ArCHer: Training Language Model Agents via Hierarchical Multi-Turn RL

Yifei Zhou
UC Berkeley
yifei_zhou@berkeley.edu

Andrea Zanette
UC Berkeley
zanette@berkeley.edu

Jiayi Pan
UC Berkeley
jiayipan@berkeley.edu

Sergey Levine
UC Berkeley
svlevine@eecs.berkeley.edu

Aviral Kumar
Google Deepmind
aviralkumar@google.com

Abstract

Large language models (LLMs) have the potential to tackle sequential decision-making problems due to their generalist capabilities. Instead of optimizing "myopic" surrogate objectives such as human preferences within a single turn, in such problems, we wish to directly optimize long-term objectives, such as user satisfaction over an entire dialogue with an LLM or delayed success metrics in web navigation. Multi-turn reinforcement learning (RL) provides an appealing approach to directly optimize long-term objectives, but how can we design effective and efficient multi-turn RL algorithms for LLMs? In this work, we propose an algorithmic framework to multi-turn RL for LLMs that preserves the flexibility of token-by-token RL used in single-turn RL problems, while still accommodating long horizons and delayed rewards more effectively. Our framework, the **A**cto**r**-**C**ritic Framework with a **Hie**rarchical Structu**r**e (**ArCHer**), combines a high-level off-policy RL algorithm that trains a value function with a low-level RL algorithm that trains a token-by-token policy. While ArCHer can be instantiated with multiple RL algorithms, a particularly convenient instantiation is to use temporal difference (TD) learning at the high level and on-policy token-level policy gradient at the low level. Empirically, we show that ArCHer significantly improves efficiency and performance of multi-turn LLM tasks, attaining sample efficiency boosts of about **100x** over prior on-policy methods and converging to a much better performance than other off-policy methods.

1 Introduction

Owing to their generalist knowledge, large language models (LLMs) have a tremendous potential to address a wide variety of decision-making problems that can be expressed as text, from writing code (Yang et al., 2023a; Li et al., 2022; Lin et al., 2018), navigating the web (Zhou et al., 2023a; Yao et al., 2023a), and using tools (Schick et al., 2023), all the way to interacting with humans (Ghosal et al., 2022; Verma et al., 2022; Jaques et al., 2020). In order to succeed in these problems, an LLM needs to make a *sequence* of intelligent decisions over multiple turns of interaction instead of generating the most *probable* text at each step.

Despite these "multi-turn" use cases, most methods for eliciting goal-directed behavior from LLMs often rely on "myopic" objectives that either attempt to mimic successful demonstrations Zeng et al. (2023); Chen et al. (2023), or otherwise optimize for single-turn human preferences (Touvron et al., 2023; Ouyang et al., 2022; Bai et al., 2022). Policies trained via single-turn approaches often fail to do appropriate credit assignment (e.g. fail to identify good actions from suboptimal trajectories) and degrade in performance as the length of interaction increases. To address this problem, in this paper, we apply "multi-turn" RL to directly maximize the long-term objective of interest (e.g., customer satisfaction at the end of a multi-turn conversation with an LLM assistant), which can be formulated as scalar rewards.

Unlike single-step RL (Touvron et al., 2023; Ouyang et al., 2022; Bai et al., 2022), training LLMs via multi-turn RL presents a number of unique challenges. First of all, multi-turn RL would require online interaction with external sources such as humans or web servers, which can be slow and expensive. Due to this, on-policy methods such as PPO Schulman et al. (2017) quickly become impractical due to their inability to reuse data from past interaction. While off-policy, or even fully offline methods circumvent this problem, these methods present other challenges: since the number of tokens accumulate with multiple turns, token-level methods Snell et al. (2023); Jaques et al. (2020) that consider individual tokens as actions suffer from extremely slow convergence over long horizons. Finally, to address long horizon issues, one can treat the entire utterance for each turn as an action Verma et al. (2022); Jang et al. (2022), but this comes at the cost of introducing an enormous, variable-length action space, presenting a challenge for off-policy methods based on Temporal Difference (TD) learning that require maximization over the action at each time step. In all, this means that there is no multi-turn RL appraoch for LLMs that is both performant and efficient.

In this paper, we devise a novel framework for developing multi-turn RL algorithms for LLMs. Our key insight is that a **hierarchical approach** for RL with language models can enjoy the best of both worlds, where an off-policy temporal difference learning method can train an utterance-level value function at the high level, and any policy gradient algorithm can optimize the token generation at each turn of the interaction at the low level, treating the high-level value function as the terminal reward for that turn. This allows for sample reuse and faster convergence, while avoiding the need to perform Bellman backups over each individual token, as the high-level critic is trained at a coarser time-scale given by utterances. It also directly inherits tuning implementations from any existing token-level policy gradient algorithm that has been developed for single-turn RL with preferences, but with the utterance-level value function inheriting the role of the reward model. This way we are able to obtain the best of both utterance-based and token-based, and off-policy and on-policy approaches for training LLMs.

The primary contribution of this work is a framework for developing hierarchical RL approaches for LLMs, that we call: **A**cto**r**-**C**ritic framework with a **Hie**rarchical Structu**r**e (or **ArCHer** in short). We study several concrete algorithmic instantations derived from the ArCHer framework by conducting experiments on a range of language decision-making tasks (also conventionally referred to as "language agent" tasks) when learning from its autonomously collected data (i.e., the "online" setting). We find that algorithms derived from ArCHer are 100x more sample efficient than state-of-the-art on-policy method PPO and converge to a better performance than state-of-the-art off-policy methods. Moreover, our methods are simple to implement, directly enabling plug-and-play choices of algorithms in literature.

2 Related Work

Single-turn reinforcement learning for LLMs. Most prior works that use RL for LLMs have focused on decision-making problems where the language model must produce a single decision, with no further steps of interaction with an external environment (e.g., the "single-turn" preference optimization setting (Casper et al., 2023; Christiano et al., 2023; Ziegler et al., 2019)). Typical algorithms used for this sort of single-turn RL include policy-gradient methods such as PPO Ouyang et al. (2022); OpenAI et al. (2023); Ramamurthy et al. (2023), A2C Glaese et al. (2022), offline optimization methods such as DPO Rafailov et al. (2023), and filtered supervised learning approaches (Yuan et al., 2023; Korbak et al., 2023; Gulcehre et al., 2023). Despite promising results, there remain many important agent problems that cannot be solved in a single-turn setting. Many of these problems require the agent to explicitly take the steps to gather information before making a decision, such as asking personalized preferences before making a travel plan (Hong et al., 2023) or initially attempting to read the help manual of the shell in a linux terminal, before carrying out the requested tasks (Liu et al., 2023). Single-step approaches cannot learn such nuanced strategies as they attempt to solve the problem within a single step, necessitating multi-turn RL methods for training LLMs.

Training language agents without RL. Motivated by few-shot learning and reasoning abilities of LLMs, prior works also utilize LLMs for sequential decision-making via prompt engineering. For example, ReAct (Yao et al., 2023b) and Reflexion Shinn et al. (2023) prompt the LLM to "think" and analyze past failures before executing the next action. Voyager Wang et al. (2023a) prompts the LLM agent to develop and refine a curriculum and action library based on environment input. However,

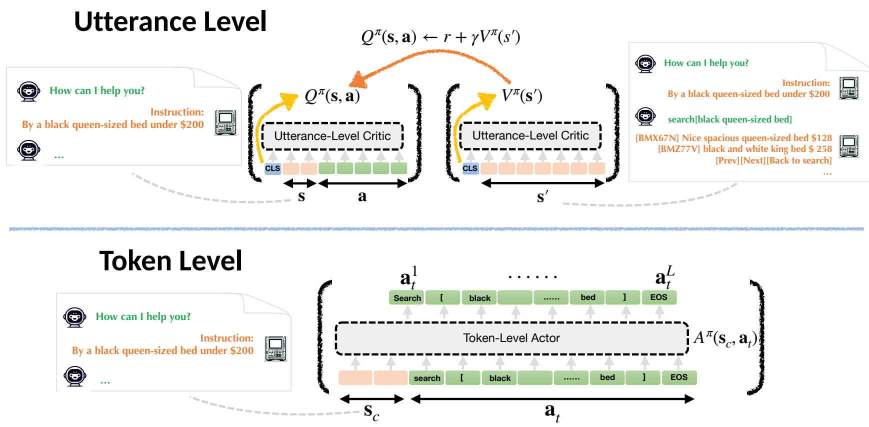


Figure 1: A diagram of our practical algorithm of Actor-Critic Framework with a Hierarchical Structure (ArCHer). Our algorithm operates both at the utterance level and the token level. At the utterance level, our algorithm learns both a Q function and a V function bootstrapping each other with TD errors. At the token level, the policy is learnt by maximizing the advantages calculated by the utterance-level critic at the end of each utterance.

without updating the parameters of the LLM, the effectiveness of these methods is inherently limited by the intrinsic capabilities obtained from pre-training Zeng et al. (2023); Chen et al. (2023). Even state-of-the-art models such as GPT-4 with in-context learning can perform very sub-optimally on out-of-distribution settings Liu et al. (2023); Yang et al. (2023b) without updating the model. To improve over pre-trained capabilities, another line of work finetunes LLMs with filtered successful trajectories (generated manually or by rolling out a strong pre-trained LLM) (Schick et al., 2023; Zeng et al., 2023; Chen et al., 2023). However, manual labels and tool call annotations are be expensive to obtain. Moreover, it would be prohibitively expensive for automated approaches to stumble upon successful rollouts will as the task horizon increases (Liu et al., 2023; Abdulhai et al., 2023). Therefore, in this paper, we sidestep these problems by attempting to directly maximize the objective of interest via RL Kumar et al. (2020); Levine et al. (2020).

Multi-turn RL for LLMs. While many prior works directly use off-the-shelf policy-gradient methods, such as PPO Schulman et al. (2017); Szot et al. (2023); Yao et al. (2023a) and REINFORCE Sutton et al. (1999); Williams (2004); Ranzato et al. (2015); Wu & Hu (2018); Paulus et al. (2017) to train LMs, these methods can become sample inefficient in multi-step problems that require interaction with an external environment Verma et al. (2022); Jang et al. (2022). To address such sample complexity issues, off-policy and offline value-based methods learn from existing static data (Snell et al., 2023; Jaques et al., 2020; Verma et al., 2022; Jang et al., 2022). However, existing off-policy methods for multi-turn language tasks either **(1)** consider a single token as an action (i.e., “token-level”) (Snell et al., 2023; Jaques et al., 2020) and must deal with long horizons, or **(2)** consider an utterance as a single action, but utilize multiple candidate utterances from pre-trained LLMs for maximization in the Bellman backup Verma et al. (2022); Jang et al. (2022), reducing the pace of policy improvement. As we discuss next, our approach will address both of these limitations of existing off-policy methods, enabling fast policy improvement even in long horizon problems.

3 Actor-Critic Framework with a Hierarchical Structure (ArCHer)

Token-level RL methods often suffer from the curse of long horizons. Utterance-level RL methods avoid this challenge, but now face the challenge of tractably optimizing for a coherent set of tokens within RL updates. In this section, we will develop a new class of hierarchical RL methods, ArCHer, that bypass these challenges by running two RL algorithms, one at the utterance-level and another at the token-level. We begin by describing how language generation can be posed as a hierarchical Markovain Decision Process (MDP), followed by multi-turn RL methods for this hierarchical MDP.

3.1 Language Generation as a Hierarchical MDP

In order to derive effective multi-turn RL algorithms, we will now present a novel formulation of language generation as acting in a hierarchical MDP. Our construction defines a *high-level* and a *low-level* MDP embedded inside the high-level MDP. For both MDPs, states are a variable-length sequence of tokens. For the high-level MDP, an action is defined as a sequence of tokens. An action in the low-level MDP is a single token, such that executing a sequence of actions in the low-level MDP corresponds to a single action in the high-level MDP. We approach multi-turn tasks via a hierarchical MDP framework, which we will argue strikes the right balance by enabling efficient off-policy learning while allowing for the direct use of token-level on-policy RL methods (similar to widely used preference optimization methods) at the low level, which does not require any online interaction with the external environment.

Formally, each state s_t for the high-level MDP consists of interaction history between the LLM and the external environment, and each a_t is a sequence of tokens. The low-level MDP models the generation of an individual utterance by the agent in a single turn, where each action a_t^h is an individual token (the h-th token in the t-th utterance). Each state is composed of a context state s_c consisting of the interaction history from the previous high-level states and $a_t^{1:h-1}$, the tokens in the current utterance preceding the current token. The next state is obtained by concatenating the current action to the token history. Figure 1 shows a visualization of our notations for the hierarchical MDP.

A policy π in the high-level MDP directly aims to maximize cumulative task reward, whereas a policy in the low-level MDP attempts to find a sequence of L tokens $a_t^{1:L}$ that maximizes reward equal to the value function of the high-level MDP $Q^\pi(s, a_t^{1:L})$, provided at the end of the utterance.

The multi-turn setting provides a natural decomposition that fits into this hierarchical framework, where each turn of the multi-turn interaction (each "utterance") is a time step at the high-level MDP, and each token within a turn is a time step at the low-level MDP. As a concrete example, for a language agent tasked with buying requested items from the web, in the utterance-level MDP, an action would be an entire utterance, for example, "search[queen-sized bed, black]". The state space consists of the history of webpages browsed so far and previous actions. A candidate reward function would be +1 if the correct item can be bought and 0 otherwise. The dynamics function would involve the web engine that reacts to the actions taken. Each token of an utterance (e.g., "search[queen-sized bed, black]") would be an individual action in the embedded token-level MDP, for example, tokens "search", "[", "queen", and so on.

3.2 Reinforcement Learning Definitions

In order to describe the details of our framework ArCHer, we will define a few important definitions from the RL literature. The Q function of a high-level policy π defined to be the expected long-term return accumulated by executing a certain action at the current step, followed by executing π thereafter: $Q^\pi(s_h, a_h) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^i r(s_{h+t}, a_{h+t})\right]$. The value function $V^\pi(s_h)$ is the expected Q-value, $Q^\pi(s_h, a_h)$ when action a_h is sampled from the policy π. The advantage function $A^\pi(s_h, a_h)$ for a given state-action pair is the difference of the Q-value for this particular state-action pair and the value function of the state under the policy: $A^\pi(s_h, a_h) = Q^\pi(s_h, a_h) - V^\pi(s_h)$. The value function in the low-level MDP is denoted by $\widetilde{V}(s_c, a_h^{1:i-1})$.

3.3 RL Algorithms in the Language Hierarchical MDP

The proposed hierarchical MDP provides flexibility in designing multi-turn RL algorithms: we could use any choice of RL algorithm for either the high or the low level. That said, note that only the high level requires interaction with a (non-differentiable) environment, while the low level optimizes against the high-level value function, and therefore trains entirely in silico, without any interaction with an environment. Therefore, the requisite strengths of these methods must be different: the high-level algorithm should be highly sample efficient, while the low-level algorithm should be easy to optimize. A particularly convenient choice is to use TD learning at the utterance-level as the high-level RL algorithm, while using on-policy methods borrowed from existing token-level preference optimization literature Ouyang et al. (2022); Bai et al. (2022) as the low-level RL algorithm.

3.4 Practical Instantiation of ArCHer for Online RL

As shown in Figure 1, our practical algorithm for ArCHer instantiates an utterance-level critic trained with Bellman bootstrapping and a token-level policy trained with policy gradients. Following the practice in prior RL algorithms Snell et al. (2023), we train two models, one to represent the Q-function $Q_\theta^\pi(s, a)$, and a separate model for the value function $V_\psi^\pi(s)$, where θ and ψ are parameters for the Q-function and the value function. The value function is updated towards a stale copy of the Q-function, referred to as the target Q-function $Q_{\bar{\theta}}$, while the Q-function is updated towards the target value function $V_{\bar{\psi}}$ Mnih et al. (2013). The token-level policy is parameterized by ϕ.

Algorithm 1 ArCHer: Practical Framework

```
Initialize parameters φ, ψ, θ, θ̄, (Optionally) η
Initialize replay buffer D (optionally from an offline dataset).
for each iteration do
    (Only online) Collect trials, add to replay buffer D.
    Update utterance-level Q_θ and V_ψ by bootstrapping.
    Update target networks Q_θ̄ and V_ψ̄
    (Optionally) Update token-level baseline Ṽ_η.
    Update token-level actor π_φ.
end for
```

To reuse past interaction data, we maintain a replay buffer $\mathcal{D} = \{s_i, a_i, r_i, s_i'\}_{i=1}^N$ that stores all past utterance-level interaction data pairs $\{s_i, a_i, r_i, s_i'\}$ collected from the rollouts of previous policies. The objective for training the Q function is described in Equation 1:

$$J_Q(\theta) = \mathbb{E}_{s,a,r,s' \sim \mathcal{D}}[(Q_\theta(s,a) - r - \gamma V_{\bar{\psi}}(s'))^2]. \tag{1}$$

The training objective for V_ψ is to minimize the squared consistency loss with the target Q-network:

$$J_V(\psi) = \mathbb{E}_{s \sim \mathcal{D}}[\mathbb{E}_{a \sim \pi_\phi(\cdot|s)}[(V_\psi(s) - Q_{\bar{\theta}}(s,a))^2]]. \tag{2}$$

In practice, we can obtain an estimator of this loss by sampling a batch of n observations $\{s_i\}_{i=1}^n$ and multiple sequences of actions $\{a_i\}_{i=1}^n$ sampled from the actor to obtain an utterance a_i for a given observation s_i. The target networks $Q_{\bar{\theta}}$ and $V_{\bar{\psi}}$ are updated towards the current parameters Q_θ with a polyak average.

At the lower-level, we wish to train the token-level actor π_ϕ to find a consistent sequence of tokens that maximize the Q-value of the utterance as quantified by the utterance-level Q-function obtained above. We utilize an on-policy policy gradient approach to accomplish this, and train π_ϕ to maximize the sum of rewards, where the utterance-level advantage function $A^\pi(s,a)$ corresponds to the terminal reward and the reward is 0 elsewhere. This subroutine for training the actor generalizes single-turn RL methods such as PPO, except that the reward model is now an estimate of the multi-turn advantage instead of a reward model.

Concretely, for the most convenient instantiation, we update the token-level policy with the policy gradient computed via REINFORCE Williams (2004):

$$J_\phi(\pi) = \mathbb{E}_{s_c \sim \mathcal{D}, a_t^{1:L} \sim \pi(\cdot|s_c)} \left[\sum_{i=1}^{L} A(s_c, a_t^{1:L}) \log \pi_\phi(a_t^i | s_c, a_t^{1:i-1}) \right]. \tag{3}$$

3.5 Other Practical Algorithms from ArCHer

The instantiation of ArCHer described in Section 3.4 is simple and ready-to-use, but the flexibility of the ArCHer framework also enables it to incorporate other components that prior literature found to be successful. We describe two variants in the two paragraphs to follow. The first improves the sample efficiency while interacting with the external environment and the second enables ArCHer to learn entirely from a dataset of pre-collected experience.

Improvements to direct policy gradients. In applications where the horizon of the token-level MDP is long (i.e. each utterance has many tokens), vanilla REINFORCE may struggle due to a high variance in trajectory reward Schulman et al. (2015) and the variance can be reduced by introducing a token-level baseline $\widetilde{V}_\eta(\widetilde{\pi})$ parameterized by η in the token-level MDP. For simplicity, the token-level baseline is trained with Monte Carlo estimation of rewards to go without any discount in Equation 4:

$$J_\eta(\widetilde{V}) = \mathbb{E}_{s_c \sim \mathcal{D}, a_t^{1:L} \sim \pi(\cdot|s_c)}[\sum_{i=1}^{L} (A(s_c, a_t^{1:L}) - \widetilde{V}_\eta(s_c, a_t^{1:i-1}))^2]. \tag{4}$$

With a token-level baseline, the new objective for the actor is Equation 5:

$$\begin{aligned} J_\phi(\pi) = \mathbb{E}_{s_c \sim \mathcal{D}, a_t^{1:L} \sim \pi(\cdot|s_c)}[&\sum_{i=1}^{L} (A(s_c, a_t^{1:L}) \\ &- \widetilde{V}_\eta(s_c, a_t^{1:i-1})) \log \pi_\phi(a_t^i | s_c, a_t^{1:i-1})]. \end{aligned} \tag{5}$$

Offline learning with ArCHer. ArCHer can also learn from a dataset of pre-collected experience without further online interactions. A distinguishing aspect of the offline setting is that improving the policy normally results in selecting out of distribution actions, whose Q values are difficult to estimate accurately Levine et al. (2020); Kumar et al. (2020). Directly optimizing the Q values, such as in the online setting, can lead to poor policies. This suggests that adopting different algorithmic components for the actor and for the critic may be needed. The standard approach to the problem of offline learning is based on regularizing the policy towards the behavioural cloning policy. To achieve this goal, we incorporate two standard design choices, namely the IQL Kostrikov et al. (2021) critic and the AWR Peng et al. (2019) actor, and present the empirical results in Section 4.5. At a high level, the IQL critic approximates the Q value of a near optimal policy within the support of offline data, while the AWR actor optimizes with respect to these Q values while ensuring that the policy remains close to the behavioural policy.

More precisely, IQL uses *implicit* maximization over actions (i.e., without a $\max$ operator) in the offline dataset via expectile regression as follows:

$$J_{\psi}^{\text{IQL}}(V) = \mathbb{E}_{s\sim\mathcal{D}}[\mathbb{E}_{a\sim\pi_{\phi}(\cdot|s)}[L_2^{\tau}(V_{\psi}(s) - Q_{\bar{\theta}}(s,a))]], \tag{6}$$

where $L_2^{\tau}(u) = |\tau - \mathbf{1}\{u<0\}|u^2$.

The policy extracted by AWR Peng et al. (2019) trades-off between searching for high-return policies and imitating the policy that generated the dataset, by minimizing the loss

$$J_{\phi}(\pi) = -\mathbb{E}_{(s_c, a_t^{1:L})\sim\mathcal{D}}[e^{\beta A(s_c, a_t^{1:L})} \sum_{i=1}^{L} \log \pi_{\phi}(a_t^i | s_c, a_t^{1:i-1})]. \tag{7}$$

Such trade-off is controlled by a positive, user-defined scalar value β. Low values for β encourage imitating the policy that generated the dataset. On the contrary, large values of β can lead to a potentially better policy but only if the advantage is reliably estimated.

3.6 Practical Implementation Details

In our main experiments, we use a GPT-2 Radford et al. (2019) architecture for representing the policy, and a RoBERTa-base model Liu et al. (2019) with a MLP layer on top of the "[CLS]" token hidden states for the utterance-level critic. The token-level baseline is also represented by a GPT2 architecture with a MLP layer on top of the hidden states for each token. To address the issue of overestimation of Q values prevalent in deep reinforcement learning, we also employ the double Q-learning method van Hasselt et al. (2015) to maintain two critics of Q_1, V_1 and Q_2, V_2. In this way, Q_1 and V_1 bootstraps each other independently from the bootstrapping of Q_2 and V_2. And the advantage is calculated by taking a minimum over Q_1, Q_2 and V_1, V_2. To save computation and memory, we have Q_1, Q_2, V_1, V_2 share the same language model encoder backbone with separate heads of two layer MLPs. Similarly, the target functions for Q_1, Q_2, V_1, V_2 also share a separate language model backbone with their own heads. The parameters of the token-level actor are independent from the critic. The token-level baseline utilizes a separate autoregressive backbone when incorporated in the algorithm. Our model architecture is presented in Figure 1, while additional details and hyperparameters are in Appendix I.

4 Experiments

Our experiments are designed to answer the following questions: **(1)** Is ArCHer able to achieve better sample complexity and performance than prior on-policy and off-policy RL methods for LLMs when learning from interaction in the online RL setting? **(2)** Does the TD-learning design for the utterance-level critic enable an effective use of off-policy data? **(3)** How do different practical algorithms derived from our ArCHer framework compare?

4.1 Environments

While some prior works Liu et al. (2023); Wang et al. (2023b); Cheng et al. (2023); Abdulhai et al. (2023) have proposed benchmarks for LLM agent problems, such as coding and question answering, many of these tasks are not well suited for our evaluation. For example, some domains such as MultiWoz Budzianowski et al. (2018) involve dialogue, but do not involve multi-turn interaction and delayed rewards. Others are designed primarily for evaluation of pre-trained LLMs, with ground truth responses rather than reward signals Yang et al. (2023a). Yet others focus heavily on transfer of prior knowledge Liu et al. (2023); Wang et al. (2023b), such that larger and more powerful models are the main driver of improvement rather than more advanced RL algorithms. Since we aim specifically to enable deliberate maximization of delayed rewards over multiple turns of interaction, and aim to evaluate specifically the performance our algorithm rather than any single LLM model, we select our evaluation tasks from prior work on the following criteria: **(1)** Does the task necessitate multi-turn planning and reasoning with LLMs? **(2)** Is the task self-contained, meaning that the LLM agents can learn to achieve near expert performance through interaction with the environment, vs. by transferring extensive prior knowledge? **(3)** Is the environment realistic, in the sense that successful agents need to understand and generate reasonable natural language? **(4)** Does the environment allow fast iteration and easy reproduction (i.e., doesn't require costly user studies with live humans)? Based on these criteria, we selected a diverse mix of domains from prior work that we describe below.

Detective Game Hausknecht et al. (2019) is an interactive text fiction game where the agent must generate a sequence of natural language actions (e.g.,"take paper", "eat apple") based on environment feedback. A reward is given if the agent reaches some milestones towards finishing the game, where the end goal is to successfully find the murderer in a murder mystery. **Twenty Questions** Abdulhai et al. (2023) is a dialogue task where the agent plays the role of a guesser trying to guess a hidden word from a list of 157 words within twenty yes/no questions. The oracle answers the questions with "Yes.", "No.", or "Invalid Question." The oracle is simulated with a "flan-t5-small" Chung et al. (2022) model trained with supervised fine-tuning on the dataset provided by Abdulhai et al. (2023). Upon guessing the correct word, the agent receives a reward of 0 and the environment terminates. Otherwise, a reward of -1 is provided at each time step. We also study a variation of this task with a list of only 10 possible underlying words, that we call **Twenty Questions Subset**. This variant challenges the algorithms to tailor a very specific strategy when there is a shortcut in the task. **Guess My City** Abdulhai et al. (2023) is a similar multi-turn task where the agent attempts to guess the name of a hidden city from a list of 100 cities within twenty questions. A crucial difference between the Guess My City task and the Twenty Questions task is that the guesser is now allowed to ask any question and can observe free-form responses (which are not necessarily "Yes" or "No"). **WebShop** Yao et al. (2023a) is a tool-use environment where the agent is instructed to buy an item from a web shopping server. A dense reward between 0 and 1 is provided based on the similarity of the item purchased by the agent and the item requested. See Appendix D for more details.

All of those chosen tasks require strategic multi-step planning under delayed rewards, and cannot be solved in one turn. They also require the agents to understand and generate good natural language while keeping the rule of the task as straightforward as possible. Evaluations of these tasks are also fast and objective. We chose these tasks over other possible choices, which were less suitable for evaluating our research questions. Most existing terminal and database tasks Yang et al. (2023a); Liu et al. (2023) rely on knowledge of Linux and database grammars, making performance depend heavily on model size rather than algorithm, and typically can be solved in a single step. Some other dialogue and tutoring tasks Hong et al. (2023); Verma et al. (2022); Zhu et al. (2020); Lee et al. (2019); Budzianowski et al. (2018) require either costly user studies or other metrics that do not directly represent task performance, making them unfavorable for fast iteration or reliable reproduction.

4.2 Comparisons in the Online Setting

We compare our method to a number of alternative RL approaches. For token-level methods, we consider token-level **PPO** Schulman et al. (2017) due to its state-of-the-art performances in RL for LLMs Ouyang et al. (2022); Ramamurthy et al. (2022). For each iteration, **PPO** collects new on-policy data by rolling out the current actor and uses these data to estimate policy gradients. The actor is trained with a clipped policy gradient loss. Importantly, data collected by previous actors is simply discarded and is not used for later iterations. We use an existing PPO implementation by Abdulhai et al. (2023). We tried implementing a **token-level DQN** Mnih et al. (2013) method. We omitted its performance in the paper because it fails to learn in any task and we did not find results reported of this baseline in language tasks in the prior literature. We also consider a simple non-RL baseline **Filtered BC** which has competitive performances with state-of-the-art RL methods for LLMs Abdulhai et al. (2023); Snell et al. (2023). Filtered BC maintains a fixed size replay buffer of recent trajectories and the actor is trained with a Behavior Cloning (BC) loss on the top 10% trajectories with the highest episode total rewards. For utterance-level methods, we compare with a state-of-the-art utterance-level RL method **CHAI** Verma et al. (2022). The original version of CHAI is designed for offline RL. To extend it to the online setting, we simply remove the CQL loss and update CHAI with data from the off-policy replay buffer. The utterance-level critic of CHAI is trained with the same Bellman bootstrapping objective as ArCHer. However, a key difference from ArCHer is that CHAI maintains a frozen actor from a Supervised Fine-Tuning (SFT) checkpoint. Each time when an action needs to be sampled, k utterances are sampled from the frozen actor. The utterance critic ranks those k utterances and chooses the utterance with the highest Q value. We use CHAI with $k = 5$ and for each sampling 5 actions are sampled independently from the frozen actor and all of them need to be evaluated separately by another critic, while ArCHer only samples one action from the actor. We suspect that increasing the the number of k to even larger can result in an improved performance, but it is impractical due to its significant computation overhead (the training speed for $k = 5$ is already around four times slower than ArCHer). Another offline method, **GPTCritic** Jang et al. (2022), will follow a similar design if we extend it to the online setting. Model architectures for all baseline methods are the same as our instantiation of ArCHer described in Section 3.6.

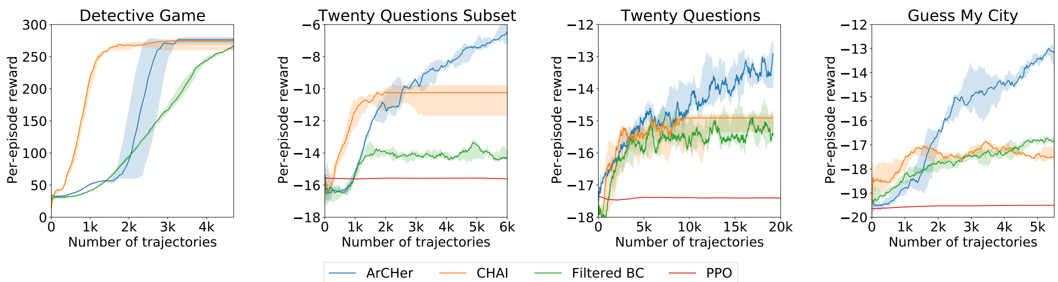


Figure 2: The online learning performance of our method compared with prior state-of-the-art methods across five different tasks. The median of each method is plotted and the min/max values are shaded with 3 random seeds. The comparison on WebShop can be found in Figure 3. Due to computational constraints, we have to stop some some baseline learning curves early when they have converged. We found that PPO still gradually improves despite being slow. A zoom-in plot of the learning curve for PPO can be found in Figure 6.

4.3 Sample Efficiency in the Online Setting

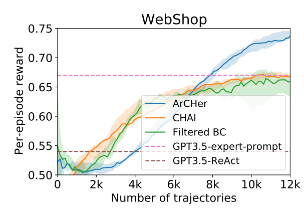


Figure 3: Continued from Figure 2. The online learning performance of our method compared with prior state-of-the-art methods on WebShop.

Figure 2 and 3 show the comparison between ArCHer with other baseline methods across five natural language tasks, and example rollouts of ArCHer for each environment can be found in Appendix H. In order to efficiently perform exploration in the environment, we initialize the policy language models from a policy checkpoint obtained by running supervised fine-tuning on sub-optimal data (see Appendix D for how data for each environment is generated). Only the amount of online interaction data is compared in Figure 2 and 3, and all methods use the supervised fine-tuning to initialize this process. Overall, we found that ArCHer converges to a much better performance than all other baselines in the four harder tasks with more stochasticity (Twenty Questions Subset, Twenty Questions, Guess My City, and WebShop). Comparing with a larger LLM with state-of-the-art prompts (Expert Prompt and ReAct), we observed that with proper prompting, GPT-3.5 can achieve competitive performance on WebShop without fine-tuning to this specific task. However, by learning from online interaction, ArCHer is able to gradually improve itself and achieve even higher performance.

First of all, we found token-level PPO fails to achieve competitive performances with other off-policy methods using the same amount of data. This is perhaps unsurprising, as PPO is an on-policy method and therefore cannot effectively reuse samples from past iterations, leading to significantly worse sample efficiency. In particular, for Twenty Questions , we observed that PPO could only stably improve when provided with at least 1024 on-policy rollouts for each gradient step, likely because of high gradient variance. This observation corroborates the finding of Abdulhai et al. (2023), suggesting that online PPO is less practical in this setting. Quantitatively, as shown in a zoom-in figure of PPO in the supplementary Figure 6, while it takes more than 100k samples for PPO to attain an average return just higher than -17, ArCHer can attain this value with fewer than 1000 samples, indicating at least a **100x boost** in sample efficiency.

While filtered BC generally converges very quickly, the resulting policy often performs suboptimally and does not improve with more data. On the other hand, ArCHer exhibits steady improvements as it collects more samples. Finally, we observed that CHAI often learns faster than ArCHer in most environments initially, but it also converges to a worse solution. We suspect this is because the critic in CHAI is directly used to rerank samples from the frozen behavior policy, which only enables a narrow margin for policy improvement. On the contrary, ArCHer needs an initial learning phase to pick up the actor via gradient-based training, after which it continues to improve. Additionally, we note that sampling an action with CHAI is 10x slower than ArCHer in terms of wall-clock time, as it requires sampling multiple responses and evaluating their Q-values.

4.4 IMPORTANCE OF OFF-POLICY DATA

In (b) of Figure 4, we investigate the importance of off-policy data by varying the size of replay buffer on Guess My City. A smaller replay buffer means that updates rely more on repeatedly sampling on-policy data. In our experiments, we observed that using a replay buffer containing only the most recent 48 trajectories results in unstable learning, likely due to overfitting on limited data. On the other hand, larger buffers are most stable. However, increasing the size of the buffer beyond a certain point is benign and does not result in meaningful changes to performance. This experiment results show that indeed making use of off-policy data can improve the stability and performance of our practical algorithm in ArCHer framework.

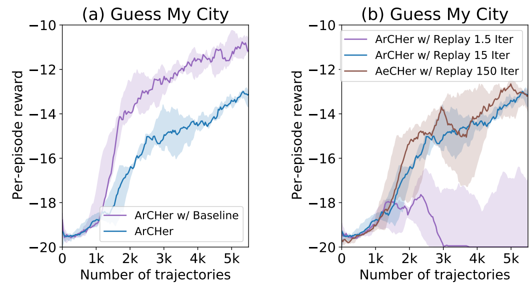


Figure 4: (a) Ablation study of the token-level baseline on Guess My City. (b) Ablation study of the importance of off-policy data by varying the size of the replay buffer on Guess My City.

4.5 ALTERNATE PRACTICAL DESIGNS OF ARCHER

Enhancing online ArCHer with a token-level baseline. In (a) of Figure 4, we compared the performance of ArCHer with and without a token-level baseline on Guess My City, where the utterances of the agent are longer and more diverse than other tasks. We observed that ArCHer w/ Baseline outperforms ArCHer by a large margin, indicating that the introduction of the token-level baseline can effectively reduce the variance of vanilla policy gradients while updating the token-level actor, when the utterances are long and diverse. However, this improvement comes at the cost of another language model backbone and more computations, which might not be necessary when each utterance is short. This illustrates one of the benefits of our hierarchical design: we can choose the best method for the higher and lower level based on their distinct requirements.

Offline ArCHer with IQL and AWR. In this section we present a preliminary study of ArCHer in the offline setting. Due to computational constraints, we did not perform extensive comparisons with the state-of-the-art algorithms; rather, we investigated the effect of several design choices in order to derive a preliminary version of ArCHer for the offline setting, including IQL and AWR losses described in Section 3.5. We report the performances in Table 1 on Twenty Questions, where we report the median performance across 5 random seeds for ArCHer and for our own implementation the baselines BC and Filtered BC similar to described in Section 4.2 except the replay buffer is kept fixed with the pre-collected offline dataset. Overall, ArCHer compares favorably over our implemented BC and Filtered BC, and its performance is on par with that of the ILQL and the MC return baselines reported by Abdulhai et al. (2023), despite the disadvantage of using a smaller model (GPT2-small as opposed to GPT2-medium).

	Twenty Questions
ArCHer (IQL + AWR)	$\mathbf{-14.1}$
ArCHer (IQL + REINFORCE)	-20
ArCHer (IQL + REINFORCE + BC)	-15.3
ArCHer (SARSA + AWR)	-14.5
Filtered BC	-15.4
BC	-16.8
ILQL*	-14.2
MC Return*	$\mathbf{-13.9}$

Table 1: Summary of offline performances on Twenty Questions. The ILQL and MC return scores are taken from Abdulhai et al. (2023), and were computed by the authors with a larger (GPT2-medium) LLM.

In the same table, we test the ablation of different components in ArCHer. With REINFORCE loss alone, the performance quickly collapses due to the lack of regularization. We also tested a combination of REINFORCE and the BC loss, which we found however not to match the performance of AWR and is more prone to collapse. Finally, we ablated the IQL loss (and performed the regular TD backup in Section 3.4) while keeping AWR in place, but such variant did not offer the same level of performance as the IQL loss in the offline setting, highlighting the need for an implicit maximization operator within support of offline dataset.

REFERENCES

Marwa Abdulhai, Isadora White, Charlie Snell, Charles Sun, Joey Hong, Yuexiang Zhai, Kelvin Xu, and Sergey Levine. Lmrl gym: Benchmarks for multi-turn reinforcement learning with language models, 2023.

Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun. *Reinforcement learning: Theory and algorithms*. 2019. URL https://rltheorybook.github.io/.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022.

Pawel Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. Multiwoz - A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *CoRR*, abs/1810.00278, 2018. URL http://arxiv.org/abs/1810.00278.

Stephen Casper, Xander Davies, Claudia Shi, Thomas Krendl Gilbert, Jérémy Scheurer, Javier Rando, Rachel Freedman, Tomasz Korbak, David Lindner, Pedro Freire, Tony Wang, Samuel Marks, Charbel-Raphaël Segerie, Micah Carroll, Andi Peng, Phillip Christoffersen, Mehul Damani, Stewart Slocum, Usman Anwar, Anand Siththaranjan, Max Nadeau, Eric J. Michaud, Jacob Pfau, Dmitrii Krasheninnikov, Xin Chen, Lauro Langosco, Peter Hase, Erdem Bıyık, Anca Dragan, David Krueger, Dorsa Sadigh, and Dylan Hadfield-Menell. Open problems and fundamental limitations of reinforcement learning from human feedback, 2023.

Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. Fireact: Toward language agent fine-tuning. *ArXiv*, abs/2310.05915, 2023. URL https://api.semanticscholar.org/CorpusID:263829338.

Ching-An Cheng, Andrey Kolobov, Dipendra Misra, Allen Nie, and Adith Swaminathan. Llf-bench: Benchmark for interactive learning from language feedback. *ArXiv*, abs/2312.06853, 2023. URL https://api.semanticscholar.org/CorpusID:266174230.

Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences, 2023.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022.

Dylan J. Foster, Akshay Krishnamurthy, David Simchi-Levi, and Yunzong Xu. Offline reinforcement learning: Fundamental barriers for value function approximation. *CoRR*, abs/2111.10919, 2021. URL https://arxiv.org/abs/2111.10919.

Deepanway Ghosal, Siqi Shen, Navonil Majumder, Rada Mihalcea, and Soujanya Poria. Cicero: A dataset for contextualized commonsense inference in dialogues. In *Annual Meeting of the Association for Computational Linguistics*, 2022. URL https://api.semanticscholar.org/CorpusID:247762111.

Amelia Glaese, Nat McAleese, Maja Trebacz, John Aslanides, Vlad Firoiu, Timo Ewalds, Maribeth Rauh, Laura Weidinger, Martin Chadwick, Phoebe Thacker, Lucy Campbell-Gillingham, Jonathan Uesato, Po-Sen Huang, Ramona Comanescu, Fan Yang, Abigail See, Sumanth Dathathri, Rory Greig, Charlie Chen, Doug Fritz, Jaume Sanchez Elias, Richard Green, Soňa Mokrá, Nicholas Fernando, Boxi Wu, Rachel Foley, Susannah Young, Iason Gabriel, William Isaac, John Mellor, Demis Hassabis, Koray Kavukcuoglu, Lisa Anne Hendricks, and Geoffrey Irving. Improving alignment of dialogue agents via targeted human judgements, 2022.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, et al. Reinforced self-training (rest) for language modeling. *arXiv preprint arXiv:2308.08998*, 2023.

Matthew J. Hausknecht, Prithviraj Ammanabrolu, Marc-Alexandre Côté, and Xingdi Yuan. Interactive fiction games: A colossal adventure. *CoRR*, abs/1909.05398, 2019. URL `http://arxiv.org/abs/1909.05398`.

Joey Hong, Sergey Levine, and Anca Dragan. Zero-shot goal-directed dialogue via rl on imagined conversations, 2023.

Youngsoo Jang, Jongmin Lee, and Kee-Eung Kim. GPT-critic: Offline reinforcement learning for end-to-end task-oriented dialogue systems. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=qaxhBG1UUaS`.

Natasha Jaques, Judy Hanwen Shen, Asma Ghandeharioun, Craig Ferguson, Àgata Lapedriza, Noah Jones, Shixiang Shane Gu, and Rosalind W. Picard. Human-centric dialog training via offline reinforcement learning. *CoRR*, abs/2010.05848, 2020. URL `https://arxiv.org/abs/2010.05848`.

Tomasz Korbak, Kejian Shi, Angelica Chen, Rasika Bhalerao, Christopher L. Buckley, Jason Phang, Samuel R. Bowman, and Ethan Perez. Pretraining language models with human preferences, 2023.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning, 2021.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *CoRR*, abs/2006.04779, 2020. URL `https://arxiv.org/abs/2006.04779`.

Sungjin Lee, Qi Zhu, Ryuichi Takanobu, Xiang Li, Yaoqin Zhang, Zheng Zhang, Jinchao Li, Baolin Peng, Xiujun Li, Minlie Huang, and Jianfeng Gao. Convlab: Multi-domain end-to-end dialog system platform. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *CoRR*, abs/2005.01643, 2020. URL `https://arxiv.org/abs/2005.01643`.

Yujia Li, David H. Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom, Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de, Masson d'Autume, Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey, Cherepanov, James Molloy, Daniel Jaymin Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de, Freitas, Koray Kavukcuoglu, and Oriol Vinyals. Competition-level code generation with alphacode. *Science*, 378:1092 – 1097, 2022. URL `https://api.semanticscholar.org/CorpusID:246527904`.

Xi Victoria Lin, Chenglong Wang, Luke Zettlemoyer, and Michael D. Ernst. Nl2bash: A corpus and semantic parser for natural language interface to the linux operating system. *CoRR*, abs/1802.08979, 2018. URL `http://arxiv.org/abs/1802.08979`.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. Agentbench: Evaluating llms as agents, 2023.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL `http://arxiv.org/abs/1907.11692`.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013. URL `http://arxiv.org/abs/1312.5602`.

OpenAI, :, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2023.

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 E. Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Francis Christiano, Jan Leike, and Ryan J. Lowe. Training language models to follow instructions with human feedback. *ArXiv*, abs/2203.02155, 2022. URL `https://api.semanticscholar.org/CorpusID:246426909`.

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

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *CoRR*, abs/1910.00177, 2019. URL `http://arxiv.org/abs/1910.00177`.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. URL https://api.semanticscholar.org/CorpusID:160025533.

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.

Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. Is reinforcement learning (not) for natural language processing?: Benchmarks, baselines, and building blocks for natural language policy optimization. 2022. URL https://arxiv.org/abs/2210.01241.

Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. Is reinforcement learning (not) for natural language processing: Benchmarks, baselines, and building blocks for natural language policy optimization, 2023.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks, 2015.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools, 2023.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. *CoRR*, abs/1506.02438, 2015. URL https://api.semanticscholar.org/CorpusID:3075448.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

Noah Shinn, Federico Cassano, Beck Labash, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. 2023. URL https://api.semanticscholar.org/CorpusID:258833055.

Charlie Snell, Ilya Kostrikov, Yi Su, Mengjiao Yang, and Sergey Levine. Offline rl for natural language generation with implicit language q learning, 2023.

Yuda Song, Yifei Zhou, Ayush Sekhari, J. Andrew Bagnell, Akshay Krishnamurthy, and Wen Sun. Hybrid rl: Using both offline and online data can make rl efficient, 2023.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In S. Solla, T. Leen, and K. Müller (eds.), *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. URL https://proceedings.neurips.cc/paper_files/paper/1999/file/464d828b85b0bed98e80ade0a5c43b0f-Paper.pdf.

Andrew Szot, Max Schwarzer, Harsh Agrawal, Bogdan Mazoure, Walter Talbott, Katherine Metcalf, Natalie Mackraz, Devon Hjelm, and Alexander Toshev. Large language models as generalizable policies for embodied tasks, 2023.

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.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. *CoRR*, abs/1509.06461, 2015. URL http://arxiv.org/abs/1509.06461.

Siddharth Verma, Justin Fu, Mengjiao Yang, and Sergey Levine. Chai: A chatbot ai for task-oriented dialogue with offline reinforcement learning, 2022.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi (Jim) Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *ArXiv*, abs/2305.16291, 2023a. URL https://api.semanticscholar.org/CorpusID:258887849.

Xingyao Wang, Zihan Wang, Jiateng Liu, Yangyi Chen, Lifan Yuan, Hao Peng, and Heng Ji. Mint: Evaluating llms in multi-turn interaction with tools and language feedback. *ArXiv*, abs/2309.10691, 2023b. URL https://api.semanticscholar.org/CorpusID:262053695.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 2004. URL https://api.semanticscholar.org/CorpusID:19115634.

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

Tengyang Xie, Ching-An Cheng, Nan Jiang, Paul Mineiro, and Alekh Agarwal. Bellman-consistent pessimism for offline reinforcement learning. *CoRR*, abs/2106.06926, 2021. URL https://arxiv.org/abs/2106.06926.

John Yang, Akshara Prabhakar, Karthik Narasimhan, and Shunyu Yao. Intercode: Standardizing and benchmarking interactive coding with execution feedback, 2023a.

Kaiyu Yang, Aidan M Swope, Alex Gu, Rahul Chalamala, Peiyang Song, Shixing Yu, Saad Godil, Ryan Prenger, and Anima Anandkumar. LeanDojo: Theorem Proving with Retrieval-Augmented Language Models. *arXiv preprint arXiv:2306.15626*, 2023b.

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

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023b.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. Rrhf: Rank responses to align language models with human feedback without tears, 2023.

Andrea Zanette. When is realizability sufficient for off-policy reinforcement learning?, 2023.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. Agenttuning: Enabling generalized agent abilities for llms, 2023.

Wenhao Zhan, Baihe Huang, Audrey Huang, Nan Jiang, and Jason D. Lee. Offline reinforcement learning with realizability and single-policy concentrability, 2022.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. *ArXiv*, abs/2307.13854, 2023a. URL https://api.semanticscholar.org/CorpusID:260164780.

Yifei Zhou, Ayush Sekhari, Yuda Song, and Wen Sun. Offline data enhanced on-policy policy gradient with provable guarantees, 2023b.

Qi Zhu, Zheng Zhang, Yan Fang, Xiang Li, Ryuichi Takanobu, Jinchao Li, Baolin Peng, Jianfeng Gao, Xiaoyan Zhu, and Minlie Huang. Convlab-2: An open-source toolkit for building, evaluating, and diagnosing dialogue systems. *CoRR*, abs/2002.04793, 2020. URL https://arxiv.org/abs/2002.04793.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul F. Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *CoRR*, abs/1909.08593, 2019. URL http://arxiv.org/abs/1909.08593.

Algorithm 2 ArCHer: Practical Framework

Initialize parameters $\phi, \psi, \theta, \bar{\theta}$, (Optionally) η
Initialize replay buffer $\mathcal{D}$ (optionally from an offline dataset).
for each iteration **do**
    ## Data Collection. $\triangleright$ [only online mode]
    for each environment step **do**
        Execute $a_t \sim \pi_\phi(\cdot|s_t)$, obtain the next state s_{t+1}, add to buffer $\mathcal{D}$.
    end for
    for each critic step **do**
        ## Update utterance-level Q and V functions by target function bootstrapping.
        $\theta \leftarrow \theta - \nabla J_\theta(Q)$ $\triangleright$ Equation 1
        $\psi \leftarrow \psi - \nabla J_\psi(V)$ $\triangleright$ Equation 2 or 6
        ## Update target Q and V functions.
        $\bar{\theta} \leftarrow (1-\tau)\bar{\theta} + \tau\theta$
        $\bar{\psi} \leftarrow (1-\tau)\bar{\psi} + \tau\psi$
    end for
    ## Update token-level baseline by MC regression.
    for each baseline step **do**
        $\eta \leftarrow \eta - \nabla J_\eta(\widetilde{V})$ $\triangleright$ (Optionally), Equation 4
    end for
    ## Update token-level actor with utterance-level critic.
    for each actor step **do**
        $\phi \leftarrow \phi - \nabla J_\phi(\pi)$ $\triangleright$ Equation 3, 5, or 7
    end for
end for

A Framework Summary

The overall procedure in the framework is summarized in Algorithm 2, where in practice we use the gradient estimated from a minibatch instead of the true gradient. In the ArCHer framework, we maintain a replay buffer $\mathcal{D}$ and we periodically update the critic and actor with data from the replay buffer. In the offline mode, $\mathcal{D}$ is set to be the offline dataset while in the online mode $\mathcal{D}$ is initialized to be empty and gradually expand as more online data is collected. The trajectories are stored in the replay buffer in the form of tuples $(s_t, a_t, r(s_t, a_t), s_{t+1})$ at the utterance level. We first update the utterance-level Q function by TD error with respect to the Bellman backup calculated by the target V function via Equation 1. The utterance-level V function is updated either with Equation 2 or 6. The target networks for Q and V functions are then updated as their polyak averages. Optinally, we can maintain another token-level baseline that is trained via MC regression in the token-level MDP as in Equation 4. Finally, the actor is updated with either vanilla REINFORCE as in Equation 3, REINFORCE with token-level baseline that reduces variances as in Equation 5, or AWR as in Equation 7 for conservatism in the offline setting.

B Theoretical Analysis

In additional to empirical experiments presented in Section 4, we also investigate the theoretical benefits of our hierarchical design. In particular, an important difference between ArCHer and a token-level RL algorithm such as ILQL (Snell et al., 2023) is an utterance-level critic. As the critic provides an estimate of on-policy advantages to calculate the policy gradients for updating the actor, it is critical to understand the estimation error of the on-policy advantages for an utterance-level critic and a token-level critic. This analysis is shared for both the online algorithm and the offline algorithm.

To start with, we need two assumptions regarding the capacity of the function class of the critic and the quality of the off-policy data in the replay buffer, both of which being standard in the RL theory literature. Intuitively, a token-level critic should require a stronger assumption on the capacity of the function class than an utterance-level critic. This is because some weaker function classes such as a smaller transformer might only be able to be accurate for each utterance instead for each token in the utterance. This intuition is captured theoretically by the Bellman Completeness Song et al. (2023); Zhou et al. (2023b); Zanette (2023); Xie et al. (2021) in Assumption 1 for the capacity of the function class, and Lemma 1 shows that indeed this assumption is weaker for an utterance-level critic than a token-level critic. Another assumption is about how related the off-policy data is for estimating on-policy advantages. Intuitively, if the off-policy data covers a completely different region from the on-policy distribution of states and actions, it will not help us to estimate on-policy advantages. A common assumption is Density Ratio Coefficient Zhan et al. (2022); Foster et al. (2021) in Assumption 2 that assumes all the on-policy states and actions are covered by the off-policy data. Lemma 2 states that the Density Ratio Coefficient for utterance-level and token-level critic is exactly the same. To summarize, theoretically a token-level critic requires the same data quality assumption while a stronger assumption of the function class compared to an utterance-level critic.

With the assumptions set up, we are able to use standard analysis for Fitted Policy Evaluation Zhou et al. (2023b); Song et al. (2023); Agarwal et al. (2019), and show a regret bound as in Theorem 1.

Theorem 1 (Main Theorem). *For an utterance-level MDP with discount factor γ, where L is the maximum length of each utterance, suppose utterance-level Assumption 1 and 2 holds, let f be the final Q function returned by Fitted Policy Evaluation formalized in Algorithm 3 at the utterance level, f yields a suboptimality gap of*

$$\begin{aligned}&\mathbb{E}_{s,a\sim d^\pi}[((\bar{f}(s,a)-\mathbb{E}_{a'\sim\pi(\cdot|s)}[\bar{f}(s,a)])-A^\pi(s,a))^2]\\ \leq&\frac{1}{\gamma L^{1/2}}\mathcal{O}(\frac{1}{(1-\gamma)(1-\gamma^{1/L})L^{1/2}}(\epsilon_{stat}+\sqrt{\epsilon_{stat}})).\end{aligned}$$

For an equivalent token-level MDP with discount factor $\gamma^{1/L}$, suppose token-level Assumption 1 and 2 holds, let f be the final Q function returned by Fitted Policy Evaluation formalized in Algorithm 3 at the token level, f yields a suboptimality gap of

$$\begin{aligned}&\mathbb{E}_{s,a\sim\widetilde{d}^\pi}[((\bar{f}(s,a)-\mathbb{E}_{a'\sim\pi(\cdot|s)}[\bar{f}(s,a)])-\widetilde{A}^\pi(s,a))^2]\\ \leq&\mathcal{O}(\frac{1}{(1-\gamma)(1-\gamma^{1/L})L^{1/2}}(\epsilon_{stat}+\sqrt{\epsilon_{stat}})),\end{aligned}$$

where ϵ_{stat} is the statistical error defined in Line 13 proportional to $N^{-1/2}$ the number of utterance-level transitions. This error term is defined the same for both utterance-level MDP and token-level MDP.

As shown in Theorem 1, the error of the estimates of the advantages for a token-level critic is $\gamma\sqrt{L}$ worse than that for an utterance-level critic, where L is the maximum number of tokens in each utterance. This is due to the error accumulation over a longer horizon. In practice, a common choice for γ is greater than 0.95 while L can be as large as 64 tokens. Our theoretical analysis shows that in addition to requiring a weaker assumption, using an utterance-level critic enjoys weaker assumptions and suffers less from error accumulation over the horizon.

C Proof of Main Theorem

C.1 Equivalent Utterance and Token Level MDPs

We consider the groundtruth utterance-level discounted infinite horizon MDP $\mathcal{M}=\{\mathcal{S},\mathcal{A},\gamma,r,\mu_0,P\}$ as defined in Section 3.1. An equivalent token-level infinite horizon MDP can be constructed with $\widetilde{\mathcal{M}}=\{\widetilde{\mathcal{S}},\widetilde{\mathcal{A}},\gamma^{1/L},r,\mu_0,\widetilde{P}\}$, where L is the length of each utterance, $\widetilde{\mathcal{A}}$ contains all individual tokens with a special padding token (such that each utterance is padded to the same length L), $\widetilde{\mathcal{S}}\in\mathcal{S}\times\widetilde{A}^L$ contains both the state in the utterance-level MDP and the partial sequence of utterance that has already been generated, $\gamma^{1/L}$ is the equivalent discount factor. Note that this definition of token-level MDP is not the same as the definition of token-level MDP in the hierarchical language MDP defined for ArCHer in Section 3.1. The token-level MDP for ArCHer is only embedded in one particular utterance and the only reward that it receives is at the end of the utterance from the utterance-level critic while the equivalent token-level MDP spans multiple utterances and receives rewards from the external environment at the end of each utterance. This construction of token-level MDP is "equivalent" to the utterance-level MDP in the sense that for any autoregressive policy π that generates an utterance token by token, we have:

$$\forall s \in \mathcal{S}, V^\pi(s) = \widetilde{V}^\pi(s),$$

where V^π and $\widetilde{V}^\pi$ are value functions of π in the utterance-level MDP $\mathcal{M}$ and the token-level MDP $\widetilde{\mathcal{M}}$ respectively. We use $\widetilde{\pi}$ for the same utterance-level policy π when it generates one token at a time.

As usual, for any MDP $\mathcal{M}$, we define $d^\pi \in \Delta(\mathcal{S} \times \mathcal{A})$ as the average state-action occupancy measure of policy π. We denote $V^\pi = \mathbb{E}_{s_0 \sim \mu_0} V^\pi(s_0)$ as the expected total discounted reward of π. We denote $\mathcal{T}^\pi$ as the Bellman operator associated with π, i.e., given a function $f \in \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$, we have

$$\mathcal{T}^\pi f(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim P(s,a), a' \sim \pi(s')}[f(s',a')].$$

Similar definitions can be made in the token-level MDP $\widetilde{\mathcal{M}}$ for $\widetilde{d}^\pi$, $\widetilde{V}^\pi$, and $\widetilde{\mathcal{T}}^\pi$. We also define $\widetilde{a}^\pi$ as the advantage function in the token level.

C.2 Fitted Policy Evaluation Subroutine

In this section, we present our theoretical subroutine for fitting the critic. On a high level, it just repeats finding the Q function that minimizes the bellman error with respect to the Q function in the last iteration, and returns the average of all Q functions in the end. Both critic fitting in the token-level MDP $\widetilde{\mathcal{M}}$ or in the utterance-level MDP $\mathcal{M}$ follows from the same subroutine with the same function class $\mathcal{F}$ that map the space of seuqnces of tokens to real values. This theoretical algorithm is simply a more fomalized version of the critic update in Algorithm 1.

Algorithm 3 **F**itted **P**olicy **E**valuation (**FPE**)

Require: Policy π, function class $\mathcal{F}$, number of iterations K, weight λ

Require: K independent datasets $\mathcal{D}_{1:K} = \{(s,a,r,s')\}$ of M many samples each from the same offline distribution ν.

1: Initialize $f_0 \in \mathcal{F}$.
2: **for** $k = 1, \ldots, K$ **do**
3: Solve the square loss regression problem to compute:

$$f_k \leftarrow \underset{f \in \mathcal{F}}{\operatorname{argmin}}\ \widehat{\mathbb{E}}_{\mathcal{D}_k}[(f(s,a) - r - \gamma f_{k-1}(s', \pi(s')))^2] \tag{8}$$

4: **end for**
5: **Return** $\bar{f} = \frac{1}{K}\sum_{k=1}^K f_k$.

C.3 Assumptions

We present the two important assumption that we use for analyzing FPE subroutine, and both assumptions share the same definition for utterance-level MDP $\mathcal{M}$ and token-level MDP $\widetilde{\mathcal{M}}$.

Assumption 1 (Bellman Completeness)**.** *We say that $\mathcal{F}$ is Bellman Complete for some policy π, if for all $f \in \mathcal{F}$, there exists a $f' \in \mathcal{F}$ such that $|f'(s,a) - \mathcal{T}^\pi f(s,a)\|_\infty = 0$.*

Assumption 2 (Density Ratio Coefficient)**.** *Given the offline distribution ν, for any policy π^e, we define the Density Ratio Coefficient as*

$$C_{\nu,\pi} := \max_{s,a} \frac{d^\pi(s,a)}{\nu(s,a)}.$$

The following two lemmas compare the utterance-level assumptions and token-level assumptions.

Lemma 1. *For any stationary policy π, token-level Bellman Completeness for $\widetilde{\mathcal{M}} \implies$ utterance-level Bellman Completeness for $\mathcal{M}$*

Proof. $\forall s \in \mathcal{S}, a \in \mathcal{A}$ being state and action in the utterance-level, the utterance a can be decomposed into L tokens in the action space of the token-level MDP:

$$a = \widetilde{a}_{1:L}, \widetilde{a}_i \in \widetilde{\mathcal{A}}, i = 1, \ldots, L.$$

Therefore,

$$
\begin{aligned}
&\min_{f'\in\mathcal{F}} |f'(s,a) - \mathcal{T}^\pi(s,a)| \\
=&\min_{f_1,\dots,f_L\in\mathcal{F}} |f_1(s,a) - \widetilde{\mathcal{T}}^\pi f_2(s,a) + r(s,a) + \gamma^{1/L}\mathbb{E}_{s'\sim P(\cdot|s,a),\widetilde{a}_1\sim\widetilde{\pi}(\cdot|s')} f_2(s',\widetilde{a}_1) \\
&- \gamma^{1/L}\mathbb{E}_{s'\sim P(\cdot|s,a),\widetilde{a}_1\sim\widetilde{\pi}(\cdot|s')}\widetilde{\mathcal{T}}^\pi f_3(s',\widetilde{a}_1) + \dots + \gamma^{(L-1)/L}\mathbb{E}_{s'\sim P(\cdot|s,a),\widetilde{a}_{1:L-1}\sim\widetilde{\pi}(\cdot|s')} f_L(s',\widetilde{a}_{1:L-1}) \\
&- r(s,a) - \gamma^{(L-1)/L}\mathbb{E}_{s'\sim P(\cdot|s,a),\widetilde{a}_{1:L-1}\sim\widetilde{\pi}(\cdot|s')}\widetilde{\mathcal{T}}^\pi f(s',\widetilde{a}_{1:L-1})| \\
\le&\min_{f_1,\dots,f_L\in\mathcal{F}} |f_1(s,a) - \widetilde{\mathcal{T}}^\pi f_2(s,a)| \sum_{i=2}^{L} \gamma^{(i-1)/L}\mathbb{E}_{s'\sim P(\cdot|s,a),\widetilde{a}_{1:i-1}\sim\widetilde{\pi}(\cdot|s'))} |f^i(s,\widetilde{a}_{1:i-1}) - \widetilde{\mathcal{T}}^\pi((s,\widetilde{a}_{1:i-1})| \\
\le& 0,
\end{aligned}
$$

where the last inequality follows from the assumption of token-level Bellman Comlete. $\square$

The next lemma assumes that the offline distribution ν is the same for both token-level MDP and utterance-level MDP, i.e. the token-level transitions are derived by splitting utterance-level transitions. We use $\widetilde{\nu}$ to denote the token-level offline distribution created in this way.

Lemma 2. *For any stationary policy π, and any offline distribution ν, we have Density Ratio Coefficient for token-level $\widetilde{\mathcal{M}}$ = Density Ratio Coefficient for utterance-level $\mathcal{M}$.*

$$
C_{\nu,\pi} = C_{\widetilde{\nu},\pi}
$$

Proof. The proof is constructed by noticing the fact that each token-level state consists of not only the utterance-level state but also all the tokens that have been generated in the current utterance:

$$
\max_{\widetilde{s},\widetilde{a}} \frac{\widetilde{d^\pi}(\widetilde{s},\widetilde{a})}{\widetilde{\nu}(\widetilde{s},\widetilde{a})} = \max_{s,\widetilde{a}_{1:L},i} \frac{\widetilde{d^\pi}(s,\widetilde{a}_{1:i})}{\widetilde{\nu}(\widetilde{s},\widetilde{a}_{1:i})} = \max_{s,\widetilde{a}_{1:L}} \frac{\widetilde{d^\pi}(s,\widetilde{a}_{1:L})}{\widetilde{\nu}(\widetilde{s},\widetilde{a}_{1:L})} = \max_{s,a} \frac{d^\pi(s,a)}{\nu(s,a)}
$$

$\square$

C.4 Technical Lemmas

First, we would like to prove an interesting lemma that allows us to compare the different discount factors for token level $\gamma^{1/L}$ and utterance level γ.

Lemma 3. *With the discount factor $\gamma \in (0,1)$, and L being a positive number, we have that:*

$$
\begin{aligned}
\frac{1}{1-\gamma} =& \frac{1}{(1-\gamma^{1/L})(1+\gamma^{1/L}+\gamma^{2/L}+\dots+\gamma^{(L-1)/L})} \\
\le& \frac{1}{L\gamma(1-\gamma^{1/L})}
\end{aligned}
$$

Below are some common technical lemmas useful for reinforcement learning.

Lemma 4 ((Zhou et al., 2023b, Lemma 2))**.** *For any policy π, and non-negative function $g(s,a)$, we have:*

$$
\mathbb{E}_{\bar{s},\bar{a}\sim d^\pi}\mathbb{E}_{s\sim P(\cdot|\bar{s},\bar{a}),a\sim\pi(a|s)}[g(s,a)] \le \frac{1}{\gamma}\mathbb{E}_{s,a\sim d^\pi}[g(s,a)].
$$

where μ_0 denotes the initial state distribution (which is the same for all policies π).

Proof. Recall that $\lim_{h\to\infty}\gamma^h = 0$. We start by noting that:

$$\begin{aligned}
d^\pi(s,a) &= (1-\gamma)(\mu_0(s,a) + \gamma d_1^\pi(s,a) + \gamma^2 d_2^\pi(s,a) + \dots) && (9)\\
&\geq \gamma(1-\gamma)\left(\sum_{\bar{s},\bar{a}} \mu_0(\bar{s},\bar{a})P(s|\bar{s},\bar{a})\pi(a|s) + \gamma \sum_{\bar{s},\bar{a}} d_1^\pi(\bar{s},\bar{a})P(s|\bar{s},\bar{a})\pi(a|s) + \dots\right)\\
&= \gamma(1-\gamma)\sum_{\bar{s},\bar{a}} (\mu_0(\bar{s},\bar{a}) + \gamma d_1^\pi(\bar{s},\bar{a}) + \dots)\, P(s|\bar{s},\bar{a})\pi(a|s)\\
&= \gamma \sum_{\bar{s},\bar{a}} d^\pi(\bar{s},\bar{a})P(s|\bar{s},\bar{a})\pi(a|s) && (10)\\
&= \gamma \mathbb{E}_{\bar{s},\bar{a}\sim d^\pi}[P(s|\bar{s},\bar{a})\pi(a|s)],
\end{aligned}$$

where Line 10 follows by plugging in the relation in Line 9 for $\bar{s},\bar{a}$. The above implies that for any function $g \geq 0$,

$$\sum_{s,a} d^\pi(s,a)g(s,a) \geq \sum_{s,a} \gamma \mathbb{E}_{\bar{s},\bar{a}\sim d^\pi}[P(s|\bar{s},\bar{a})\pi(a|s)g(s,a)],$$

which implies that

$$\mathbb{E}_{\bar{s},\bar{a}\sim d^\pi}\mathbb{E}_{s\sim P(\cdot|\bar{s},\bar{a}),a\sim\pi(a|s)}[g(s,a)] \leq \frac{1}{\gamma}\mathbb{E}_{s,a\sim d^\pi}[g(s,a)].$$

□

Lemma 5 (Least squares generalization bound, (Song et al., 2023, Lemma 3))**.** *Let $R > 0$, $\delta \in (0,1)$, and consider a sequential function estimation setting with an instance space $\mathcal{X}$ and target space $\mathcal{Y}$. Let $\mathcal{H} : \mathcal{X} \mapsto [-R, R]$ be a class of real valued functions. Let $\mathcal{D} = \{(x_1, y_1), \dots, (x_T, y_T)\}$ be a dataset of T points where $x_t \sim \rho_t := \rho_t(x_{1:t-1}, y_{1:t-1})$, and y_t is sampled via the conditional probability $p_t(x_t)$ (which could be adversarially chosen). Additionally, suppose that $\max_t |y_t| \leq R$ and $\max_h \max_x |h(x)| \leq R$. Then, the least square solution $\widehat{h} \leftarrow \mathrm{argmin}_{h\in\mathcal{H}} \sum_{t=1}^T (h(x_t) - y_t)^2$ satisfies*

$$\sum_{t=1}^T \mathbb{E}_{x\sim\rho_t, y\sim p_t(x)}\left[(\widehat{h}(x) - y)^2\right] \leq \inf_{h\in\mathcal{H}} \sum_{t=1}^T \mathbb{E}_{x\sim\rho_t, y\sim p_t(x)}\left[(h(x) - y)^2\right] + 256R^2\log(2|\mathcal{H}|/\delta)$$

with probability at least $1-\delta$.

C.5 Proof of Main Theorem

Now we are ready to analyze the sample complexity of Fitted Policy Evaluation in Algoritm 3.

Lemma 6 (Guarantees of FPE)**.** *For the algorithm described in Algorithm 3 with K independent datasets $\mathcal{D}_{1:K} = \{(s,a,r,s')\}$ such that the effective number of samples $N = MK$, if the function class satisfies $\max_{f\in\mathcal{F}, s\in\mathcal{S}, a\in\mathcal{A}} |f(s,a)| < R$, then with probability at least $1-\delta$ the output value function satisfies:*

$$\mathbb{E}_{s,a\sim d^\pi}[(\bar{f}(s,a) - \mathcal{T}^\pi \bar{f}(s,a))^2] \leq C_{\nu,\pi}(\frac{4(R+1)^2}{K} + \frac{256R^2\log(2|\mathcal{F}|K/\delta)}{M}).$$

Proof. By applying Lemma 5, to each iteration $1,\dots,K$, and apply a union bound over all iterations, we have that:

$$\forall k, M\mathbb{E}_{s,a\sim\nu}[(f_k(s,a) - \mathcal{T}^\pi f_{k-1}(s,a))^2] \leq M \inf_{f\in\mathcal{F}} \mathbb{E}_{s,a\sim\nu}[(f(s,a) - \mathcal{T}^\pi f_{k-1}(s,a))^2] + 256R^2\log(2|\mathcal{F}|K/\delta)$$

$$\forall k, \mathbb{E}_{s,a\sim\nu}[(f_k(s,a) - \mathcal{T}^\pi f_{k-1}(s,a))^2] \leq \frac{256R^2\log(2|\mathcal{F}|K/\delta)}{M}, \tag{11}$$

where the second line holds by Assumption 1. To combine the guarantees that we have from each round to the guarantee of the returned value function $\bar{f}$:

$$
\begin{aligned}
&\mathbb{E}_{s,a\sim\nu}[(\bar{f}(s,a)-\mathcal{T}^\pi \bar{f}(s,a))^2] \\
=&\mathbb{E}_{s,a\sim\nu}[(\frac{1}{K}(f_1(s,a)-\mathcal{T}^\pi f_K(s,a)+\sum_{k=2}^{K} f_k(s,a)-\mathcal{T}^\pi f_{k-1}(s,a)))^2] \\
\leq&\frac{1}{K}\mathbb{E}_{s,a\sim\nu}[(f_1(s,a)-\mathcal{T}^\pi f_K(s,a))^2+\sum_{k=2}^{K}(f_k(s,a)-\mathcal{T}^\pi f_{k-1}(s,a))^2] \\
\leq&\frac{1}{K}[4(R+1)^2+(K-1)\frac{256R^2\log(2|\mathcal{F}|K/\delta)}{M}] \\
\leq&\frac{4(R+1)^2}{K}+\frac{256R^2\log(2|\mathcal{F}|K/\delta)}{M},
\end{aligned}
\tag{12}
$$

where the first inequality follows by Jensen's Inequality and the second inequality follows by plugging in Line 11. Then, we can plug in Assumption 2 to conclude the proof.

$$
\begin{aligned}
\mathbb{E}_{s,a\sim d^\pi}[(\bar{f}(s,a)-\mathcal{T}^\pi \bar{f}(s,a))^2] \leq& C_{\nu,\pi}\mathbb{E}_{s,a\sim\nu}[(\bar{f}(s,a)-\mathcal{T}^\pi \bar{f}(s,a))^2] \\
\leq& C_{\nu,\pi}(\frac{4(R+1)^2}{K}+\frac{256R^2\log(2|\mathcal{F}|K/\delta)}{M}). \\
\leq& 64C_{\nu,\pi}R(R+1)\sqrt{\frac{\log(2|\mathcal{F}|K/\delta)}{KM}} \\
=& 64C_{\nu,\pi}R(R+1)\sqrt{\frac{\log(2|\mathcal{F}|K/\delta)}{N}} \\
:=& C_{\nu,\pi}\epsilon_{stat}
\end{aligned}
\tag{13}
$$

□

Theorem 1 (Main Theorem)**.** *For an utterance-level MDP with discount factor γ^L, where L is the maximum length of each utterance, suppose utterance-level Assumption 1 and 2 holds, let f be the final Q function returned by Fitted Policy Evaluation formalized in Algorithm 3 at the utterance level, yields a suboptimality gap of*

$$
\begin{aligned}
&\mathbb{E}_{s,a\sim d^\pi}[((\bar{f}(s,a)-\mathbb{E}_{a'\sim\pi(\cdot|s)}[\bar{f}(s,a)])-A^\pi(s,a))^2] \\
\leq&\frac{4}{1-\gamma}(\epsilon_{stat}+2(R+1)\sqrt{\epsilon_{stat}}) \\
\leq&\frac{4}{\gamma L(1-\gamma^{1/L})}(\epsilon_{stat}+2(\frac{1}{1-\gamma}+1)\sqrt{\epsilon_{stat}}).
\end{aligned}
$$

For an equivalent token-level MDP with discount factor $\gamma^{1/L}$, suppose token-level Assumption 1 and 2 holds, let f be the final Q function returned by Fitted Policy Evaluation formalized in Algorithm 3 at the token level, yields a suboptimality gap of

$$
\begin{aligned}
&\mathbb{E}_{s,a\sim d^\pi}[((\bar{f}(s,a)-\mathbb{E}_{a'\sim\pi(\cdot|s)}[\bar{f}(s,a)])-\widetilde{A}^\pi(s,a))^2] \\
\leq&\frac{4}{(1-\gamma^{1/L})L^{1/2}}(\epsilon_{stat}+2(\frac{1}{1-\gamma}+1)\sqrt{\epsilon_{stat}}L^{1/4}),
\end{aligned}
$$

where ϵ_{stat} is the statistical error defined in Line 13 proportional to $N^{-1/2}$ the number of utterance-level transitions. This error term is defined the same for both utterance-level MDP and token-level MDP.

Proof. First, we start with analyzing utterance-level FPE with effective number of samples, we start by bounding the errors of the Q functions:

$$\begin{aligned}
&\mathbb{E}_{s,a\sim d^\pi}(\bar{f}(s,a) - Q^\pi(s,a))^2\\
=&\mathbb{E}_{s,a\sim d^\pi}(\bar{f}(s,a) - \mathcal{T}^\pi(s,a) + \mathcal{T}^\pi\bar{f}(s,a) - Q^\pi(s,a))^2\\
=&\mathbb{E}_{s,a\sim d^\pi}[(\bar{f}(s,a) - \mathcal{T}^\pi(\bar{f}s,a))^2 + 2(\bar{f}(s,a) - \mathcal{T}^\pi(s,a))(\mathcal{T}^\pi\bar{f}(s,a) - Q^\pi(s,a)) + (\mathcal{T}^\pi\bar{f}(s,a) - Q^\pi(s,a))^2]\\
\leq&\mathbb{E}_{s,a\sim d^\pi}[(\bar{f}(s,a) - \mathcal{T}^\pi(\bar{f}s,a))^2] + 2\sqrt{\mathbb{E}_{s,a\sim d^\pi}(\bar{f}(s,a) - \mathcal{T}^\pi(s,a))^2(\mathcal{T}^\pi\bar{f}(s,a) - Q^\pi(s,a))^2}\\
&+ \mathbb{E}_{s,a\sim d^\pi}[(\mathcal{T}^\pi\bar{f}(s,a) - Q^\pi(s,a))^2]\\
\leq&\mathbb{E}_{s,a\sim d^\pi}[(\bar{f}(s,a) - \mathcal{T}^\pi(\bar{f}s,a))^2] + 2(R+1)\sqrt{\mathbb{E}_{s,a\sim d^\pi}(\bar{f}(s,a) - \mathcal{T}^\pi(s,a))^2} + \mathbb{E}_{s,a\sim d^\pi}[(\mathcal{T}^\pi\bar{f}(s,a) - Q^\pi(s,a))^2]\\
=&\mathbb{E}_{s,a\sim d^\pi}[(\bar{f}(s,a) - \mathcal{T}^\pi(\bar{f}s,a))^2] + 2(R+1)\sqrt{\mathbb{E}_{s,a\sim d^\pi}(\bar{f}(s,a) - \mathcal{T}^\pi(s,a))^2}\\
&+ \gamma^2\mathbb{E}_{s,a\sim d^\pi, s'\sim P(\cdot|s,a), a'\sim\pi(\cdot|s')}[(\mathcal{T}^\pi\bar{f}(s',a') - Q^\pi(s',a'))^2]\\
\leq&\mathbb{E}_{s,a\sim d^\pi}[(\bar{f}(s,a) - \mathcal{T}^\pi(\bar{f}s,a))^2] + 2(R+1)\sqrt{\mathbb{E}_{s,a\sim d^\pi}(\bar{f}(s,a) - \mathcal{T}^\pi(s,a))^2} + \gamma\mathbb{E}_{s,a\sim d^\pi}(\bar{f}(s,a) - Q^\pi(s,a))^2,
\end{aligned}$$

where the first inequality follows from Jensen's inequality, and the second inequality follows from $\max_{s,a}\max\{|\mathcal{T}^\pi\bar{f}(s,a)|, |Q^\pi(s,a)|\} \leq R+1$, and the third inequality follows from Lemma 4. Finally, we get:

$$\begin{aligned}
&\mathbb{E}_{s,a\sim d^\pi}(\bar{f}(s,a) - Q^\pi(s,a))^2\\
\leq&\frac{1}{1-\gamma}(\mathbb{E}_{s,a\sim d^\pi}[(\bar{f}(s,a) - \mathcal{T}^\pi(\bar{f}s,a))^2] + 2(R+1)\sqrt{\mathbb{E}_{s,a\sim d^\pi}(\bar{f}(s,a) - \mathcal{T}^\pi(s,a))^2})\\
=&\frac{1}{1-\gamma}(\epsilon_{stat} + 2(R+1)\sqrt{\epsilon_{stat}}),
\end{aligned}$$

where ϵ_{stat} is defined in Line 13. Finally, we can directly bound the errors for the advantages:

$$\begin{aligned}
&\mathbb{E}_{s,a\sim d^\pi}[((\bar{f}(s,a) - \mathbb{E}_{a'\sim\pi(\cdot|s)}\bar{f}(s,a)) - A^\pi(s,a))^2]\\
=&\mathbb{E}_{s,a\sim d^\pi}[((\bar{f}(s,a) - \mathbb{E}_{a'\sim\pi(\cdot|s)}\bar{f}(s,a)) - (Q^\pi(s,a) - \mathbb{E}_{a'\sim\pi(\cdot|s)Q^\pi(s,a)}))^2]\\
\leq&2\mathbb{E}_{s,a\sim d^\pi}[(\bar{f}(s,a) - Q^\pi(s,a))^2] + 2\mathbb{E}_{s\sim d^\pi}[(\mathbb{E}_{a'\sim\pi(\cdot|s)}[\bar{f}(s,a') - Q^\pi(s,a')])^2]\\
\leq&4\mathbb{E}_{s,a\sim d^\pi}[(\bar{f}(s,a) - Q^\pi(s,a))^2]\\
\leq&\frac{4}{1-\gamma}(\epsilon_{stat} + 2(R+1)\sqrt{\epsilon_{stat}}),\\
\leq&\frac{4}{(1-\gamma^{1/L})\gamma L}(\epsilon_{stat} + 2(R+1)\sqrt{\epsilon_{stat}})\\
=&\frac{4}{(1-\gamma^{1/L})\gamma L}(\epsilon_{stat} + 2(\frac{1}{1-\gamma}+1)\sqrt{\epsilon_{stat}})\\
&,
\end{aligned}$$

where the second inequality follows from Jensen's inequality, and the fourth inequality follows from Lemma 3. A similar analysis can be done with token-level FPE, noticing that the effective number of samples for token-level FPE is NL because each utterance transition can be splitted into L token transitions:

$$\begin{aligned}
\mathbb{E}_{s,a\sim\widetilde{d}^\pi}[((\bar{f}(s,a) - \mathbb{E}_{a'\sim\pi(\cdot|s)}\bar{f}(s,a)) - \widetilde{A}^\pi(s,a))^2] &\leq\frac{4}{1-\gamma}(\epsilon_{stat}L^{-1/2} + 2(R+1)\sqrt{\epsilon_{stat}}L^{-1/4}).\\
&\leq\frac{4}{(1-\gamma)L^{1/2}}(\epsilon_{stat} + 2(\frac{1}{1-\gamma}+1)\sqrt{\epsilon_{stat}}L^{1/4}).
\end{aligned}$$

□

D Environment and Dataset Details

In this section, we provide more details for the environments and datasets that we used in our experiments. Example actions and observations for each environment are shown in Figure 5.

Detective Game Hausknecht et al. (2019). In this game, the agent plays the role of a detective looking into a case where the Mayor got murdered. At each time step, the agent generates a free-form text where the game engine parses the text and determines the next state at each time step. The game engine provides a feedback of "Illegal Action." if the generated text is an illegal action or cannot be correctly parsed. The optimal policy takes 51 steps to solve and reaches a maximum reward of 360. The game timeouts and terminates after 60 steps (including steps where illegal actions are generated). The observation at each time step includes the current surroundings, items carried, environment feedback for the outcome of the last action, and a list of available actions. The Supervised Fine-Tuning (SFT) dataset for this environment consists of 1000 trajectories of an agent picking a random action from the list of available actions at each timestep.

Detective Game

Action: take paper

Observation:

<< Chief's office >>
You are standing in the Chief's office. He is telling you "The Mayor was murdered yeaterday night at 12:03 am. I want you to solve it before we get any bad publicity or the FBI has to come in. "Yessir!" You reply. He hands you a sheet of paper. Once you have read it, go north or west.

You are carrying:
a piece of white paper

Taken.

[Your score has just gone up by ten points.]
['east', 'west', 'put paper down', 'north']

Guess My City

Action: Is your city located in Europe?

Observation:

Questions:
What is the primary language spoken in your city?
The primary language spoken in the city is Spanish.

Is your city located on the east coast of the country?
Yes, it is located on the east coast of the country.

Is your city located in North America?
No, my city is not located in North America.

Is your city located in Europe?
Yes, my city is located in Europe.

Twenty Questions

Action: Is the object an artwork?

Observation:

Questions:
Is the object alive?
No.
Is the object man-made?
Yes.
Is the object a vehicle?
No.
Is the object a piece of furniture?
No.
Is the object a piece of clothing?
No.
Is the object a piece of technology?
No.
Is the object a piece of jewelry?
No.
Is the object a piece of clothing?
No.
Is the object an artwork?
No.

Web Shopping

Action:

search[wireless bluetooth 4.0 power amplifier board]

Observation:

WebShop
Instruction:
i am looking for a wireless bluetooth 4.0 power amplifier board, and price lower than 60.00 dollars
[Search]
Action:
search[wireless bluetooth 4.0 power amplifier board]
Observation:
[Back to Search]
Page 1 (Total results: 50)
[Next >]
[B08LDPQCB4]
Audio Amplifier Module,TDA7492P Wireless Bluetooth 4.0 Audio Receiver Power Amplifier Board Module,CSR8635 Bluetooth Chip,Auto Pairing,Amplifier Speakers Board
$25.19
[B08GPVK8LB]
Liccx TDA7492P Power Amplifier Board£¬Wireless Bluetooth 4.0 Audio Receiver Power Amplifier Board Module Audio Amplifier Module
$24.99

Figure 5: Example actions and observations for each environment

Twenty Questions and Twenty Questions Subset Abdulhai et al. (2023). In this environment, for each episode, a random word is chosen from a list of 157 words of household items such as "basketball", "apple", and "car". The word is held hidden from the agent and the agent is tasked to guess the hidden word within 20 questions. The questions are limited to yes/no questions and the answers from the oracle are limited to "Yes.", "No.", and "Invalid Question.". As opposed to using "flan-t5-xl"Chung et al. (2022) as the oracle Abdulhai et al. (2023), we train a 'flan-t5-small" to simulate the oracle with the same data and use it for our online experiments due to computational constraints. The agent gets a reward of 0 if it guesses the correct word and the episode terminates. Otherwise, the agent gets a reward of -1 for each question it raises. This reward structure results in a minimum reward of -20 if the agent does not guess the corret word with twenty questions and a maximum reward of 0 if the agent guesses the correct word with the first question although it is very unlikely. We use the official offline dataset provided by Abdulhai et al. (2023) with 100K simulated episodes. Our SFT checkpoints for online experiments for both Twenty Questions and Twenty Questions Subset are also trained with this dataset. Twenty Questions Subset keeps everything else the same except that it uses a subset of 10 hidden words in the word list. Since the offline dataset and the SFT checkpoint for online experiments are based on the entire Twenty Questions, Twenty Questions Subset challenges different algorithms with a significant distribution shift and requires the agent to come up with an entirely different strategy from behavior cloning.

Guess My City Abdulhai et al. (2023). This environment is a similar dialogue task to Twenty Questions. For each episode, a random city is chosen from a list of 100 cities in the world. The city is held hidden from the agent and the agent is tasked to guess the name of the city within 20 questions. Both the questions and answers are free-form except that the answers are not allowed to contain the name of the city. As opposed to using "flan-t5-xl"Chung et al. (2022) as the oracle Abdulhai et al. (2023), we train a 'flan-t5-small" to simulate the oracle with the same data and use it for our online experiments due to computational constraints. We found in our online experiments that the agent can easily learn to "exploit" the oracle by tricking it to directly output the name of the city. Therefore, we simply replace the answer with a hardcoded template "I cannot answer that question." if the name of the city is found in the output of the oracle language model to reduce reward hacking. The reward structure is the same as Twenty Questions. We use the official offline dataset provided by Abdulhai et al. (2023) with 100K simulated episodes.

Web Shopping Yao et al. (2023a). This environment challenges the ability of the agents to interact with external tools. For each episode, a random instruction requesting a specific item is chosen and shown to the agent. The agent needs to make use of a simplified web shopping server to make the purchase. Every successful purchase is consisted of searching the keywords in the search engine, selecting an item from searched results, clicking on features and attributes for the item, and finally making the purchase. Followimg ReAct Yao et al. (2023b), the agent can choose to take a "think" action before taking any actual actions such as "search" adn "click". The observation contains the instruction and a history of visited webpages (described in text) and actions. The reward is a scalar between 0 and 1 depending on the similarity of the purchased item with the requested item. For example, a partial reward will be given if the agent purchases a black king-sized bed while a black queen-sized bed is requested. The episode timeouts after 10 interaction steps and a reward of 0 is issued. Our main online environments use a subset of 100 instructions from index 2000 to 2100 for a fast evaluation. We collect the offline dataset using the insrtructions from index 0 to 1000 with GPT-3 text-davinci-002 with prompts from ReAct's official implementation.

E OFFLINE ALGORITHM AND PRACTICAL CONSIDERATION

Our offline algorithm is a hierarchical version of the IQL algorithm Kostrikov et al. (2021) according to the ArCHer framework. More specifically, the critic leverages Equation (6) while the actor update is based on Equation (7).

This choices for the actor and for the critic update identify two key hyperparameters, the expectile value τ (defined in Equation 6 and 7) and the temperature β, whose effect is described in the respective sections. These hyper-parameters are already present in the original IQL algorithm Kostrikov et al. (2021), and they have a similar interepretation here. By choosing τ and β appropriately, the algorithm identifies a policy whose performance should be between the optimal one and the one that generated the dataset. (In general, recovering the optimal policy by just using a dataset may not be possible as the dataset may not contain information about an optimal policy).

The offline algorithm shares most of the ingredients with its online counterpart, such as the double critic, target networks, soft updates, and value function heads. However, some unique features inherited from IQL allow to considerably simplify several algorithmic choices.

- The actor and the critic no longer need to be synchronized by using a certain update ratio. This is because the critic update defined in Equation (6) is independent of the actor's current policy, and so the two can be updated with any desired frequency without introducing instabilities.
- It is not necessary to pre-train the policy with a behavioural cloning objective, because such objective is already included in the actor's loss function in Equation (7).
- The warmup steps for the critic are also not necessary, because the initially small advantage function has a neglegible effect in the AWR loss.

F Additional Baseline Details

F.1 Performance of PPO

In Figure 6, we provide a zoom-in of the learning curves of PPO for Twenty Questions, Twenty Questions Subset, and Guess My City. We observed that PPO does improve over the SFT checkpoint, especially in the more simple task Twenty Questions Subset. However, as PPO is unable to reuse past off-policy data, we need to collect at least 1024 trajectories of on-policy data for each PPO update, as shown in Appendix I. This observation is consistent with Abdulhai et al. (2023).

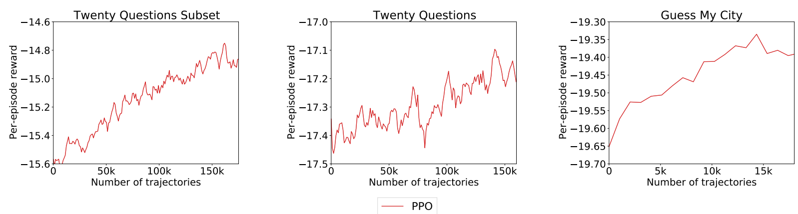


Figure 6: A zoom-in of learning curves of PPO. PPO gradually improves despite its worse sample complexity compared to other off-policy methods.

F.2 Additional Reproduction Details for WebShop Experiment

For our WebShop experiment, we utilized the environment and the few-shot prompting baselines from ReAct Yao et al. (2023b). ReAct introduces two prompting strategies: the ReAct method and the Act-only method, which we denote as *ReAct* and *expert-prompt* in Figure 3. The ReAct method additionally allows the agent to articulate its reasoning before making an action, whereas the Act-only method does not. We use the original prompts in ReAct's implementation without any modifications, and to ensure that our paper is self-contained, we also include these prompts in Figure 8 and 7.

We assessed these algorithms on the webshop index from 2000 to 2100 to establish the LLM-based few-shot baselines in Figure 3. Notably, by early January 2024, OpenAI had deprecated the GPT-3 text-davinci-002 model used in the original ReAct study. Consequently, we switched to the gpt-3.5-turbo-instruct model. While the expert-prompt approach yielded performance comparable to the best performing text-davinci-002 based method, the ReAct method underperformed significantly with the newer model. This discrepancy, also observed and replicated in other tasks independetly by other researchers [1], is faithfully depicted in Figure 3.

[1]For more details, see the Github Issue on ReAct's repository: `https://github.com/ysymyth/ReAct`

ReAct Prompt
Webshop
Instruction:
i would like a 3 ounce bottle of bright citrus deodorant for sensitive skin, and price lower than 50.00 dollars
[Search]
Action: search[3 ounce bright citrus deodorant sensitive skin]
Observation:
[Back to Search]
Page 1 (Total results: 50)
[Next >]
[B078GWRC1J]
Bright Citrus Deodorant by Earth Mama — Natural and Safe for Sensitive Skin, Pregnancy and Breastfeeding, Contains Organic Calendula 3-Ounce
$10.99
[B078GTKVXY]
Ginger Fresh Deodorant by Earth Mama — Natural and Safe for Sensitive Skin, Pregnancy and Breastfeeding, Contains Organic Calendula 3-Ounce
$10.99
[B08KBVJ4XN]
Barrel and Oak - Aluminum-Free Deodorant, Deodorant for Men, Essential Oil-Based Scent, 24-Hour Odor Protection, Cedar & Patchouli Blend, Gentle on Sensitive Skin (Mountain Sage, 2.7 oz, 2-Pack)
$15.95
Action: think[B078GWRC1J and B078GTKVXY are bright citrus deodorant less then 50 dollars. I can check B078GWRC1J first.]
Observation: OK.
Action: click[B078GWRC1J]
Observation:
[Back to Search]
[< Prev]
scent [assorted scents][bright citrus][calming lavender][ginger fresh][simply non-scents]
size [travel set (4-pack)][3 ounce (pack of 1)][3-ounce (2-pack)]
Bright Citrus Deodorant by Earth Mama — Natural and Safe for Sensitive Skin, Pregnancy and Breastfeeding, Contains Organic Calendula 3-Ounce
Price: $10.99
Rating: N.A.
[Description]
[Features]
[Reviews]
[Buy Now]
Action: think[For 3 ounce bottle of bright citrus deodorant for sensitive skin, the item has options 'bright citrus' and '3 ounce (pack of 1)' and seems good to buy.]
Observation: OK.
Action: click[bright citrus]
Observation: You have clicked bright citrus.
Action: click[3 ounce (pack of 1)]
Observation: You have clicked 3 ounce (pack of 1).
Action: click[Buy Now]

Figure 7: Prompts used by the act-only (expert) baseline.

Act-only (expert) Prompt
Instruction:
i would like a 3 ounce bottle of bright citrus deodorant for sensitive skin, and price lower than 50.00 dollars
[Search]
Action: search[3 ounce bright citrus deodorant sensitive skin]
Observation:
[Back to Search]
Page 1 (Total results: 50)
[Next >]
[B078GWRC1J]
Bright Citrus Deodorant by Earth Mama — Natural and Safe for Sensitive Skin, Pregnancy and Breastfeeding, Contains Organic Calendula 3-Ounce
$10.99
[B078GTKVXY]
Ginger Fresh Deodorant by Earth Mama — Natural and Safe for Sensitive Skin, Pregnancy and Breastfeeding, Contains Organic Calendula 3-Ounce
$10.99
[B08KBVJ4XN]
Barrel and Oak - Aluminum-Free Deodorant, Deodorant for Men, Essential Oil-Based Scent, 24-Hour Odor Protection, Cedar & Patchouli Blend, Gentle on Sensitive Skin (Mountain Sage, 2.7 oz, 2-Pack)
$15.95
Action: click[B078GWRC1J]
Observation:
[Back to Search]
[< Prev]
scent [assorted scents][bright citrus][calming lavender][ginger fresh][simply non-scents]
size [travel set (4-pack)][3 ounce (pack of 1)][3-ounce (2-pack)]
Bright Citrus Deodorant by Earth Mama — Natural and Safe for Sensitive Skin, Pregnancy and Breastfeeding, Contains Organic Calendula 3-Ounce
Price: $10.99
Rating: N.A.
[Description]
[Features]
[Reviews]
[Buy Now]
Action: click[bright citrus]
Observation: You have clicked bright citrus.
Action: click[3 ounce (pack of 1)]
Observation: You have clicked 3 ounce (pack of 1).
Action: click[Buy Now]

Figure 8: Prompts used by the act-only (expert) baseline.

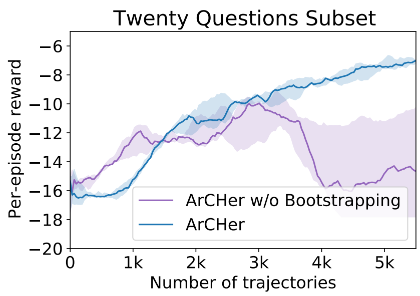
G ADDITIONAL DISCUSSIONS

G.1 TD-LEARNING V.S. MC REGRESSION

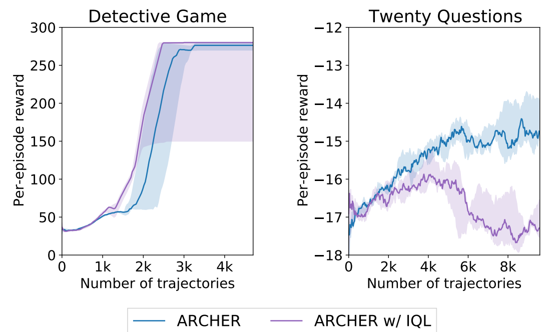
To validate whether TD-learning plays an important role in ArCHer, we carried out an ablation study where we replaced TD-learning in ArCHer with MC regression for critic updates. To make sure that data in the replay buffer are generated by similar policies, we use a smaller replay buffer that contains trajectories collected by three latest policies. The ablation results are shown in Figure 9a. We observed that MC regression may learn faster in the beginning as the information propagates faster than per step TD learning, but it fails to learn reliably over then entire training process. This ablation result shows the importance of TD-learning to effectively make use of off-policy data.

G.2 Online IQL Critic Loss

In our ablation study of using IQL critic loss in the online setting, we set $\tau = 0.9$ to encourage more risk-seeking for better explorations. As we can see from Figure 9b, the use of IQL critic can indeed accelerate explorations in simple tasks such as Detective Game. However, a naive instantiation of IQL fails to provide an unbiased estimate for the policy gradient resulting in potential instabilities in harder tasks such as Twenty Questions.



(a) Ablation study of replacing ArCHer critic update with MC Regression. As data in the replay buffer is generated by different policies, MC regression fails to improve reliably.



(b) Ablation study of replacing ArCHer critic update with an IQL loss. Using an IQL critic loss can sometimes accelerate training but introduce more instability.

H Reward Hacking

To understand whether our agent has really learnt to behave more strategically with reinforcement learning instead of exploiting the environment, we present sample trajectories for Twenty Questions Subset, Twenty Questions, Guess My City, and WebShop in Figure 10, 11, 12, 13. Note that for detective game, there is only a limited number of legal actions and all illegal actions will lead to making no progress in the game, resulting in a lower rewards. For Twenty Questions Subset and Twenty Questions, we observed that the successful dialogues tend to be mostly natural and strategic while some unsuccessful dialogues tend to contain more meaningless repetitions. For Guess My City, we observed that the agent learnt to ask the predominant language of the city which is one of the most informative questions in the SFT distribution we considered in Appendix D. However, as there are some vulnerabilities and hallucinations of the imperfect oracle model for generating free-form answers, some of the conversations might diverge to meaningless repetitions in the end. For WebShop, as the environment forces the format of legal actions for the web server to process, the trajectories tend to be natural and realistic.

Twenty Questions Subset
Questions:
Is it a living thing? No.
Is it a man-made? Yes.
Is it a machine? No.
Is it a tool? No.
Is it a piece of clothing? No.
Is it a piece of furniture? Yes.
Is it a chair? Yes.

Figure 10: Example environment interactions of ArCHer for Twenty Questions Subset at the end of training. It can be seen that in this environment, ArCHer learnt a near-optimal binary search strategy.

Twenty Questions
Questions:
Is it a living thing? Yes.
Is the object a mammal? No.
Is the object a plant? Yes.
Is the object edible? Yes.
Is the object a vegetable? No.
Is the object a fruit? Yes.
Is the object a citrus fruit? No.
Is the object a berry? Yes.
Is the object a strawberry? No.
Is the object a strawberry? No.
Is the object a blueberry? No.
Is the object a rasepeberry? No.
Is the object a currant? No.
Is the object a grape? Yes.

Questions:
Is the object a living thing? No.
Is the object man-made? Yes.
Is the object a vehicle? No.
Is the object a piece of jewelry? No.
Is the object a musical instrument? Yes.
Is the object a stringed instrument? No.
Is the object a wind instrument? Yes.
Is the object a trumpet? No.
Is the object a trombone? No.
Is the object a flute? No.
Is the object a saxophone? No.
Is the object a clarinet? No.
Is the object a clarinet? No.
Is the object a clarinet? No.
Is the object a clarinet? No.
Is the object a clarinet? No.
Is the object a clarinet? No.
Is the object a clarinet? No.
Is the object a clarinet? No.
Is the object a clarinet? No.

Figure 11: Example environment interactions of ArCHer for Twenty Questions (one successful and one unsuccessful trajectory) at the end of training. In this environment, the conversation is smooth and natural for successful trajectories. However, for unsuccessful trajectories, the agent may sometimes keep repeating the same question.

Guess My City
Questions:
What is the predominant language spoken:
The predominant language spoken is Mandarin Chinese.
Is the city you are from Shanghai?
No, I am not from Shanghai.
Is the city you are from Taipei,Is the city you are from TaipeiIs the city you are from Tis aipeiIs the city you are from Tai
Yes

Questions:
What is the predominant language spoken:
English
Is the city you are from London?
No, I am not from London.
Is the city you are from LondonIs the city you are from London Is the city you are from London
Yes, my city is from London.
Is the city you are from LondonIs the city you are from London Is the city you are from London
Yes, my city is from London.
Is the city you are from LondonIs the city you are from London Is the city you are from London
Yes, my city is from London.
Is the city you are from LondonIs the city you are from London Is the city you are from London
Yes, my city is from London.
Is the city you are from LondonIs the city you are from London Is the city you are from London
Yes, my city is from London.
......

Figure 12: Example environment interactions of ArCHer for Guess My City at the end of training. The agent learns to ask the question for the predominant language, which is one of the most informative questions from the offline SFT distribution. However, as the oracle model provides free-form responses can be out of distribution, the agent is more prone to collapsing in out-of-distribution states.

WebShop
Observation:
WebShop
Instruction:
i would like a citrus yao conditioner made with natural ingredients, and price lower than 50.00 dollars
[Search]
Action:
search[citrus yao conditioner natural ingredients]
Observation:
[Back to Search]
Page 1 (Total results: 50)
[Next >]
[B099KWBV49]
VIORI Terrace Garden Conditioner Bar - Handcrafted with Longsheng Rice Water & Natural Ingredients - Sulfate-free, Paraben-free, Cruelty-free, Phthalate-free, pH balanced 100% Vegan, Zero-Waste
$16.25
[B09KQZ9GTK]
VIORI Hidden Waterfall Shampoo & Conditioner Bar Set WITHOUT Bamboo Holder - Handcrafted with Longsheng Rice Water & Natural Ingredients - Sulfate-free, Paraben-free, Phthalate-free, 100% Vegan
$29.25
[B099KWKY36]
VIORI Shampoo Bar, Hidden Waterfall - Handcrafted with Longsheng Rice Water & Natural Ingredients - Sulfate-free, Paraben-free, Cruelty-free, Phthalate-free, pH balanced 100% Vegan, Zero-Waste
$16.25
Action:
click[B099KWBV49]
Observation:
[Back to Search]
[< Prev]
style [bamboo bar holder][citrus yao][hidden waterfall][native essence][terrace garden]
size [bamboo bar holder][bars + bamboo holder set][conditioner][shampoo][shampoo + conditioner set]
VIORI Terrace Garden Conditioner Bar - Handcrafted with Longsheng Rice Water & Natural Ingredients - Sulfate-free, Paraben-free, Cruelty-free, Phthalate-free, pH balanced 100% Vegan, Zero-Waste
Price: $16.25
Rating: N.A.
[Description]
[Features]
[Reviews]
[Attributes]
[Buy Now]
Action:
click[citrus yao]
Observation:
You have clicked citrus yao.
Action:
click[Buy Now]
Observation:
Your score (min 0.0, max 1.0): 0.75

Figure 13: Example environment interactions of ArCHer for WebShop at the end of training. It can be seen that in this environment, the behavior of ArCHer is very natural.

I Hyperparameters

Table 2 shows hyperparameters for ArCHer and other baselines for all environments. As shown in the table, most hyperparameters are held the same across all environment except that a smaller actor learning rate is used for Twenty Questions and a larger number of rollout trajectories is used for Web Shopping. This shows that ArCHer is relatively insensitive to selection of hyperparameters and does not require too much hyperparameter tuning to work in different environments.

Table 2: Hyperparameters for All Experiments

		Detective Game	Twenty Questions Subset	Twenty Questions	Guess My City	Web Shopping
SFT	actor lr	2e-4	2e-4	2e-4	2e-4	2e-4
	batch size	32	32	32	32	32
Filtered BC	actor lr	3e-4	3e-4	3e-5	3e-4	3e-4
	batch size	256	256	256	256	256
	rollout trajectories	32	32	32	32	128
	replay buffer size	10000	10000	10000	10000	10000
	filtering percentage	0.1	0.1	0.1	0.1	0.1
	actor updates per iteration	10	10	10	10	10
CHAI	actor lr	3e-4	3e-4	3e-5	3e-4	3e-4
	critic lr	6e-4	6e-4	6e-4	6e-4	6e-4
	batch size	256	256	256	256	256
	rollout trajectories	32	32	32	32	128
	replay buffer size	10000	10000	10000	10000	10000
	critic updates per iteration	50	50	50	50	50
	discount	0.98	0.95	0.95	0.95	0.9
	polyak alpha	0.9	0.9	0.9	0.9	0.9
ArCHer	actor lr	3e-4	3e-4	3e-5	3e-4	3e-4
	critic lr	6e-4	6e-4	6e-4	6e-4	6e-4
	batch size	256	256	256	256	256
	rollout trajectories	32	32	32	32	128
	replay buffer size	10000	10000	10000	10000	10000
	critic updates per iteration	50	50	50	50	50
	discount	0.98	0.95	0.95	0.95	0.9
	polyak alpha	0.9	0.9	0.9	0.9	0.9
	actor updates per iteration	3	3	3	3	3
	warm up iters with no actor update	10	10	20	10	20
ArCHer w/ Baseline	actor lr	\	\	\	3e-4	\
	critic lr	\	\	\	6e-4	\
	batch size	\	\	\	256	\
	rollout trajectories	\	\	\	32	\
	replay buffer size	\	\	\	10000	\
	critic updates per iteration	\	\	\	50	\
	discount	\	\	\	0.95	\
	actor updates per iteration	\	\	\	3	\
	baseline updates per iteration	\	\	\	60	\
	warm up iters with no actor update	\	\	\	10	\
	polyak alpha	\	\	\	0.9	\
PPO	actor lr	\	6e-6	6e-6	6e-4	\
	batch size	\	1024	1024	1024	\
	rollout trajectories	\	2048	2048	1024	\
	PPO epochs	\	10	20	4	\
	discount	\	0.95	0.95	0.95	\
	GAE lambda	\	0.95	0.95	0.95	\
	clip range	\	0.2	0.2	0.2	\

Table 3: Hyperparameters for ArCHer and baseline methods for all experiments.

A Appendix

You may include other additional sections here.