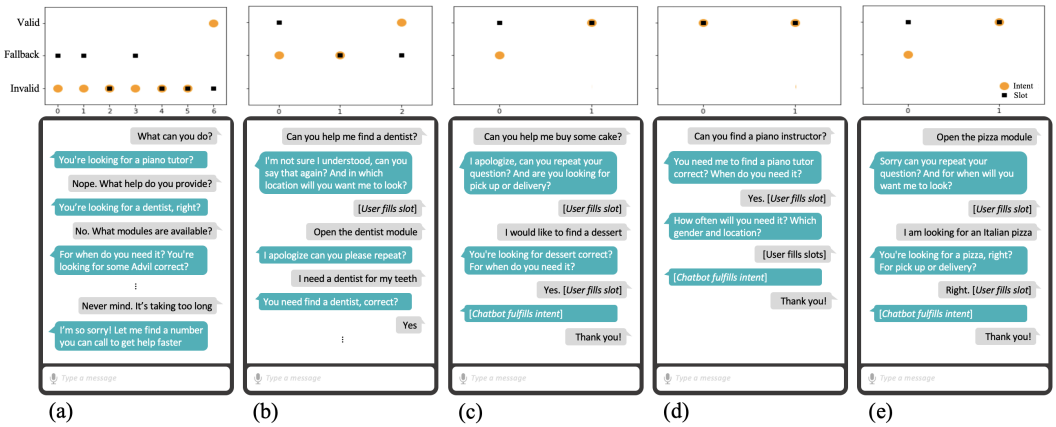
A Appendix / supplemental material

Optionally include supplemental material (complete proofs, additional experiments and plots) in appendix. All such materials **SHOULD be included in the main submission.**

INTENT	CORRESPONDING SLOTS
/NAVIGATION	NO SLOT
/TUTOR/PIANO	LOCATION, AGE, GENDER, WHEN, HOW OFTEN
/DOCTOR/DENTIST	LOCATION, WHEN
/PHARMACY/ADVIL	LOCATION, WHEN, HOW OFTEN, PICKUP OR DELIVERY
/FOOD/PIZZA	WHEN, PICKUP OR DELIVERY
/FOOD/DESSERT	WHEN, PICKUP OR DELIVERY
/CLARIFICATION	NO SLOT

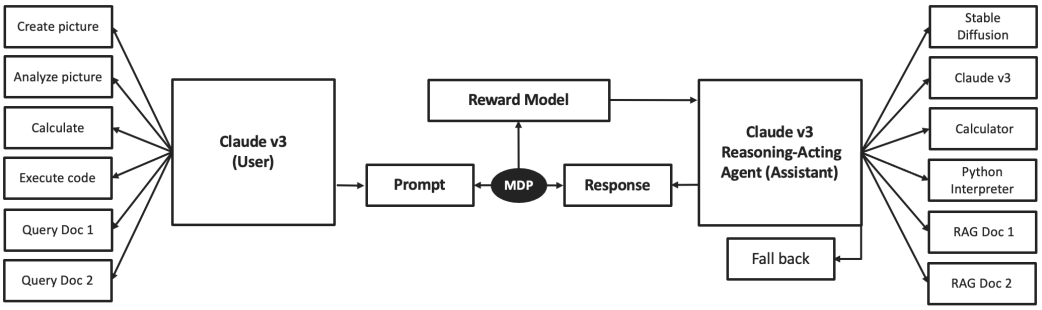
A Appendix A.

Map of intents and slots defining the custom orchestration environment in the two PPO fine tuning experiments (sections 3.1-3.2). The complete JSON file mapping all prompts to all intents and slot request (resulting in approximately 1230 possible combinations of prompts and slot requests for the RL agent to explore i.e., 7 intents $\times$ 5 prompts $\times$ 7 slot requests $\times$ 5 slots) is available upon request to the author.



B Appendix B.

Sample of user-PPO agent interactions before training (a), after an optimal policy has been identified by single agent PPO (b, c), and after an optimal policy has been identified by multi-agent PPO (d, e). The validity of every action taken at every step by the PPO agent is shown above each sampled dialogue.



C Appendix C.

Map of queries and tools in the custom orchestration environment used to define the reward model for VRL in the third experiment (section 3.3).

D Appendix D.

Prompts generated using Claude v3 Sonnet in Amazon Bedrock for all queries in the custom orchestration environment used for the VRL experiment (section 3.3). Most prompts were truncated to fit into the paper format; a JSON file mapping complete prompts (i.e., not truncated) for all queries is available upon request to the authors.


```
{"query": 1,
"prompts": [
"Please generate an image depicting several dogs running on a beach",
"I would like you to create a picture with multiple dogs playing on the...",
"Can you make an image showing a few dogs running and playing on the...",
"Create a drawing of multiple canines racing across the sand close to...",
"Produce an illustration displaying numerous dogs sprinting near the tide..."
]},
{"query": 2,
"prompts": [
"On the given picture, determine the number of dogs that are running on...",
"Please examine the provided image and count the quantity of dogs that...",
"Can you inspect the shown illustration and enumerate how many canines...",
"I need you to visually analyze the displayed graphic and tally the...",
"Kindly check the presented picture and calculate the total number of..."
]},
{"query": 3,
"prompts": [
"Calculate the result of the mathematical expression: 5 plus 5 plus 10",
"Can you determine the solution when 5 is added to 5 and then 10 is added...",
"I would like you to compute the total of 5 added to 5, with 10 added to...",
"Find the end quantity when 5 plus 5 is summed with 10",
"Derive the final value when 5 plus 5 plus 10 are combined together..."
]},
{"query": 4,
"prompts": [
"Calculate the result of the expression 5 plus 5 plus 10.",
"With i initially 0, iterate four times over this loop: i = i + 5, and tell...",
"Let i begin at 0. Repeat this process four times: add 5 to the current...",
"Initialize i to 0. Perform the following operation four times: i = i + 5. ...",
"With i initially assigned the value 0, cycle through this loop four..."
]},
{"query": 5,
"prompts": [
"What process did Anthropic use to adapt and adjust the harmful...",
"Please describe Anthropic's methodology to specialize and optimize the...",
"Can you explain the technique Anthropic leveraged to refine and enhance...",
"I would like you to delineate the approach taken by Anthropic to tune ...",
"How did Anthropic fine tuned the toxicity of their LLM?"
]},
{"query": 6,
"prompts": [
"What methods did Anthropic utilize to expand and amplify the precision...",
"Please discuss Anthropic's tactics to broaden and multiply the selective...",
"Can you clarify the procedures used by Anthropic to scale up and increase...",
"Describe how Anthropic was able to grow and augment the customized...",
"How did Anthropic scaled the fine tuning of the toxicity of their LLM?"
]}
```

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]

   : The abstract and introduction explain a key limitation of methods known as *reinforcement learning from human preferences* now commonly used to fine tune language models, and propose a method to complement (not replace) these methods specifically to improve the clarification fallback behavior in multi-turn conversations. It defines a simple reward function specifically for a predefined orchestration environment and can scale to any such environment. The main limitation which is to only apply when the orchestration environment is known in advance is clearly indicated. The rest of the paper demonstrates this claim by applying the method in multiple and very different orchestration environments.

   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: The main limitation is indicated in the introduction (need to operate in a predefined environment) and developed in more details (lines 305-312) in the conclusion together with additional limitations and potentail future work to address it (lines 313-316).

   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: [Yes]

   Justification: Even though the theoretical aspect of the proposed method is very simple relative to the methods commonly used nowaday for *reinforcement learning from human preferences*, for which multiples references are provided in the paper, a few assumptions are made and developed in the main text, in particular on lines 83-93, as it is assumed that modern language models are good enough to disambiguate some user prompts for any given user query, but not all of them, which create the need to fine tune by asking for clarification to the user. This assumption seems fair given the human-level capabilites of modern language models and the typically very high number of possible ways that a user, in a given natural language, may formulate a query.

   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: All experiments are explained in the methodology section, and results are also introduced with a sub-section describing the detailed simulation setup for each experiment. Finally, more details are provided in Appendix including a statement proposing to provide the complete and exact data files upon request to the authors.

   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: All experiments are explained in the methodology section, and results are also introduced with a sub-section describing the detailed simulation setup for each experiment. More details are provided in Appendix including a statement proposing to provide the complete and exact data files upon request to the authors. Code can also be provided upon reasonable request to the authors.

   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: All experiments are explained in the methodology section, and results are also introduced with a sub-section describing the detailed simulation setup for each experiment. More details are also provided in Appendix.

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: Detailed statistics including error bars on multiple metrics are provided for every experimental result.

   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: Compute resourses are explained in details, in particular for the PPO experiments which require significant compute resources (lines 201-208).

   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: In particular, all data used in all experiments is custom-made due to the nature of the method proposed which relies on pre-defining a custom orchestration environment, and thus present no conflict of interest.

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: The paper proposes a method explicitly designed to create positive societal impact by reducing hallucination and maximizing honesty of reasoning agents based on language models. Some emergent strategies learned by the agent during PPO are reported (lines 228-233) and discussed in the context of AI superalignment efforts (lines 324-329).

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: [Yes]

Justification: All data used in all experiments is custom-made and presents no risk for misuse. The code itself is not publicly released, instead it can be shared upon reasonable request to the authors.

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: All data used in all experiments is custom-made due to the nature of the method proposed which relies on pre-defining a custom orchestration environment, and thus present no conflict of interest.

    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: The method including detailed simulation setups are clearly documented in the paper. The code itself is not publicly released but can easily be shared upon reasonable request to the authors.

    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: The proposed method does not require human subject nor human feedback, although the reward function could be extended to include explicit human feedback as in RLHF. But the main proposition is a form of RLAIF where the reward model relies exclusively on a deterministic map between queries and tools (or intents and actions) defined by the orchestration environment, augmented with a language model to create prompt variation and increase potential for scaling and generalization. So it explicitly proposes a superaligment method that does not directly use human feedback.

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: As explained above, this paper proposes a superaligment method that does not involve human subjects.

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.