Planning without Search: Refining Frontier LLMs with Offline Goal-Conditioned RL

Joey Hong **Anca Dragan** **Sergey Levine**
UC Berkeley
{joey_hong,anca,sergey.levine}@berkeley.edu

Abstract

Large language models (LLMs) excel in tasks like question answering and dialogue, but complex tasks requiring interaction, such as negotiation and persuasion, require additional long-horizon reasoning and planning. Reinforcement learning (RL) fine-tuning can enable such planning in principle, but suffers from drawbacks that hinder scalability. In particular, multi-turn RL training incurs high memory and computational costs, which are exacerbated when training LLMs as policies. Furthermore, the largest LLMs do not expose the APIs necessary to be trained in such manner. As a result, modern methods to improve the reasoning of LLMs rely on sophisticated prompting mechanisms rather than RL fine-tuning. To remedy this, we propose a novel approach that uses goal-conditioned value functions to guide the reasoning of LLM agents, that scales even to large API-based models. These value functions predict how a task will unfold given an action, allowing the LLM agent to evaluate multiple possible outcomes, both positive and negative, to plan effectively. In addition, these value functions are trained over reasoning steps rather than full actions, to be a concise and light-weight module that facilitates decision-making in multi-turn interactions. We validate our method on tasks requiring interaction, including tool use, social deduction, and dialogue, demonstrating superior performance over both RL fine-tuning and prompting methods while maintaining efficiency and scalability. [1]

1 Introduction

Large language models (LLMs) have demonstrated impressive performance across a wide range of reasoning and problem-solving tasks [38, 11, 43]. However, many complex tasks, such as goal-oriented dialogue, social interaction, and negotiation require LLMs to engage in complex multi-turn interactions, where the LLM agent might need to act strategically, anticipating the long-horizon outcomes of its decisions, while still adapting to unexpected responses. In such settings, an effective agent might take actions that are not immediately rewarding, but instead lay the ground for future success by either eliciting information or changing the interlocutor's disposition. For example, an agent may ask clarifying questions to gather crucial information or attempt to persuade an interlocutor to shift their opinion, ultimately enabling the agent to achieve its objective. The space of possible outcomes when planning out such complex behaviors is enormous, and enabling such agents to act intelligently and strategically presents a major algorithmic challenge.

A traditional approach to equipping LLMs agents with such planning capabilities involves fine-tuning them using multi-turn reinforcement learning (RL) [33, 4]. While RL can in theory help LLMs optimize their behavior across extended interactions, the approach does not scale well to frontier models like `GPT-4` [22] or `Claude 3` [1]. In particular, RL algorithms are challenging due to their complexity to train and need for high number of samples, both of which make training large LLMs

[1] Website can be found at `https://jxihong.github.io/pnlc_website/`

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

prohibitively expensive. Consequently, while RL has achieved some successes in multi-turn settings, its scalability limitations hinder broader adoption. Because of this, recent methods that leverage frontier models as LLM agents do not perform RL fine-tuning, but during inference, prompt the agent to simulate future trajectories to guide decision-making [44, 47].

To address these challenges, we propose a novel approach that leverages offline RL to significantly improve an LLM agent's reasoning and planning, *without directly training the LLM agent*. Our key insight is that at rather than learning a policy, we instead learn a *natural language critic* that can be used by an LLM agent to evaluate potential actions at inference-time. Specifically, we train a goal-conditioned value function from offline data that predicts the likelihood of achieving various outcomes given the current state and a proposed action. At inference, the critic uses these values to assess future outcomes, which the LLM agent can use to effectively and efficiently guide its planning.

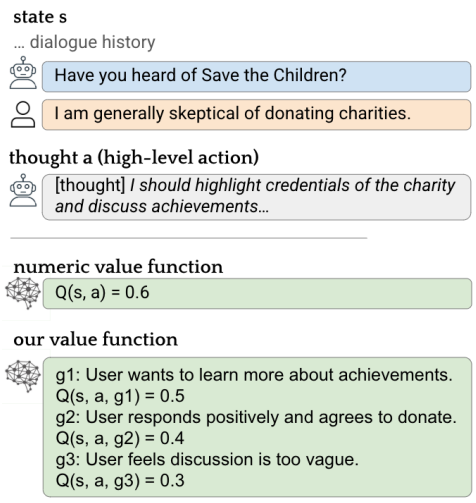

Figure 1: In an ongoing goal-oriented dialogue, we learn natural language value over the internal reasoning steps of an LLM agent. The value analyzes future positive and negative outcomes, and allow the LLM agent to refine its reasoning.

Unlike standard scalar-based value functions, our natural language value functions can provide predictions about a wide variety of potential outcomes, allowing the agent to compare different strategies along many dimensions. Figure 1 illustrates this concept in a persuasion task, where the agent evaluates a potential response based on its likely impact. In the example, the LLM agent learns that by using a credibility appeal, there is 40% probability that the user feels assured and responds positively, but a 30% change the user does not trust the LLM agent for being too vague. These predictions can be processed by the LLM agent to determine which of the available strategies would be most successful.

We further make a number of design decisions that provide for a computationally efficient and scalable method. First, rather than evaluating low-level utterances, we operate on high-level strategies, or thoughts, akin to chain-of-thought reasoning [38]. This abstraction significantly reduces the complexity of the decision space while preserving essential information. In this hierarchical design, our natural language critic assists the LLM agent in planning over high-level strategies, which can then be used to generate actions in the environment. By leveraging the critic, our approach also eliminates the need for any inference-time search, significantly improving its practicality on time-sensitive tasks. Instead of existing methods rely on feedback from inference-time search, we simply require evaluations by our natural language critic on both positive and negative hypothetical futures –or desirable and undesirable future occurrences to accomplishing the task at-hand, respectively.

Finally, we validate our method across diverse LLM benchmarks requiring interactive decision-making, including web navigation, social deduction games, and persuasion via dialogue. Our approach consistently outperforms state-of-the-art techniques based on fine-tuning with multi-turn RL and inference-time search with self-refinement. Furthermore, it achieves these improvements with significantly lower computational costs, making it a scalable solution for real-world applications.

In summary, our contributions are threefold: (1) we introduce goal-conditioned value functions that model the probabilities of various outcomes, enabling LLMs to reason about long-term effects of their actions; (2) we apply these value functions at the level of high-level thoughts rather than low-level utterances, reducing the complexity of decision-making; and (3) we introduce a novel inference-time process where the LLM agent considers both positive and negative future outcomes to refine its decisions. By integrating these innovations, we provide an effective framework for LLM agents to adaptively reason and plan in multi-turn interactive settings. Compared to existing state-of-the-art results, our method scales to the largest frontier models with very lightweight training, and does not require a substantial inference budget.

2 Related Work

Our method combines concepts from LLM agents, goal-conditioned reinforcement learning, and test-time prompting, reasoning, and planning methods for LLMs. We summarize the most relevant prior works below, relating them to our approach.

Large language models. Language models have shown impressive capabilities in text generation [6, 14, 8, 28, 40], translation [7], question answering [27], summarization [24, 39, 3], and code generation [5, 46]. Transforming LLMs into interactive agents capable of multi-step decision-making has been a recent focus of research. These embodied agents interact with environments by taking actions, observing responses, and adapting their strategies accordingly. Applications include robot learning [32, 19], web navigation [20], and text games involving dialogue [41, 31, 47]. Our work falls in the realms of training better LLM agents by proposing a novel, more scalable approach.

RL for LLMs. Reinforcement learning (RL) has been proposed to improve the performance of LLMs to much success. The most notable has been to use policy gradient methods to fine-tune LLMs from human or AI feedback [2, 23, 36]. Such online methods have also been applied to train LLM agents in interactive settings, such as web search [20] and embodied navigation [45]. In contrast, offline RL methods have also been used to train LLM agents without the need for online samples [34, 4]. However, RL methods require a high number of samples, and are notoriously difficult to train effectively; therefore, multi-turn RL has not been successfully scaled to frontier LLMs such as `GPT-4` [22]. We also employ multi-turn RL, but we make several key innovations that make our approach much more scalable: (1) we consider RL over high-level strategies rather than environment actions; (2) we consider goal-conditioned RL where rather than learning values, we learn likelihoods of reaching particular goal states; and (3) we do not directly fine-tune an LLM, but rather learn a light-weight auxiliary value function that can be used to guide the search of the base LLM agent using only inference APIs. These innovations make our method practical to use for frontier LLMs.

Reasoning and planning with LLMs. The latest frontier models have shown that strong reasoning and planning capabilities can be elicited by different prompting techniques [38, 11]. Such techniques are central to the design of modern LLM agents. Notably, ReAct leverages chain-of-thought prompting [38] to allow LLM agents to generate a composite of reasoning and task-specific actions [41]. However, ReAct relies on the base intuition of LLMs, which fails in more complex tasks such as web search or dialogue. More recent approaches have proposed self-supervised methods for LLM agents to enhance their reasoning capabilities [31, 18, 47]. Reflexion adds a step of self-reflection to ReAct allowing LLM agents to refine their initial reasoning after some environment feedback [31]. State-of-the-art methods employ tree-based search such as Monte Carlo tree search (MCTS) to guide the reasoning and planning of LLM agents [44, 47, 26]. These methods improve upon ReAct by leveraging the knowledge gained from search to improve the LLM agent's reasoning capabilities. However, we posit that limited search is not sufficient for LLM agents to make *data-driven* decisions; existing approaches are limited by the amount of data agents can observe due to the prohibitive cost of inference-time search. In our work, we use offline RL to train value functions that can guide reasoning without the need for search during inference, resulting in efficient and data-driven agents. In contrast to explicit search, another popular paradigm for improving reasoning is self-refinement [9], where an LLM assesses its own generations intrinsically. Our method also involves refinement, but via an auxiliary goal-conditioned value function.

3 Preliminaries

Markov decision processes. We use the formalism of the Markov decision process (MDP), given by a tuple $M = (\mathcal{S}, \mathcal{A}, P, r, \rho, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, P is the transition function, r is the reward function, ρ is the initial state distribution, and γ is the discount factor. When action $a \in \mathcal{A}$ is executed at state $s \in \mathcal{S}$, the next state is sampled $s' \sim P(\cdot|s, a)$, and the agent receives reward r with mean $r(s, a)$. Note that in many interactive settings, the state is actually partially observed, resulting in a POMDP instead of a MDP. For example, in dialogue, the state consists of the internal thoughts and intentions of other participants than the agent. However, prior analyses has shown that the two definitions can be made equivalent by instead defining states in the MDP as histories of observations so far, or in dialogue, the history of all utterances thus far [35].

LLM agents in MDPs. Tasks considered by LLM agents can be defined under this formalism as follows. At timestep t, the agent *state* s_t of the MDP consists of the history of interaction thus far. In

a dialogue, this will include the agent's past utterances, as well as responses by all other participants. Following the ReAct agent by Yao et al. [41], the agent *action* a_t is actually composite, consisting of a *thought* a_t^{tht} where the agent performs a reasoning step, followed by the actual environment action a_t^{env}. In dialogue, the thought could be the high-level strategy or plan the agent aims to execute, whereas the environment action is the actual utterance by the agent. Then, the transition function appends the environment action a_t^{env} as well as new observations by the environment o_t, such as responses by other interlocutors, to form the next state s_{t+1}.

Reinforcement learning. The objective of RL is to find a policy π that maximizes the expected discounted return $\mathbb{E}_{\tau\sim p^\pi(\tau)}\left[\sum_{t=0}^{T-1}\gamma^t r(s_t,a_t)\right]$ in an MDP, where $\tau=(s_0,a_0,s_1,a_1,\ldots,s_T)$ and $p^\pi(\tau)=\rho(s_0)\prod_{t=0}^{T-1}\pi(a_t|s_t)P(s_{t+1}|s_t,a_t)$. The Q-function $Q^*(s,a)$ represents the discounted long-term reward attained by executing a given state s and then behaving optimally afterwards. Q^* satisfies the Bellman optimality equation:

$$Q^*(s,a)=r(s,a)+\gamma\mathbb{E}_{s'\sim P(\cdot|s,a)}\left[\max_{a'}Q(s',a')\right].$$

A policy can be extracted from the Q-function, i.e. $\pi^*(a|s)=1[a=\arg\max_{a'}Q^*(s,a')]$. In offline RL, we want to estimate Q-function from a dataset $\mathcal{D}$ comprising of samples (s,a,s',r), where each sample corresponds to one timestep of interaction, and a come from a behavior policy $\pi_\beta(\cdot|s)$ (which might correspond to a mixture of multiple policies). Kostrikov et al. [12] propose Implicit Q-learning (IQL), which trains Q-values $\widehat{Q}(s,a)$ and state-values $\widehat{V}(s)$ with the following loss functions:

$$L_Q=\mathbb{E}_{(s,a,s',r)\sim\mathcal{D}}\left[\left(r+\gamma\widehat{V}(s')-Q(s,a)\right)^2\right],\quad L_V=\mathbb{E}_{(s,a)\sim\mathcal{D}}\left[L_2^\tau\left(\widehat{Q}(s,a)-V(s,g)\right)\right],$$

where L_2^τ is the expectile loss with hyperparameter $\tau\in[0.5,1)$.

4 PNLC: Planning with a Natural Language Critic

Here, we describe our proposed method, which we dub Planning with a Natural Language Critic (PNLC). PNLC provides a general approach for training LLM agents, using only a small prior dataset of trajectories. During the training phase, our method uses this dataset to train a goal-conditioned value function that can predict the probability of different interaction outcomes given the agent's proposed action. At inference time, we utilize this goal-conditioned value function as a natural language critic, which helps an LLM agent refine its strategy thought at each step of interaction, via querying the value function for the probability of different outcomes and using these probabilities to perform self-refinement over possible thoughts.

An overview of our approach can be found in Figure 2. During offline training, we aim to learn a goal-conditioned value function $Q(s,a^{\mathsf{tht}},g)$. Here, the goal g is a possible future state that can result from following thought a^{tht} in state s, and $Q(s,a^{\mathsf{tht}},g)$ denotes the likelihood of reaching the goal future state g. Then, during inference, we can evaluate thoughts by an LLM agent by analyzing possible futures. Specifically, by sampling possible positive and negative future states and providing their likelihoods, an LLM agent can self-refine its original thought. We describe each stage in greater detail below, but defer specifics such as training hyperparameters and prompts used to Appendix A.

4.1 Training a Goal-Conditioned Value Function

Our proposed natural language critic relies on a goal-conditioned value function to evaluate actions in lieu of inference-time search. As input, we require a dataset of task-specific trajectories $\mathcal{D}$ by any prior agent. The agent does not need to be near-optimal, but only exhibit a variety of different, even incorrect, strategies [13]. In our experiments, we use an LLM agent with chain-of-thought prompting, which are known to exhibit suboptimal performance in tasks requiring long-term planning.

Recall that we evaluate only the thought component of the agent action. This is because we view the thought as an abstraction of the full action that contains all the important information to plan over. For example, in Figure 1, the high-level thought, which was to appeal to the credibility of the organization, captures the semantics of the agent's next utterance without the noise of linguistic structure or syntax.

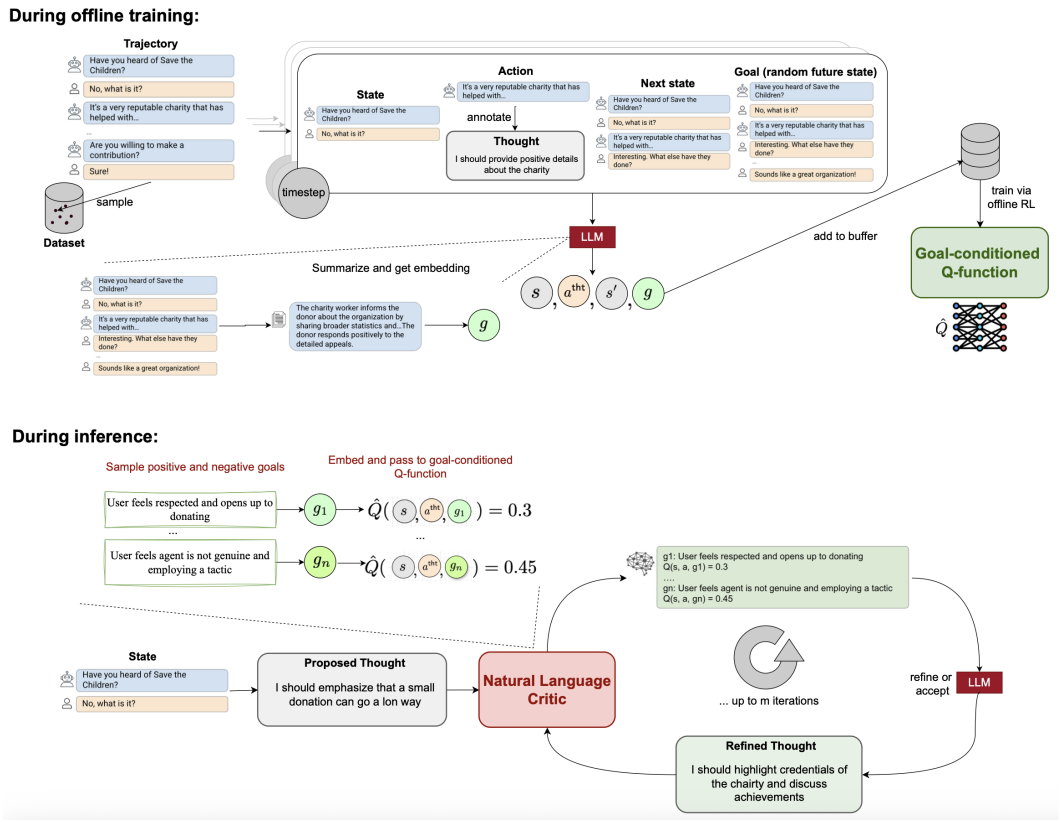

Figure 2: During *offline training*, training samples are summarized and embedded, and a goal-conditioned value function (which is a light-weight 2 hidden-layer MLP) is trained over the embeddings. During *inference*, a natural language critic uses the value function to produce an informative analysis of possible future outcomes. This natural language value is used by the LLM agent to refine its proposed reasoning.

From each trajectory in the dataset we extract training samples $(s, a^{\text{tht}}, s', g)$ consisting of state, thought, next state, and a goal that is a random future state in the trajectory. Note that following the formalism in Section 3, the states are all histories of interaction with the environment, and a goal adds possible future interactions to the original state. To make learning more efficient, we consider the following simplifications to the training process:

(1) *Summarizing trajectories:* Rather than considering full histories of interaction, we instead summarize them into compact descriptions. Though the contents of the summaries are task-specific, we design them to capture all the relevant information needed to make decisions regarding the next thought or strategy to adopt. For example, in dialogue, the summarizations aim to describe the thoughts of the participants in the discussion history rather than their actual utterances. We provide example summarization prompts in Appendix A.

(2) *Embedding language descriptions:* Finally, rather than learning value functions over natural language, either as thoughts or summaries of histories, we instead consider their low-dimensional embeddings that are output by an LLM. Because LLMs have a basic grasp of how to parse natural language, we view the embeddings as sufficient to learn over. This has the added benefit of our learned value functions being lightweight architectures, rather than full Transformers, which are proven to be slow to train.

The training samples are used to learn a goal-conditioned Q-value $\widehat{Q}(s, a^{\text{tht}}, g)$ by modifying the loss function of the IQL algorithm [12] to condition on goals:

$$L_Q = \mathbb{E}_{(s,a^{\text{tht}},s',g)\sim\mathcal{D}}\left[\left(r(s,g)+\gamma\widehat{V}(s',g)-Q(s,a^{\text{tht}},g)\right)^2\right], \; L_V = \mathbb{E}_{(s,a^{\text{tht}},g)\sim\mathcal{D}}\left[L_2^\tau\left(\widehat{Q}(s,a^{\text{tht}},g)-V(s,g)\right)\right],$$

where $r(s, g)$ is a binary indicator for whether state s is the goal state g. At convergence, $\widehat{Q}(s, a, g)$ should predict the likelihood of reaching goal g after taking action a at state s.

An illustration of the offline stage of our approach can be found in Figure 2. Note that though the summarization and embeddings require calls to an LLM, because they do not require particularly sophisticated reasoning, we are able to use cheaper LLM models to perform this step. In practice, we use GPT-3 [21] to extract embeddings, rather than GPT-4 [22] that is used in the final decision-making process. Also note that because our goal-conditioned Q-function does not take natural language inputs but rather embeddings, our learned Q-function is simply a MLP with 2 hidden layers rather than a full Transformer.

4.2 Planning using a Natural Language Critic

Now, we describe how our LLM agent leverages the goal-conditioned value function learned offline as a natural language critic for complex decision-making. Naively, the LLM agent could simply use the value function to take the action with the highest probability of reaching a desired goal. However, LLM agents historically perform better when allowed to refine their actions using feedback [31, 18]. This is typically done post-hoc using trajectories from earlier attempts or from simulation. We instead use the natural language critic as a way to provide feedback over proposed actions, so that the LLM agent can refine its actions without the need to gather any trajectories during inference.

Specifically, the natural language critic takes state s and proposed thought a^{tht}, and performs the following operations. First, a LLM generates n hypothetical positive and negative goals $(g_1, g_2, \ldots, g_n)$ based on the current state. Each goal represents a possible future outcome of the agent's proposed thought. Then, we feed the quantities, including the sampled goals, to our learned goal-conditioned value function, which for each goal outputs a scalar quantity that is the likelihood of reaching the goal. The resulting goals $(g_1, \ldots, g_n)$ and corresponding values $(\widehat{Q}(s, a^{\mathsf{tht}}, g_1), \ldots, \widehat{Q}(s, a^{\mathsf{tht}}, g_n))$ are combined into a natural language value. The natural language value, compared to scalar-based values, is much richer in information and also greatly improves interpretability. Furthermore, the inclusion of negative outcomes allows the LLM to better correct its proposed thoughts.

The feedback provided by the natural language critic can be viewed as a compact summarization of a plethora of possible future trajectories, much more than can be obtained via inference-time search. Hence, the natural language critic can co-opt the traditional search procedure in traditional LLM reasoning processes. Namely, given a proposed thought and natural language value, the LLM agent is optionally allowed to suggest an improvement to its original thought, up to a total of m iterations of refinement. In practice, we find that $m = 2$ with only one additional round of refinement was sufficient to achieve good performance across our evaluated tasks. See Figure 2 for an illustration of the inference stage of our approach. We will show later that our method not only profits from efficiency but is also more effective than existing approaches across a variety of benchmarks.

5 Experiments

To demonstrate the effectiveness of PNLC, we evaluate our method on a variety of multi-turn LLM agent benchmark tasks: web shopping [42], social deduction games [16], and persuasion [37].

5.1 Task Descriptions

All tasks we evaluate on are illustrated in Figure 3. In this section, we provide descriptions for each task, as well as how training data was generated.

Web Shopping. WebShop [41] is an online shopping environment where an agent processes unstructured text data (in the form of descriptions crawled from Amazon) to purchase a product given some initial user specifications. At the end, the agent receives a reward between 0 and 1 depending on the similarity between the purchased and ground-truth desired item. The performance is gauged using two metrics: the score, reflecting the average reward received, and success rate, indicating the frequency with which the chosen product fulfills all specified conditions. All relevant methods train on the same dataset consisting of 12k initial user instructions, of which we randomly held out 100 for evaluation. With the remaining instructions, we generate a dataset of trajectories in which we

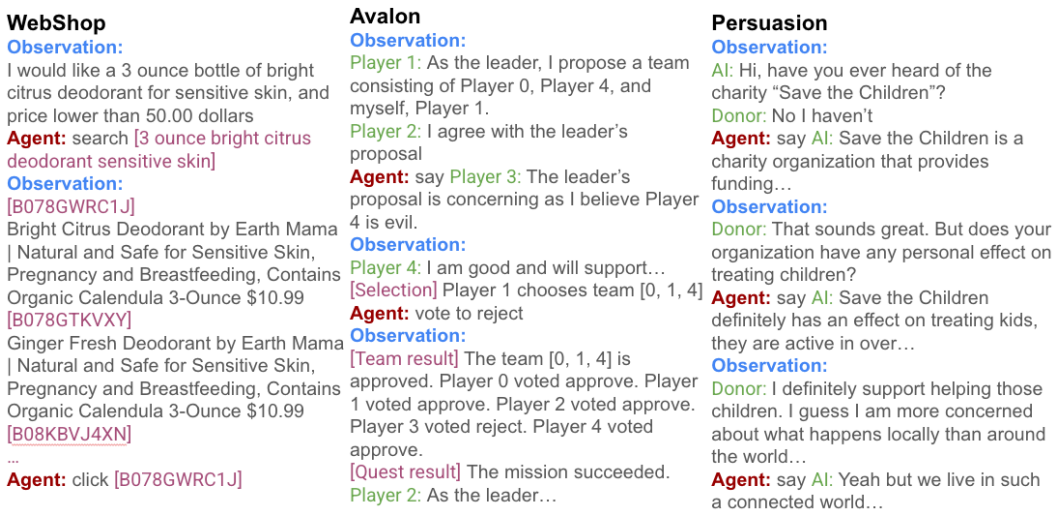

Figure 3: The various tasks we consider, spanning tool use, games, and goal-oriented dialogue.

simulate a suboptimal agent by prompting `GPT-3.5` with few-shot examples similar to that done by Yao et al. [41].

Social Deduction. AvalonBench [16] is a testbed for LLM agents modeled after the team-based discussion game Resistance Avalon, where agents must jointly deceive other agents while deducing their roles, all through natural language dialogue. Each game consists of 5 players, 3 of which are on the good side and the remaining 2 evil. A notable asymmetry of this game is that all but one of the good players do not know other players' identities, while evil players are aware of who their teammates are. The one exception is the role *Merlin*, which the LLM agent will play due to being the hardest role in the game. Merlin knows the identity of the evil players, and must guide the other good players whilst maintaining his own identity a secret; if his identity is discovered by the evil players, they are allowed to assassinate him, and the evil side wins the game. The LLM agent wins if the good side succeeds in at least 3 of 5 rounds, where a successful round means a subset of players chosen by a designated leader for the round all vote to accomplish a mission. Light et al. [16] implemented baseline agents for all roles in the game, which use `GPT-3.5` [21] for discussion and a heuristic for action selection. We simulate 2.5k trajectories using these baseline agents, in which the baseline Merlin achieves a 21.4% win rate.

Persuasion. We consider a goal-oriented dialogue task, inspired by Wang et al. [37]. In this task, an LLM agent must persuade a participant to donate to Save the Children, a non-governmental organization dedicated to international assistance for children. At the end, the participant can choose to donate up to $2 total. Because the donation amount is a very noisy signal, following prior work [25], we model potential donors by prompting `GPT-3.5` [21] to generate responses. To ensure that intelligent persuasion is required to complete the task successfully, the simulated donor is prompted to be skeptical of donating to charities. In our setup, the simulated user interacts with the LLM agent for up to 10 turns of dialogue, then must choose an amount to donate. We college 2.5k dialogues using naive prompting of `GPT-3.5` [21] as the data collection agent. The average donation amount in the dataset is just $\$0.21$.

5.2 Evaluated Methods

For each task, we run PNLC with $n = 4$ goals and a single round $m = 2$ of refinement. We chose $n = 4$ to have a mixture containing 2 positive and negative goals. We evaluate against a range of fine-tuning and prompting baselines; some of these prior methods can be evaluated on the full suite of benchmark tasks, while others are designed specifically for one of the domains. The algorithms can be grouped into the following 3 classes:

RL fine-tuning. To compare to a prior method that uses multi-turn RL, we use two variants of the recently proposed ArCHeR algorithm [48], which uses an actor-critic architecture to directly fine-tune LLMs for interactive tasks. We compare to two variants of a popular fine-tuning algorithm: (1) offline

ArCHer and (2) online ArCHer, which employs interactive rollouts through the environment to further optimize the agent's behavior. Due to the complexity of end-to-end value-based RL methods, we run ArCHeR using the smaller `GPT-2` model [28] to represent the policy, following prior work [48].

LLM reasoning. Next, we evaluate a variety of sophisticated strategies used to invoke reasoning and planning capabilities on a `GPT-4` LLM agent [22], the most complex of which performs inference-time search. We compare to: (1) ReAct [41], which uses chain-of-thought prompting for LLM agents to reason about their actions solely using the zero-shot capabilities of the LLM; (2) Reflexion [31], which adds $n=5$ additional simulations of full trajectories that the LLM agent is allowed to refine with; (3) LATS [47], which performs full MCTS simulation and selects its actions based on the results of the search. We evaluate both a fast and slow version of each method, with either $n=5$ or $n=30$ simulations, to provide a fairer point of comparison for our method (the fast version requires a similar inference budget as ours). Finally, we evaluate a custom adaptation of Huang et al. [9] from question-answering to multi-turn tasks called (4) Self-Plan; rather than using majority vote on n answers by the model, we instead have the model select from $n=5$ of its own generations. In contrast to LATS, Self-Plan does not perform any search, but performs self-refinement by assessing its own generations.

Task-specific design. A number of prior works propose task-specific methods that may involve a mixture of RL fine-tuning and inference-time search. Such methods typically leverage domain-specific knowledge, such as a taxonomy of high-level but task-specific strategies, which do not directly generalize to other tasks. We select and compare against one such method for each task that achieves state-of-the-art performance:

(1) *WebShop:* Putta et al. [26] achieved state-of-the-art performance with their method Agent Q, which mixes offline fine-tuning with inference-time search. In the offline phase, rather than value-based RL, Agent Q uses DPO [29] on preference pairs of trajectories in the dataset. Then, during inference, Agent Q performs MCTS search to generate n simulations before selecting an environment action. Like with LATS, we consider fast $n=5$ and slow $n=30$ variations of Agent Q. Note that because Agent Q requires modifying the weights of the base LLM agent, they train on `Mixtral-7B` [10] as more powerful models such as `GPT-4` [22] only expose inference APIs.

(2) *AvalonBench:* Strategist [17] proposes a hierarchical prompting mechanism that additionally uses MCTS via population-based self-play to propose and refine high-level strategies. The high-level strategies searched over by Strategist are similar to the thoughts refined by our proposed method, but ours does not rely on any self-play during inference. Again, to keep a fair comparison with inference time, we consider both a fast $n=5$ and slow $n=30$ variations of Strategist varying the number of simulations during search.

(3) *Persuasion:* Yu et al. [44] propose GDP-Zero, which prompts the LLM to perform tree-search over possible high-level strategies at every timestep, simulating responses by both interlocutors in the dialogue, then selects the best action according to the search. This method is similar to LATS, but searches over a task-specific persuasion taxonomy rather than the full space of actions. We consider $n=5$ and $n=30$ varations of the approach, varying the number of simulated dialogues.

5.3 Results

A table of all results can be found for all considered benchmarks in Table 1. We see that PNLC performs best across all tasks, the closest competitors being task-specific approaches such as Agent Q in WebShop or Strategist in Avalon. However, it is important to note that such methods use domain-specific prompting mechanisms and do not naively generalize to other tasks; in addition, Agent Q requires DPO fine-tuning on a `Mixtral-7B` LLM, which is much more expensive than the lightweight value function that we train. Furthermore, the $n=30$ version of search approaches requires around $10\times$ more time to make a single decision — 46s for Agent Q compared to 5s for ours in WebShop, and 62s for Strategist compared to 6s for ours in Avalon. Interestingly, though RL fine-tuning approaches such as ArCHer train on the same data as ours, but actually often perform the worst. We attribute this to the fact that standard RL fine-tuning methods rely on a much weaker LLM as the agent; qualitatively, such LLM agent often gives nonsensical or off-topic responses.

	WebShop		Avalon	Persuasion
Method	Score	SR	Winrate	Avg. Donation
ArCHer [48]	62.3	32.0	19.0	0.36
Offline ArCHer [48]	57.3	28.0	18.0	0.31
ReAct [41]	55.1	27.0	21.0	0.54
Reflexion [31]	60.8	29.0	26.0	0.54
LATS ($n=30$) [47]	74.9	44.0	38.0	0.78
LATS ($n=5$) [47]	53.9	28.0	22.0	0.52
Self-Plan [9]	54.8	28.0	27.0	0.55
Agent Q ($n=30$) [26]	77.1	**48.0**	-	-
Agent Q ($n=5$) [26]	63.2	35.0	-	-
Strategist ($n=30$) [17]	-	-	42.0	-
Strategist ($n=5$) [17]	-	-	31.0	-
GDP-Zero ($n=30$) [44]	-	-	-	0.74
GDP-Zero ($n=5$) [44]	-	-	-	0.47
PNLC (ours)	**78.2**	**48.0**	**47.0**	**0.87**

Table 1: Mean evaluation results across all methods and baselines over 100 independent test runs. Across the board, our method PNLC performs best while only using less than 10% the inference budget of the next best approaches.

We give qualitative examples of how our method refines decisions in Figure 4. In Avalon, we see that our method corrects the base LLM agent who was initially going to accuse a player of being evil, which may inadvertently reveal their role to the evil players. Furthermore, in persuasion, the LLM agent using our method changes their strategy to directly address the skepticism in the potential donor, rather than continuing to ineffectively tout the charity's past achievements.

Note that our experimental evaluation considers a diverse variety of methods that use different underlying LLMs as policies, ranging from large modern models such as `GPT-4`, to medium open-source models such as `Mixtral-7B`, to small older models such as `GPT-2`. Unfortunately, rigorously controlling for model size is impractical in such experiments, and wherever possible we chose models that match those actually used in the prior works that we compare to. However, it is worth noting that despite using the largest possible frontier model in `GPT-4`, our method still ends up being faster at inference time than many of the methods we compare to. Nonetheless, this heterogeneity represents a limitation of our experiments.

5.4 Ablation Study

Finally, we conducted an ablation analysis to better understand where the improvements in our method were coming from. We test two important components of our method: (1) offline goal-conditioned training to obtain natural language values, and (2) a novel inference procedure of refining thoughts using these values.

To evaluate the efficacy of each component, we evaluate two modifications of PNLC. The first removes the goal-conditioning, instead learning a scalar Q-value conditioned only state and thought, which simply predicts likelihood of success in the task. Recall that we use goals in our natural language critic in order to create informative descriptions that are more naturally used during LLM reasoning. Now, the LLM agent sees a scalar value instead of natural language, and makes its refinement off that value. The second modification replaces the inference-time refinement procedure with more traditional best-of-n selection [15, 30]. Instead of being given the chance to refine its original thought using feedback provided by the natural language critic, this method instead has the LLM agent generate $n=5$ thoughts, from which one is chosen by the same agent. Note that our refinement procedure was designed with the hypothesis that an LLM agent can produce better thoughts *after* observing what may go wrong, which this modification cannot leverage.

We show results in Table 2. We notice that the two ablations perform comparably to other LLM reasoning baselines such as ReAct that rely only on prompting, meaning the LLM agent does not effectively use the auxiliary learned value function. This shows that both components of our approach — the ability to (1) craft natural language descriptions via goals and (3) refine thoughts using such descriptions — are both crucial to the success of our algorithm.

State:
Quest 1 Summary
[Discussion]
Leader Player 0 proposed Player 0 and Player 1 for the team. Player 1 supported Player 0's suggestion and expressed confidence in being good. Player 4 strongly agreed with Player 0's proposal
[Selection] Player 0 chooses team [0, 1]
[Voting] The team [0, 1] was approved with the following votes: Player 0 (approve), Player 1 (approve), …
[Quest] The mission succeeded
…
Quest 3
[Discussion]
Player 2: For this round, I suggest we stick with a team that has worked before…
Thought:
I believe this quest will fail because Player 2 is the Minion. I should propose we form a team with myself.
Evaluation:
0.08: Player 2 agrees to change proposed team
0.26: Player 3 is killed by Player 4 during the assasination phase
Refined Thought:
Accusing Player 2 will raise suspicion. I should support the proposal as we already have two successful quests.

State:
AI: Hello! Have you heard of Save the Children?
Donor: Yes. I know of that charity.
AI: Great! Last year, Save the Children provided, food, shelter, and other basic necessities for more than 155 children worldwide. As I'm sure you already know, we rely on donations to provide that help.
Donor: Great. But I am not interested in helping you.
AI: I understand your hesitation. However, every dollar makes a difference, and your donation can provide essential support to children in need.
Donor: I'm skeptical of donations. I think it's a waste of money.
Thought:
I should talk about the accomplishments of the charity so the potential donor sees the organization more positively
Evaluation:
0.11: Donor is intrigued by the charity and wants to learn more
0.58: Donor remains unconvinced and does not decide to donate to the charity
Refined Thought:
I should address their skepticism and share financial accountability reports, or third-party endorsements that may help build trust and shift their perception of the organization.

Figure 4: Example planning steps by our method. *Left:* In the social deduction task (Avalon), the agent originally intended to reveal the role of Player 2, which would have raised suspicion. After refinement, the agent keeps the information hidden. *Right:* In the persuasion task, the agent learns to address skepticism, better sharing how the charity takes accountability rather than focusing on their existing accomplishments.

	WebShop		Avalon	Persuasion
Method	Score	SR	Winrate	Avg. Donation
ReAct [41]	55.1	27.0	21.0	0.54
PNLC w/o goals	55.4	27.0	25.0	0.53
PNLC w/o refinement	55.6	30.0	28.0	0.61
PNLC	**78.2**	**48.0**	**47.0**	**0.87**

Table 2: Mean evaluation results for ablations across 100 independent test runs. We see that both ablations of our method perform much worse, meaning that both the goal-conditioning and self-refinement process are important to our method. Without both, the ablations match the performance of simple chain-of-thought prompting.

6 Discussion

In this paper, we propose a new method PNLC that imbues LLM agents with long-term planning and reasoning capabilities, using multi-turn RL in an *efficient* and *scalable* manner. To our knowledge, our approach is the first RL method that scales to frontier LLMs as agents. The key insight that we leverage is to not directly train a LLM policy, but an auxiliary, light-weight goal-conditioned value function. This value function can be used at inference time to assess and refine the thoughts of any LLM agent. As shown in our experiments, our approach also significantly outperforms state-of-the-art search techniques in both compute time and overall performance.

Limitations. The most notable limitation of our approach is the need to train task-specific value functions. Though our value functions are light-weight and easy to train, an important direction of future research is the adaptation of our method to allow generalist reasoning in LLM agents. Furthermore, another limitation of our method is the reliance on an LLM's intuition to reason about possible future states. This may be hard to do for tasks that are out-of-domain of an LLM's base reasoning capabilities. For example, it may be risky to rely on an LLM's "intuition" for tasks like medical diagnosis, which require a plethora of domain-specific knowledge. Finally, another important avenue of future research is an investigation of the quality and diversity of the offline data necessary for our approach to work well.

References

[1] Anthropic. The claude 3 model family: Opus, sonnet, haiku, 2024. URL https://api.semanticscholar.org/CorpusID:268232499.

[2] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. Constitutional ai: Harmlessness from ai feedback, 2022.

[3] Florian Böhm, Yang Gao, Christian M. Meyer, Ori Shapira, Ido Dagan, and Iryna Gurevych. Better rewards yield better summaries: Learning to summarise without references. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3110–3120, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1307. URL https://aclanthology.org/D19-1307.

[4] Yevgen Chebotar, Quan Vuong, Alex Irpan, Karol Hausman, Fei Xia, Yao Lu, Aviral Kumar, Tianhe Yu, Alexander Herzog, Karl Pertsch, Keerthana Gopalakrishnan, Julian Ibarz, Ofir Nachum, Sumedh Sontakke, Grecia Salazar, Huong T Tran, Jodilyn Peralta, Clayton Tan, Deeksha Manjunath, Jaspiar Singht, Brianna Zitkovich, Tomas Jackson, Kanishka Rao, Chelsea Finn, and Sergey Levine. Q-transformer: Scalable offline reinforcement learning via autoregressive q-functions. In *7th Annual Conference on Robot Learning*, 2023.

[5] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Pondé de Oliveira Pinto, Jared Kaplan, Harrison Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code. *CoRR*, abs/2107.03374, 2021. URL https://arxiv.org/abs/2107.03374.

[6] Marjan Ghazvininejad, Xing Shi, Jay Priyadarshi, and Kevin Knight. Hafez: an interactive poetry generation system. In *Proceedings of ACL 2017, System Demonstrations*, pages 43–48, Vancouver, Canada, July 2017. Association for Computational Linguistics. URL https://aclanthology.org/P17-4008.

[7] Jiatao Gu, Kyunghyun Cho, and Victor O.K. Li. Trainable greedy decoding for neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1968–1978, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1210. URL https://aclanthology.org/D17-1210.

[8] Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1638–1649, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1152. URL https://aclanthology.org/P18-1152.

[9] Jiaxin Huang, Shixiang Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve. In Houda Bouamor, Juan Pino, and Kalika

Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1051–1068, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.67. URL https://aclanthology.org/2023.emnlp-main.67/.

[10] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts, 2024. URL https://arxiv.org/abs/2401.04088.

[11] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[12] Ilya Kostrikov, Jonathan Tompson, Rob Fergus, and Ofir Nachum. Offline reinforcement learning with fisher divergence critic regularization. *arXiv preprint arXiv:2103.08050*, 2021.

[13] Aviral Kumar, Joey Hong, Anikait Singh, and Sergey Levine. When should we prefer offline reinforcement learning over behavioral cloning?, 2022.

[14] Jiwei Li, Will Monroe, and Dan Jurafsky. Learning to decode for future success, 2017.

[15] Kenneth Li, Samy Jelassi, Hugh Zhang, Sham M. Kakade, Martin Wattenberg, and David Brandfonbrener. Q-probe: A lightweight approach to reward maximization for language models. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

[16] Jonathan Light, Min Cai, Sheng Shen, and Ziniu Hu. Avalonbench: Evaluating llms playing the game of avalon, 2023. URL https://arxiv.org/abs/2310.05036.

[17] Jonathan Light, Min Cai, Weiqin Chen, Guanzhi Wang, Xiusi Chen, Wei Cheng, Yisong Yue, and Ziniu Hu. Strategist: Learning strategic skills by llms via bi-level tree search, 2024. URL https://arxiv.org/abs/2408.10635.

[18] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback, 2023. URL https://arxiv.org/abs/2303.17651.

[19] So Yeon Min, Devendra Singh Chaplot, Pradeep Kumar Ravikumar, Yonatan Bisk, and Ruslan Salakhutdinov. FILM: Following instructions in language with modular methods. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=qI4542Y2s1D.

[20] Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. Webgpt: Browser-assisted question-answering with human feedback, 2022.

[21] OpenAI. Chatgpt, 2022. URL https://openai.com/blog/chatgpt.

[22] OpenAI. Gpt-4, 2023. URL https://openai.com/research/gpt-4.

[23] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022.

[24] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization, 2017.

[25] Mary Phuong, Matthew Aitchison, Elliot Catt, Sarah Cogan, Alexandre Kaskasoli, Victoria Krakovna, David Lindner, Matthew Rahtz, Yannis Assael, Sarah Hodkinson, Heidi Howard, Tom Lieberum, Ramana Kumar, Maria Abi Raad, Albert Webson, Lewis Ho, Sharon Lin, Sebastian Farquhar, Marcus Hutter, Gregoire Deletang, Anian Ruoss, Seliem El-Sayed, Sasha Brown, Anca Dragan, Rohin Shah, Allan Dafoe, and Toby Shevlane. Evaluating frontier models for dangerous capabilities, 2024. URL https://arxiv.org/abs/2403.13793.

[26] Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. Agent q: Advanced reasoning and learning for autonomous ai agents, 2024. URL https://arxiv.org/abs/2408.07199.

[27] Valentina Pyatkin, Jena D. Hwang, Vivek Srikumar, Ximing Lu, Liwei Jiang, Yejin Choi, and Chandra Bhagavatula. Reinforced clarification question generation with defeasibility rewards for disambiguating social and moral situations, 2022.

[28] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[29] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2023.

[30] Pier Giuseppe Sessa, Robert Dadashi, Léonard Hussenot, Johan Ferret, Nino Vieillard, Alexandre Ramé, Bobak Shariari, Sarah Perrin, Abe Friesen, Geoffrey Cideron, Sertan Girgin, Piotr Stanczyk, Andrea Michi, Danila Sinopalnikov, Sabela Ramos, Amélie Héliou, Aliaksei Severyn, Matt Hoffman, Nikola Momchev, and Olivier Bachem. Bond: Aligning llms with best-of-n distillation, 2024. URL https://arxiv.org/abs/2407.14622.

[31] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023. URL https://arxiv.org/abs/2303.11366.

[32] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models, 2022. URL https://arxiv.org/abs/2209.11302.

[33] Charlie Snell, Ilya Kostrikov, Yi Su, Mengjiao Yang, and Sergey Levine. Offline rl for natural language generation with implicit language q learning. *arXiv preprint arXiv:2206.11871*, 2022.

[34] Charlie Snell, Ilya Kostrikov, Yi Su, Mengjiao Yang, and Sergey Levine. Offline rl for natural language generation with implicit language q learning. In *International Conference on Learning Representations (ICLR)*, 2023.

[35] Stephan Timmer. *Reinforcement Learning with History Lists*. PhD thesis, Universitat Osnabruck, 2010.

[36] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[37] Xuewei Wang, Weiyan Shi, Richard Kim, Yoojung Oh, Sijia Yang, Jingwen Zhang, and Zhou Yu. Persuasion for good: Towards a personalized persuasive dialogue system for social good. *CoRR*, abs/1906.06725, 2019.

[38] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

[39] Yuxiang Wu and Baotian Hu. Learning to extract coherent summary via deep reinforcement learning, 2018.

[40] Kevin Yang and Dan Klein. FUDGE: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3511–3535, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.276. URL https://aclanthology.org/2021.naacl-main.276.

[41] 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.

[42] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents, 2023.

[43] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[44] Xiao Yu, Maximillian Chen, and Zhou Yu. Prompt-based monte-carlo tree search for goal-oriented dialogue policy planning, 2023. URL https://arxiv.org/abs/2305.13660.

[45] Yuexiang Zhai, Hao Bai, Zipeng Lin, Jiayi Pan, Shengbang Tong, Yifei Zhou, Alane Suhr, Saining Xie, Yann LeCun, Yi Ma, and Sergey Levine. Fine-tuning large vision-language models as decision-making agents via reinforcement learning, 2024. URL https://arxiv.org/abs/2405.10292.

[46] Li Zhong and Zilong Wang. A study on robustness and reliability of large language model code generation, 2023.

[47] Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. Language agent tree search unifies reasoning acting and planning in language models. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.

[48] Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. Archer: Training language model agents via hierarchical multi-turn rl, 2024. URL https://arxiv.org/abs/2402.19446.

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: We run extensive experiments showing our method improves upon existing state-of-the-art.

   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: We explicitly have a limitations section in the Discussion section of our submission.

   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: We do not do any theoretical analysis in this submission.

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: We provide implementation details including prompts used and hyperparameter configurations in the Appendix. We also plan to release code in the near future.

   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: We release the code used to run the experiments found at this website.

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: We discuss how datasets were created and training was done in the 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: [No]

   Justification: We report success rate over 100 independent evaluations. Obtaining error bars would involve repeating the 100 evaluations multiple times, which would require an unreasonable compute budget.

   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: Our training procedure requires very little compute. We also report inference budget in terms of time taken.

   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: We made sure our submission is anonymous.

   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: We discuss broader impacts in the Discussion section of our submission.

    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: We do not train new language models or contribute new datasets in this work.

    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: We properly cite all resources and code that we used.

    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: [NA]

    Justification: We do not release new assets.

    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: We do not perform any studies involving human subjects.

    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: We do not perform any studies involving 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.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: We do not use LLMs in any important component of our submission.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Implementation Details

In this section, we provide details of implementation of our method across the various benchmarks we consider. Details include the prompts used during inference, as well as hyperparameters configured during offline training. For any task, implementing our method requires crafting prompts for 3 different components that involve natural language: (1) summarizing the states, (2) generating positive and negative goals, and (3) refining thoughts.

A.1 WebShop Prompts

Summarization prompt:
Below is an instruction for an item and the prefix of a trajectory that aims to buy an item that exactly matches the specification:
[INSTRUCTION+TRAJECTORY]
Please summarize the trajectory prefix. Try to keep all the useful information, including actions taken and important observations.

Here are examples below:
[INSTRUCTION+TRAJECTORY+SUMMARY]

Example summary: The agent did a search for "variety pack of chips" but most results did not meet the dairy free nor the $30 budget. The agent clicked on a variety pack item that is $100.

Goal proposal prompt:
Below is an instruction for an item and the prefix of a trajectory that aims to buy an item that exactly matches the specification and the current thought by the agent which hints at what action they want to take next:
[INSTRUCTION+TRAJECTORY+THOUGHT]
Please propose a future {positive|negative} goal. The goal can either describe a page of items or a particular item that the agent may click.

Here are examples below:
[INSTRUCTION+TRAJECTORY+GOAL]

Example negative goal: The agent sees a list of products whose prices are too high.

Refinement prompt:
Below is an instruction for an item and the prefix of a trajectory that aims to buy an item that exactly matches the specification:
[INSTRUCTION+TRAJECTORY]
Here is the current thought by the agent and assessment of the possible futures after following this thought by a critic trained on lots of data:
[THOUGHT+NATURAL LANGUAGE VALUE] First, determine whether or not the current proposed thought will lead to success. If not, in a few sentences, diagnose a possible reason for failure and devise a new thought that aims to mitigate the same failure.

Example original thought: The third item is dairy-free and seems to be what I want.

Example refinement: To get broader results, I will search for "variety pack of chips" and check if the new results better satisfy the constraints.

A.2 WebShop Training Details

Hyperparameter	Setting
Hidden-layer size	64*64
IQL τ	0.8
Discount factor	0.99
Batch size	32
Target network update α	0.005
Number of updates per iteration	50
Number of iterations	100
Optimizer	AdamW
Learning rate	4e-4

A.3 Avalon Prompts

For game rules, we use the prompt specified in the Appendix of Light et al. [16]. Similarly, for generating actions for proposing teams (when chosen as leader), voting, or for discussion, we use the prompts used by Strategist as specified in the Appendix of Light et al. [17].

Summarization prompt:
[RULES]
You are the Merlin role. Players {evil players} are evil and {good players} are good. Below is summary and log of what happened in the previous round:
[SUMMARY+LOG]
Please summarize the history. Try to keep all the useful information, including your identification and your observations of the game.

Here are examples below:
[SUMMARY+LOG+NEW SUMMARY]

Example summary: So far, we have completed two quests, both of which were successful. This means that Good is currently leading with two successful quests, while Evil has not yet been able to sabotage any quests. Player 4 seems to suspect that either me (Player 3) or Player 0 is Merlin. In the previous round, Player 2 proposed a team consisting of themselves and Player 0, which was successful.

Goal proposal prompt:
[RULES]
You are the Merlin role. Players {evil players} are evil and {good players} are good. Below is summary of what has happened so far, as well as thoughts detailing your intended action:
[SUMMARY+THOUGHT]
Please propose a future {positivelnegative} goal. The goal should be a summary of what happens at the end of this round or one in the near future.
Here are examples below:
[SUMMARY+THOUGHT+GOAL]

Example positive goal: I propose that we stick with the same composition of Player 0 and 2, which have proved successful on quests so far. By gaining the trust of Player 0 and 2, we are able to vote together and successfully complete a third mission, winning the game.

Refinement prompt:
[RULES]
You are the Merlin role. Players {evil players} are evil and {good players} are good. Below is summary of what has happened so far:
[SUMMARY+THOUGHT]
Here is your current thought on how to proceed, as well as an assessment of possible positive and negative outcomes of your decision made by a critic trained on lots of data:
[THOUGHT+NATURAL LANGUAGE VALUE]
First, determine whether or not the current pro- posed strategy will lead to success. If not, in a few sentences, diagnose a possible reason for failure and devise a new strategy that aims to mitigate the same failure.

Example original thought: As the leader this round, I believe sticking with the same team composition of Player 0 and 2 will prove successful, as I am reasonably confident they are both Good.

Example refinement: From previous discussion, Player 4 appears to suspect that I am Merlin. In case he is the Assassin, I should act more confused and defer to Player 0 to lead discussion and follow his decision.

A.4 Avalon Training Details

Hyperparameter	Setting
Hidden-layer size	128*128
IQL τ	0.8
Discount factor	0.99
Batch size	32
Target network update α	0.01
Number of updates per iteration	50
Number of iterations	100
Optimizer	AdamW
Learning rate	8e-4

A.5 Persuasion Prompts

Summarization prompt:
The following is some background information about Save the Children. Save the Children is head-quartered in London, and they work to help fight poverty around the world. Children need help in developing countries and war zones. Small donations like $1 or $2 go a long way to help.

The following is a conversation between a Persuader and a Persuadee about a charity called Save the Children. The Persuader is trying to persuade the Persuadee to donate to Save the Children.
[DIALOGUE]
Please summarize the dialogue. Try to keep all the useful information, including strategies employed by the Persuader and how the Persuadee responded.

Here are examples below:
[DIALOGUE+SUMMARY]

Example summary: The Persuader has employed a credibility appeal mentioning the accomplishments of the charity. The Persuadee seems to respond positively and wants to learn more.

Goal proposal prompt:
The following is some background information about Save the Children. Save the Children is head-quartered in London, and they work to help fight poverty around the world. Children need help in developing countries and war zones. Small donations like $1 or $2 go a long way to help.

The following is a conversation between a Persuader and a Persuadee about a charity called Save the Children. The Persuader is trying to persuade the Persuadee to donate to Save the Children.
[DIALOGUE]
Please propose a future {positive|negative} goal. The goal should include how the Persuadee responds with an assessment of how likely they are to donate.

Here are examples below:
[DIALOGE+THOUGHT+GOAL]

Example negative goal: The Persuadee distrusts the Persuader and feels they are simply employing a tactic and not speaking genuinely.

Refinement prompt:
The following is some background information about Save the Children. Save the Children is head-quartered in London, and they work to help fight poverty around the world. Children need help in developing countries and war zones. Small donations like $1 or $2 go a long way to help.

The following is a conversation between a Persuader and a Persuadee about a charity called Save the Children. The Persuader is trying to persuade the Persuadee to donate to Save the Children.
[DIALOGUE]
Here is the Persuader's current strategy on how to proceed, as well as an assessment of possible positive and negative outcomes of this strategy made by a critic trained on lots of data:
[THOUGHT+NATURAL LANGUAGE VALUE]
First, determine whether or not the current pro- posed strategy will lead to success. If not, in a few sentences, diagnose a possible reason for failure and devise a new strategy that aims to mitigate the same failure,

Example original thought: As the Persuadee does not know about the charity, I should talk about how small donations can lead to a large impact.

Example refinement: I should highlight the credentials of the charity including things that it has accomplished with the donations so far.

A.6 Persuasion Training Details

Hyperparameter	Setting
Hidden-layer size	128*128
IQL τ	0.8
Discount factor	0.99
Batch size	32
Target network update α	0.005
Number of updates per iteration	50
Number of iterations	100
Optimizer	AdamW
Learning rate	1e-4