In-Context Reinforcement Learning From Suboptimal Historical Data

Anonymous authors
Paper under double-blind review

Abstract

Large-scale transformer models have achieved remarkable empirical successes, largely due to their in-context learning capabilities. Inspired by this, we explore training an autoregressive transformer for in-context Reinforcement Learning (RL). In this setting, we initially train a transformer on an offline dataset consisting of trajectories collected from various RL instances, and then fix and use this transformer to create an action policy for new RL instances. Notably, we consider the setting where the offline dataset contains trajectories sampled from suboptimal behavioral policies. In this case, standard autoregressive training corresponds to imitation learning and results in suboptimal performance. To address this, we propose the Decision Importance Transformer (DIT), which emulates the actor-critic algorithm in an in-context manner. In particular, we first train a transformer-based value function that estimates the advantage functions of the behavior policies that collected the suboptimal trajectories. Then we train a transformer-based policy via a weighted maximum likelihood estimation loss, where the weights are constructed based on the trained value function to steer the suboptimal policies to the optimal ones. We conduct extensive experiments to test the performance of DIT on both bandit and Markov Decision Process problems. Our results show that DIT achieves superior performance, particularly when the offline dataset contains suboptimal historical data.

1 Introduction

Large-scale transformer models (LTMs) such as Large Language Models have achieved remarkable empirical successes (Radford et al., 2019; OpenAI et al., 2024). In particular, LTMs trained on vast amount of data have shown remarkable *in-context learning* (ICL) capabilities in supervised learning, effectively *solving new tasks* with just a few demonstrations and *without requiring any parameter updates* (Brown et al., 2020a; Akyürek et al., 2022). Meanwhile, substantial evidence demonstrates that autoregressive LTMs excel at solving ***individual*** Reinforcement Learning (RL) tasks where an LTM-based agent is trained and tested on the ***same*** task (Li et al., 2023b).

In-context RL. Inspired by these, recent research has explored the use of LTMs for *in-context RL* (Laskin et al., 2022; Lee et al., 2024). In this setting, we *pretrain LTMs on an offline dataset consisting of trajectories collected from various RL instances*. After pretraining, we *deploy LTMs to new and unseen RL instances*. When presented with the *context* containing history of environment interactions collected by unknown and often suboptimal policies, pretrained LTMs predict the optimal actions for current states from the environmental information provided in the context. See Figure 1 for a visual illustration. Two recent works, *Algorithm Distillation* (AD) (Laskin et al., 2022) and *Decision Pretrained Transformer* (DPT) (Lee et al., 2024), have demonstrated impressive in-context RL abilities, inferring near-optimal policies for new RL instances.

Challenges. However, existing approaches focus on training LTMs to imitate the actions in the pretraining datasets and thus have *stringent requirements on the pretraining datasets*. For example, DPT *requires access to optimal policies* to generate a set of optimal action labels for its supervised pretraining of LTMs. To overcome these limitations, this work considers training LTMs for in-context RL using only suboptimal historical data. While this presents significant challenges, it also offers substantial potential benefits by *significantly improving the feasibility of in-context RL*, as suboptimal trajectories are far easier to gather. For instance, large companies often maintain extensive databases of historical trajectories from non-expert users.

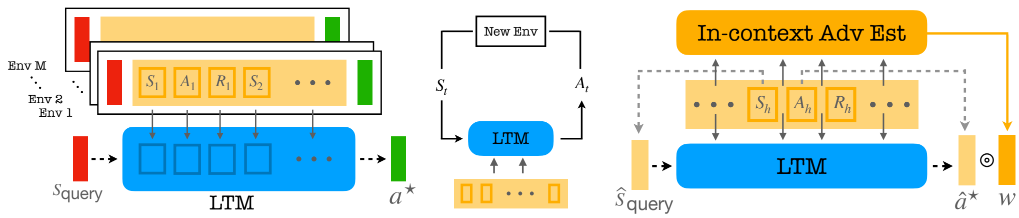


Figure 1: **Supervised Pretraining (left)**: Presented with offline trajectories and *optimal action labels*, LTMs are pretrained to predict the optimal actions for query states across RL tasks. **In-Context RL (middle)**: When deployed to unseen environments, the pretrained LTMs generate actions conditioned on the current states and offline trajectories collected by (suboptimal) behavioral policies. **Pretraining with Suboptimal Historical Data (right):** Lack of the optimal action labels, the proposed framework employs *in-trajectory* state-action pairs as query states and *pseudo*-optimal action labels, and a *weighted pretraining objective*, where the weights are based on the optimality of actions, estimated by an LTM-based *in-context advantage function estimator*.

Contributions. In pursuit of this goal, we introduce the ***Decision Importance Transformer*** (DIT), a supervised pretraining framework for in-context RL *using only historical trajectories collected by suboptimal behavioral policies across distinct RL instances*. When the pretraining datasets contain only suboptimal trajectories, existing approaches correspond to imitation learning and thus result in suboptimal performance. DIT overcomes this challenge through several techniques:

- DIT learns to infer near-optimal actions from suboptimal trajectories through an *exponential reweighting* technique that assigns *good actions* in the offline dataset with *more weights* during supervised pretraining. These assigned weights guide the suboptimal policies toward the optimal ones.
- In particular, the assigned weights are constructed from the *advantage functions* of the behavior policies such that actions with high advantage values receive more weights during pretraining, leading to *guaranteed policy improvements* over the behavior policies.
- Notably, although advantage weighted regression has been studied in standard RL, *it remains unclear how to generalize this approach to ICRL*. The main reason is that the weighting function needs to be task dependent for ICRL. Thus, it is necessary to estimation the advantages functions for all RL tasks in the pretraining dataset. The most severe technical challenge for this is that because the source tasks of training trajectories are unknown and thus we cannot combine the trajectories from the same RL tasks to improve estimation , *we need to estimate the advantage functions individually for each trajectory in the pretraining dataset*. To address this formidable challenge, DIT trains an *LTM-based advantage estimator* that interpolates across tasks for an *in-context estimation of the advantage functions* to facilitate the weighted supervised pretraining. See Figure 1 for a visualization.

Empirical Results. Through extensive experiments on various bandit and Markov Decision Process (MDP) problems, we demonstrate that *pretrained DIT models generalize to unseen decision-making problems*. On bandit problems, the performance of DIT models matches that of the theoretically optimal bandit algorithms (e.g., Thompson Sampling (Russo et al., 2018)). In four challenging MDP problems including two navigating tasks with sparse rewards (Dark Room (Laskin et al., 2022) and Miniworld (Chevalier-Boisvert et al., 2023)) and two complex continuous control tasks (Meta-World (Yu et al., 2020) and Half-Cheetah (Todorov et al., 2012)), DIT models achieves superior performance, particularly when the pretraining dataset contains suboptimal trajectories. Notably, in most scenarios, *DIT is comparable to DPT in both online and offline testings, despite being pretrained without optimal action labels*.

2 RELATED WORK

Offline Reinforcement Learning. Since we consider pretraining with historical data, our work falls within the broader field of offline RL. While online RL algorithms (Kaelbling et al., 1996; François-Lavet et al., 2018) learn optimal policies by interacting with the environments through trial and error, offline RL (Levine et al., 2020; Matsushima et al., 2020; Prudencio et al., 2023) aims to infer optimal policies from historical data collected by (suboptimal) behavioral policies. One of the substantial challenges for offline RL is the distribution shift caused by the mismatch between behavioral policies and optimal policies (Levine et al., 2020; Kostrikov et al., 2021). To this end, offline RL algorithms learn pessimistically by either policy regularization or underestimating the policy returns (Wu et al., 2019; Kidambi et al., 2020; Kumar et al., 2020; Rashidinejad et al., 2021; Yin & Wang, 2021; Jin et al., 2021; Dong et al., 2023; Fujimoto & Gu, 2021). While the goal of offline RL is solve the *same* RL tasks from where the offline datasets are collected, the goal of in-context RL is to efficiently generalize to *unseen* tasks after pretraining with offline datasets from diverse RL tasks.

Transformer Models and Autoregressive Decision Making. Large Language Models and autoregressive models (Radford et al., 2019; Brown et al., 2020b; Wu et al., 2023b; Touvron et al., 2023; OpenAI et al., 2024) have achieved astonishing empirical successes in a wide range of application areas, including medicine (Singhal et al., 2023; Thirunavukarasu et al., 2023), education (Kasneci et al., 2023), finance (Wu et al., 2023a; Yang et al., 2023), etc. As it is natural to use autoregressive models for sequential decision making, transformer models have demonstrated superior performance in both bandit and MDP problems (Li et al., 2023a; Yuan et al., 2023). In particular, Decision Transformer (DT) (Chen et al., 2021; Zheng et al., 2022; Liu et al., 2023; Yamagata et al., 2023) uses return-conditioned supervised learning to tackle offline RL. Although salable to multi-task settings (i.e., one model for multiple RL problems), DT is commonly criticised for its inability to improve upon the offline datasets and provably sub-optimal in certain scenarios, e.g., environment with high stochasticity (Brandfonbrener et al., 2022; Yang et al., 2022; Yamagata et al., 2023). More importantly, DT cannot generalize to unseen RL problems in context. To this end, Algorithm Distillation (AD) (Laskin et al., 2022) uses sequential modeling to emulate the learning process of RL algorithms, i.e., meta-learning (Vilalta & Drissi, 2002). The work most closely related to ours is the Decision Pretrained Transformer (DPT) (Lee et al., 2024), a supervised pretraining approach for in-context decision making. DPT trains transformers to predict the optimal action given a query state and a set of transitions. As delineated in Section 1, AD and DPT have stringent assumptions on the pretraining datasets. Our work overcomes those drawbacks and does not require query to optimal policies nor the complete learning histories of RL algorithms (Laskin et al., 2022; Lee et al., 2024).

3 PRELIMINARY

Markov Decision Process. Sequential decision problems can be formulated as Markov Decision Processes (MDPs). An MDP τ is described by the tuple $(\mathcal{S}, \mathcal{A}, P_\tau, R_\tau, \gamma, \rho_\tau)$ where $\mathcal{S}$ is the set of all possible states, $\mathcal{A}$ is the set of all possible actions, $P_\tau : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the dynamic function that describes the distribution of the next state given the current state and action, $R_\tau : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $\gamma \in (0, 1)$ is the discounting factor for cumulative rewards, and $\rho_\tau \in \Delta(\mathcal{S})$ is the initial state distribution. An agent (decision maker) interacts with the environment as follows. At the initial step $h = 1$, an initial state $s_1 \in \mathcal{S}$ is sampled according to ρ_τ. At each time step h, the agent chooses action $a_h \in \mathcal{A}$ and receives reward $r_h = R_\tau(s_h, a_h)$. Then the next state s_{h+1} is generated following the dynamic $P_\tau(s_h, a_h)$. A policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ maps the current state to an action distribution. Let $G_\tau(\pi) = \mathbb{E}[\sum_{h=1}^{\infty} \gamma^{h-1} r_h | \pi, \tau]$ denote the expected cumulative reward of π for task τ. The goal of an agent is to learn the optimal policy $\pi_\tau^\star$ that maximizes $G_\tau(\pi)$.

Decision-Pretrained Transformer. Our proposed approach builds upon the model architecture of DPT, which is a supervised pretraining method for transformer models to have in-context RL capabilities (see Figure 1 for its architecture). DPT assumes a set of tasks $\{\tau^i\}_{i=1}^m$ sampled independently from a task distribution p_τ. Here each τ^i is an instance of MDP. For each task τ^i, a context dataset D^i is sampled, consisting of interactions between a behavioral policy and τ^i. That is, $D^i = \{(s_h^i, a_h^i, s_{h+1}^i, r_h^i)\}_h$, where a_h^i is chosen by a behavioral policy. Additionally, for each task τ^i, a query state $s_{\text{query}}^i \in \mathcal{S}$ is sampled, and an associated optimal action label $a_i^\star$ is sampled from $\pi_{\tau^i}^\star(s_{\text{query}})$, where $\pi_{\tau^i}^\star$ is the optimal policy for τ^i. The complete pretraining dataset is

$\mathcal{D}_{pre} = \{D^i, s^i_{\text{query}}, a^\star_i\}_{i=1}^m$. Let T_θ denote a causal GPT-2 transformer with parameters θ (Radford et al., 2019). The pretraining objective of DPT is defined as

$$\min_\theta \frac{1}{m}\sum_{i=1}^{m} -\log T_\theta\left(a^\star_i | s^i_{\text{query}}, D^i\right). \tag{1}$$

In-context RL. After pretraining, the pretrained autoregressive LTM T_θ can be deployed as both an online and offline agent. During deployment (testing), an unseen testing task τ is sampled from p_τ. For offline deployment, a dataset D_{off} is first sampled from τ, e.g., D_{off} contains trajectories gathered from a random policy in τ, then DPT follows the policy $T_\theta(\cdot|s_h, D_{\text{off}})$ after observing the state s_h at time step h . For online deployment, DPT initiates with an empty dataset D_{on}. In each episode, DPT follows the policy $T_\theta(\cdot|s_h, D_{\text{on}})$ to collect a trajectory $\{s_1, a_1, r_1, \ldots, s_H, a_H, r_H\}$ which will be appended into D_{on}. This process repeats for a pre-defined number of episodes. See Algorithm 2 (in Appendix) for pseudocodes on deployment.

4 Decision Importance Transformer

Here we introduce our proposed framework *Decision Importance Transformer* (DIT).

Pretraining with Suboptimal Data. Similar to DPT, DIT assumes a family of datasets $\mathcal{D} = \{D^i\}_{i=1}^m$ where D^i consists of H transitions $\{(s^i_h, a^i_h, s^i_{h+1}, r^i_h)\}_{h=1}^H$ collected by the (suboptimal) behavioral policy $\pi^b_{\tau^i}$ in the RL instance τ^i which itself is independently sampled from the task distribution p_τ. In contrast to DPT, however, *DIT does not require the set of paired query states and optimal action labels* $\{s^i_{\text{query}}, a^\star_i\}_{i=1}^m$ across distinct environments, which is often difficult to obtain in practice.

Notations. In the sequel, for any task τ, we assume that it has an index (parameter) also denoted by τ such that the task information τ can be an explicit input to a meta-policy $\pi(s|a;\tau)$ which can generate distinct policies based on the received task τ. For example, in robotic control tasks, τ may represent the physical parameters of the robots such as *robot mass* or the environmental parameters such as *ground friction*. We use $\pi^b_\tau(a|s)$ to denote the ***behavioral policy*** for task τ. Denote

$$V^b_\tau(s) = \mathbb{E}\left[\sum_{h=1}^{\infty}\gamma^{h-1} r_h \big| s_1 = s, \tau, \pi^b_\tau\right], \quad Q^b_\tau(s,a) = \mathbb{E}\left[\sum_{h=1}^{\infty}\gamma^{h-1} r_h \big| s_1 = s, a_1 = a, \tau, \pi^b_\tau\right]$$

as its value and action-value functions respectively, and let

$$A^b_\tau(s,a) = Q^b_\tau(s,a) - V^b_\tau(s) \tag{2}$$

be its ***advantage function***.

For presentation clarity, in Section 4.1, we first consider the scenarios where (i) $A^b_\tau(s,a)$ is known and (ii) the task index τ is also known and can be provided as input to a policy. Then in Section 4.2, we introduce solutions for scenarios where $A^b_\tau(s,a)$ and τ need to be estimated. All proofs of the theoretical results in this section are deferred to the Appendix B.

4.1 Weighted Maximum Likelihood Estimation

Motivation. To motivate DIT, we first consider the setting of imitation learning where the agent is trained and tested on the same task. Given a dataset of transitions $D = \{(s_h, a_h, s_{h+1}, r_h)\}$ collected by a behavior policy $\pi^b(a|s)$, Wang et al. (2018) propose to optimize a weighted objective:

$$\pi \in \arg\max_\pi \sum_{(s_h,a_h)\in D} \exp(A^b(s_h,a_h)) \cdot \log \pi(a_h|s_h).$$

The rationale is that *the good actions in the offline dataset*, that is, a_h with high advantage value $A^b(s_h,a_h)$, *should be given more weights during imitation learning*. These weights essentially work as importance sampling ratios so that the action distribution is closer to the optimal one.

Weighted Pretraining for In-context RL. In contrast to imitation learning that focuses on *individual* RL tasks, the objective of DIT is to learn a ***task-conditioned policy*** $\pi(a|s;\tau)$ with the task index τ as input. In particular, $\pi(a|s;\tau)$ should perform well for $\tau \sim p_\tau$.

Motivated by the aforementioned weighted imitation learning objective, DIT has the following ***weighted maximum likelihood estimation*** (WMLE) loss for supervised pretraining:

$$\min_{\pi} L(\pi) = -\mathbb{E}_{\tau\sim p_\tau, s\sim d_\tau(s), a\sim \pi^b_\tau(a|s)}\left[\exp\left(A^b_\tau\left(s,a\right)/\eta\right)\cdot \log \pi(a|s;\tau)\right], \quad (3)$$

where $d_\tau(s)$ is the discounted visiting frequencies of $\pi^b_\tau(a|s)$ defined as $d_\tau(s) = (1-\gamma)\mathbb{E}\left[\sum_{h=1}^{\infty}\gamma^{h-1}\mathbb{1}\{s_h = s\}|\tau, \pi^b_\tau\right]$. The effectiveness of the objective in Equation (3) is demonstrated by the following result which states that *the optimizer to DIT's pretraining objective is also the solution to another policy optimization problem* that is easier to interpret.

Proposition 4.1. *Consider the following optimization problem:*

$$\max_{\pi} J(\pi) = \mathbb{E}_{\tau\sim p_\tau, s\sim d_\tau(s), a\sim \pi^b_\tau(a|s)}\Bigg[\underbrace{A^b_\tau\left(s,a\right)}_{(I)} - \eta\cdot \underbrace{D_{\mathrm{KL}}(\pi(\cdot|s;\tau)\|\pi^b_\tau(\cdot|s))}_{(II)}\Bigg], \quad (4)$$

where D_{KL} is the Kullback–Leibler (KL) divergence, and let $\pi^\star \in \arg\max_\pi J(\pi)$ be its optimizer. Then we have for any policy $\pi(a|s;\tau)$,

$$\begin{aligned}&\mathbb{E}_{\tau\sim p(\tau), s\sim d_\tau(s)}\left[D_{\mathrm{KL}}\left(\pi^\star(\cdot|s;\tau)\|\pi(\cdot|s;\tau)\right)\right]\\ &= -\mathbb{E}_{p(\tau), s\sim d_\tau(s), a\sim \pi^b_\tau(a|s)}\left[\frac{1}{Z_\tau(s)}\exp\left(A^b_\tau(s,a)/\eta\right)\cdot\log\pi(a|s;\tau)\right] + C,\end{aligned} \quad (5)$$

where C is a constant independent of π and $Z_\tau(s) = \sum_a \pi^b_\tau(a|s)\exp(A^b_\tau\left(s,a\right)/\eta)$.

In Equation (4), the objective is to find *a policy $\pi^\star$ that improves over the behavior policy* (by maximizing term (I)) *and does not stray too far from the behavior policy* (by minimizing term (II)). When the behavioral policy $\pi^b_\tau(a|s)$ is near-optimal, η should set to a large value so that we can have *safe* improvements over the behavioral policy. On the other hand, when the behavioral policy is highly sub-optimal, η should set to a small value so that we have more freedom for policy improvement to decrease the sub-optimality. Note that the D_{KL} constraint (term (II) in Equation (4)) is critical for pretraining large transformer models to prevent policy collapse (Schulman, 2015).

Comparing Equation (5) with the pretraining objective of DIT in Equation (3), we observe that *DIT aims to identify a policy that is closest to $\pi^\star$* by setting $Z_\tau(s) = 1$ (we provide a brief discussion for why this approach is valid in Appendix B.3). When $A^b_\tau(s,a)$ is known, the pretraining objective of DIT can be estimated with the given pretraining dataset $\mathcal{D}$ by

$$\min_{\pi} L_n(\theta) := -\frac{1}{mH}\sum_{i=1}^{m}\sum_{h=1}^{H}\exp(A^b_{\tau^i}(s^i_h, a^i_h)/\eta)\log\pi\left(a^i_h|s^i_h;\tau^i\right). \quad (6)$$

Next we establish that *DIT can provably achieve policy improvement*.

Proposition 4.2 (Policy Improvement)**.** *Let $\pi^\star$ be the policy that optimizes (4). For any task τ and policy π, let $G_\tau(\pi) = \mathbb{E}[\sum_{h=0}^{\infty}\gamma^h r_h|\pi,\tau]$ represent the expected cumulative reward of π for τ. Let $\pi^\star_\tau$ denote $\pi^\star(a|s;\tau)$. Then we have*

$$\mathbb{E}_{\tau\sim p_\tau}[G_\tau(\pi^\star_\tau) - G_\tau(\pi^b_\tau)] \geq \frac{\eta}{1-\gamma}\mathbb{E}_{\tau\sim p_\tau}[C^D_\tau] - \frac{2\gamma}{(1-\gamma)^2}\mathbb{E}_{\tau\sim p_\tau}\left[C^A_\tau\cdot\sqrt{C^D_\tau/2}\right], \quad (7)$$

where $C^D_\tau = \mathbb{E}_{s\sim d_\tau(s)}[D_{\mathrm{KL}}(\pi^\star(\cdot|s;\tau)\|\pi^b_\tau(\cdot|s))]$ and $C^A_\tau = \max_s|\mathbb{E}_{a\sim\pi^\star(a|s;\tau)}A^b_\tau(s,a)|$.

In particular, when the magnitude of the advantage function A^b_τ is small, the right-hand side of Equation (7) is nonnegative. In this case, the policy $\pi^\star$ obtained by solving Equation (4) is strictly better than the behavior policy. Equivalently, adding the exponential weights in Equation (6) is strictly better than vanilla imitation learning, when the total number of pretraining tasks m is large.

4.2 In-context Task Identification and Advantage Function Estimation

However, two key challenges remain:

1. The *task index τ is not accessible during testing* as only a context dataset D_τ is presented. In other words, we have no knowledge about the true identity of the testing task τ.

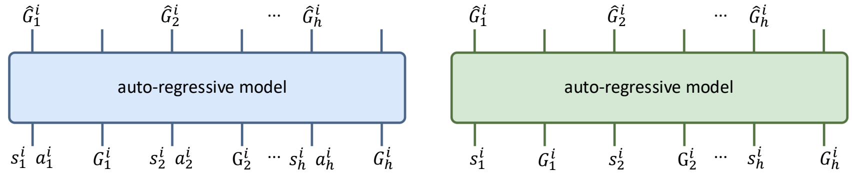


Figure 2: Model structure of the in-context action-value transformer $\widehat{Q}$ (left) and value transformer $\widehat{V}$ (right) on the trajectory of the i-th pretraining task.

2. The *advantage function* $A^b_\tau(s,a)$ *is not accessible* for pretraining.

In-context Task Identification. We follow DPT to instantiate $\pi(a|s;\tau)$ with an autoregressive transformer T_θ parameterized by θ. Conditioned on a given context dataset D_τ consisting of environment interactions collected by a behavioral policy in τ, *the LTM-based policy* $T_\theta(a|s, D_\tau)$ *first implicitly extracts task information about* τ *from the context* D_τ *and chooses an action based on the extracted task information* (see Lee et al. (2024) for a detailed discussion). During pretraining, T_θ learns to extract useful task information for the pretraining tasks $\{\tau^i\}_{i=1}^m$ conditioned on the pretraining context datasets $\{D^i\}_{i=1}^m$, and generalizes to unseen tasks during testing.

In-context Advantage Function Estimation. The second problem is more critical. Given that during pretraining the context dataset D^i may contain up to several trajectories for each task τ^i in the setting of in-context RL, *estimation of* $A^b_{\tau^i}(s,a)$ *based on* D^i *alone can be unreliable*.

To this end, in the same spirit of in-context RL, we propose to use an ***in-context advantage function estimator*** to estimate the advantage of any state-action pair (s^i_h, a^i_h) in the pretraining dataset $\mathcal{D}$ by

$$\widehat{A}_b(s^i_h, a^i_h|\tau^i) = \widehat{Q}_\zeta(s^i_h, a^i_h|D_Q^{i,h}) - \widehat{V}_\phi(s^i_h|D_V^{i,h}), \tag{8}$$

with two transformers $\widehat{V}_\phi$ and $\widehat{Q}_\zeta$, parameterized by ϕ and ζ, as ***in-context value/action value estimators*** that interpolate across tasks to have an improved estimation.

Model Architecture. Specifically, let $G^i_h = \sum_{h'=h}^{H} \gamma^{h'-h} r^i_h$ be the in-trajectory discounted cumulative reward. For any observed state-action pair (s^i_h, a^i_h) in the pretraining dataset, $\widehat{Q}_\zeta(s^i_h, a^i_h|D_Q^{i,h})$ and $\widehat{V}_\phi(s^i_h|D_V^{i,h})$ estimate the action-value function $Q^b_{\tau^i}(s^i_h, a^i_h)$ and value function $V^b_{\tau^i}(s^i_h)$ respectively, conditioned on the histories of transitions $D_Q^{i,h} = \{(s^i_j, a^i_j, G^i_j)\}_{j=1}^{h-1}$ and $D_V^{i,h} = \{(s^i_j, G^i_j)\}_{j=1}^{h-1}$, where we employ $\{G^i_j\}_{j<h}$ as the noisy labels for value functions to facilitate in-context learning. See Figure 2 for a visual representation of the model architecture.

Training. We train $\widehat{V}_\phi$ and $\widehat{Q}_\zeta$ with the following objective function:

$$\min_{\phi,\zeta} L_A(\phi,\zeta) := L_{\text{reg}}(\phi,\zeta) + \lambda \cdot \left(L_V^B(\phi) + L_Q^B(\zeta)\right), \tag{9}$$

where $\lambda > 0$ is a hyperparameter to balance

$$L_{\text{reg}}(\phi,\zeta) := \frac{1}{mH}\sum_{i=1}^{m}\sum_{h=1}^{H}\left(\widehat{V}_\phi(s^i_h|D_V^{i,h}) - G^i_h\right)^2 + \left(\widehat{Q}_\zeta(s^i_h, a^i_h|D_Q^{i,h}) - G^i_h\right)^2,$$

$$L_Q^B(\zeta) := \frac{1}{mH}\sum_{i=1}^{m}\sum_{h=1}^{H-1}\left(r^i_h + \gamma\widehat{Q}_\zeta(s^i_h, a^i_h|D_Q^{i,h}) - \widehat{Q}_\zeta(s^i_{h+1}, a^i_{h+1}|D_Q^{i,h+1})\right)^2 \text{ and}$$

$$L_V^B(\phi) := \frac{1}{mH}\sum_{i=1}^{m}\sum_{h=1}^{H-1}\left(r^i_h + \gamma\widehat{V}_\phi(s^i_h|D_V^{i,h}) - \widehat{V}_\phi(s^i_{h+1}|D_V^{i,h+1})\right)^2.$$

Here, L_Q^B and L_V^B regularize the transformer models with the Bellman equations for value functions.

DIT with In-context Advantage Estimator. After training, with $\widehat{A}_b(s^i_h, a^i_h|\tau^i)$ defined in Equation (8) as an estimation of the true advantage function, we can now optimize the objective function of DIT to have the pretrained LTM $T_{\theta^\star}$ for in-context RL, i.e.,

$$\theta^\star \in \arg\min_{\theta\in\Theta} \widehat{L}_n(\theta) := -\frac{1}{mH}\sum_{i=1}^{m}\sum_{h=1}^{H} \exp(\widehat{A}_b(s_h^i, a_h^i|\tau^i)/\eta) \log T_\theta\left(a_h^i|s_h^i, D^i\right). \tag{10}$$

We summarize the complete procedure of DIT in Algorithm 1 (in Appendix).

5 Experiments

We empirically demonstrate the efficacy of DIT through experiments on various bandit and MDP problems. In bandit problems, *DIT showcases matching performance to that of the theoretically optimal bandit algorithms* in both online and offline settings. In MDP problems, we corroborate that DIT can infer close-to-optimal policies from suboptimal pretraining datasets. Notably, *albeit without optimal action labels during pretraining, DIT models demonstrate performance as strong as that of DPT*, which has access to optimal action labels during pretraining.

Implementation. We follow Lee et al. (2024) to choose GPT-2 (Radford et al., 2019) as the backbone for T_θ, $\widehat{Q}$, and $\widehat{V}$ due to limited computation resource, and note that the performance may be further improved with larger models. Because all tasks have fairly short horizons (all less than 200), we set $\gamma = 0.8$ for all tasks. We choose $\eta = 1$ for all tasks. Due to space constraint, see Appendix F for more details.

5.1 Bandit Problems

We consider linear bandit (LB) problems with an underlying structure shared among tasks. Specifically, there exists a bandit feature function $\phi : \mathcal{A} \to \mathbb{R}^d$ that is *fixed* across tasks where d denotes the dimension of linear bandit problems. The reward of a bandit $a \in \mathcal{A}$ in task τ^i is $r^i(a) \sim \mathcal{N}\left(\mu_a^i, \sigma^2\right)$ where $\mu_a^i = \mathbb{E}[r|a, \tau^i] = \langle\theta^i, \phi(a)\rangle$ and $\sigma^2 = 0.3$. Here, θ^i is the task-specific parameter that defines task τ^i. We conduct experiments on LB problems where $K = 20$, $d = 10$ and $H = 200$. The pretraining dataset for DIT are generated as follows.

Pretraining Dataset. For LB problems, we generate the feature function $\phi : \mathcal{A} \to \mathbb{R}^d$ by sampling bandit features from independent Gaussian distributions, i.e., $\phi(a) \sim \mathcal{N}_d\left(0, I_d/d\right)$ for all $a \in \mathcal{A}$. To generate the pretraining tasks $\{\tau^i\}$, we sample their parameters $\{\theta^i\}$ independently following $\theta^i \sim \mathcal{N}_d\left(0, I_d/d\right)$. To generate context dataset D^i, we randomly generate a behavioral policy by mixing (i) a probability distribution samples a Dirichlet distribution and (ii) a point-mass distribution on one random arm. The mixing weights are uniform sampled from $\{0.0, 0.1, \ldots, 1.0\}$. At every time step h, the behavioral policy samples an action a_h^i and receives r_h^i. *We do not enforce extra coverage of the optimal actions for bandit problems*. Following the setting of DPT (Lee et al., 2024), we collect 100k context datasets for LB problems.

Comparisons. We compare to the following baselines (see Appendix A for more details): ***Empirical Mean (EMP)*** selects the bandit with the highest average reward; ***Upper Confidence Bound (UCB)*** (Auer, 2002) builds upper confidence bounds for all bandits and selects the bandit with the highest upper bound; ***Lower Confidence Bound (LCB)*** (Xiao et al., 2021) builds lower confidence bounds for all bandits and selects the bandit with the highest lower bound; ***Thompson Sampling (TS)*** (Russo et al., 2018) builds a posterior distribution for the rewards of all bandits. At each step, TS samples means for all bandits from the posterior distribution and selects the bandit with the highest sampled mean. In terms of metrics, for offline learning, we follow the convention to use the *suboptimality* defined as $(\mu_{a^\star} - \mu_{\hat{a}})$ where $\mu_{a^\star}$ is the mean reward of the optimal bandit and $\mu_{\hat{a}}$ is the mean reward of the chosen bandit; for online learning we use the *cumulative regret* defined as $\sum_h(\mu_{a^\star} - \mu_{a_h})$ where a_h is the chosen action at time step h.

Empirical Results. As can be seen in Figure 3, in the online setting, though pretrained without the optimal action labels, DIT models demonstrate superior performance to those of the theoretically optimal bandit algorithms, i.e., UCB and TS. Deployed for unseen bandit problems, DIT models quickly identify the optimal bandits at the beginning and maintain low regrets over the horizon. In the offline setting, DIT models can infer near-optimal bandits from trajectories collected by suboptimal policies. In particular, when the behavioral policies (captioned as *BEH* in Figure 3) are

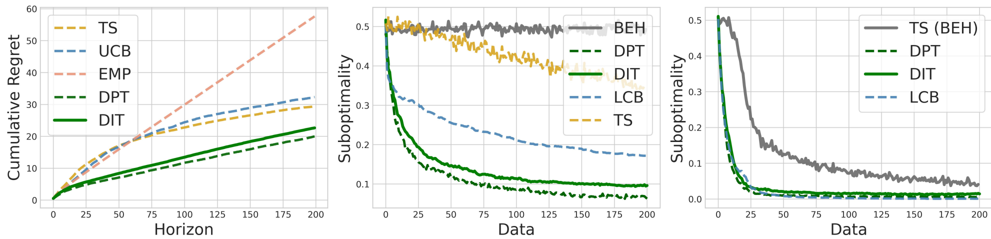


Figure 3: Results for Linear Bandits (lower values indicate better performance). **Left**: Online testing. **Middle**: Offline testing conditioned on trajectories gathered by highly suboptimal, randomly generated policies. **Right:** Offline testing condtioned on trajectories gathered by experts.

randomly generated policies, *DIT significantly outperforms both TS and LCB*, the theoretically optimal algorithm for offline bandit problems. When the context is collected by expert policies, *DIT models improve upon their performance*, achieving lower regrets through in-context decisions.

5.2 MDP Problems

Environments. We conduct experiments on four challenging MDP environments: *two navigating tasks with sparse reward* Dark Room Laskin et al. (2022) and Miniworld Chevalier-Boisvert et al. (2023), as well as *two complex continuous-control tasks* Meta-world and Half-Cheetah. In ***Dark Room***, the agent is randomly placed in a room of 10×10 grids, and there is an *unknown* goal location on one of the grid. The agent needs to move to the goal location by choosing from 5 actions in 100 steps. In ***Miniworld***, the agent is placed in a room and receives a $(25 \times 25 \times 3)$ color image and its direction as input. It can choose from four possible actions to reach a target box, out of four boxes of different colors. In ***Meta-World***, the task is to control a robot hand to reach a target position in 3D space. In ***Half-Cheetah***, the agent controls a robot to reach a target velocity, which is uniformly sampled from the interval $[0, 3]$, and is penalized based on how far its current velocity is from the target velocity. See Appendix C for details of these environments.

Pretraining Datasets. For Dark Room and Miniworld, to ensure coverage of optimal actions (so that optimal policies can be inferred), at every step, with probability p (respectively $1 - p$) we use optimal policy (respectively random policy) to choose action. We choose p so that the average reward of the trajectories in the pretraining dataset is less than 30% of that of the optimal trajectories. For Meta-World and Half-Cheetah, we construct the pretraining datasets using historical trajectories generated by agents trained with *Soft Actor Critic* (SAC). Specifically, SAC is trained until convergence for each task, then we sample from its learning trajectories to build the dataset. Our SAC model training follows the settings outlined in Haarnoja et al. (2018). See Appendix D for details regarding the pretraining dataset.

Comparisons. We compare **DIT** to other in-context algorithms as well as RL algorithms that train an agent from scratch without pretraining. The baseline algorithms are briefly described next (see their implementation details in Appendix A).

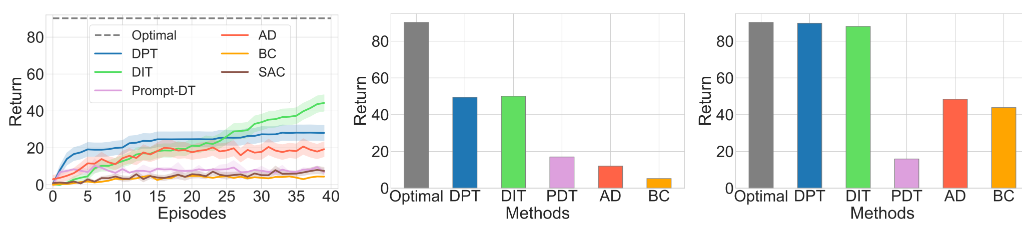
- **Soft Actor Critic (SAC)** Haarnoja et al. (2018): SAC is an online RL algorithm that trains an agent from scratch in every environment.
- **Algorithm Distillation (AD)**: AD is a sequence modeling-based approach for in-context RL that emulates the learning process of RL algorithms (Laskin et al., 2022). To this end, *AD requires the pretraining dataset to consist of complete learning histories of an RL algorithm* —from episodes generated by randomly initialized policies to those collected by nearly optimal policies— *across a wide range of RL tasks*. In this work we use SAC as the RL algorithm for AD to emulate.
- **Decision Pretrained Transformer (DPT)**: DPT and DIT use the same context datasets for pretraining. However, *DPT requires query states and their associated optimal action labels* across different tasks. We follow the original setting of DPT to uniformly sample query state from all possible states and obtain an associated optimal action label.

- **Prompt-DT (PDT)**: PDT is a Decision Transformer-based approach for in-context RL (Xu et al., 2022). PDT leverages the transformer's sequential modeling and prompt framework for few-shot adaptation. PDT uses the same pretraining dataset as DIT. Thus, *the performance gain of DIT over PDT highlights the effectiveness of DIT's design*.
- **Behavior Cloning (BC)**: To investigate the effectiveness of the proposed reweighting technique, we include *an variation of DIT without the exponential reweighting*. This approach closely imitates BC (thus we name it BC), with the following pretraining objective:

$$\min_{\theta} \frac{1}{mH} \sum_{i=1}^{m} \sum_{h=1}^{H} -\log T_{\theta} \left(a_h^i | s_h^i, D^i \right).$$

In particular, **AD** and **DPT** *require extra information during pretraining*: AD requires the complete learning history of RL algorithms while DPT requires optimal action labels. Given that *DIT only relies on suboptimal historical data*, the comparison is *inherently unfair*. Notably, despite these disadvantages, *DIT outperforms AD and matches with DPT in most scenarios*. In terms of metrics, we follow the convention to use the *episode cumulative return* $\sum_{h=1}^{H} r_h$.

In-context Decision-making for Navigating Tasks. We explore how our method generalizes to unseen RL tasks, using the Dark Room environment (Laskin et al., 2022). Following the evaluation protocol of DPT (Lee et al., 2024), we use 80 goals for training and evaluate on the remaining 20 unseen goals. For SAC, since it is an online learning method, we directly train from scratch on the 20 goals to benchmark the returns of in-context RL. Figure 4a shows the online evaluation over 40 episodes. After 40 episodes, SAC gains little in return, demonstrating the difficulty of the RL tasks for testing. Restricted by their capability to efficiently explore in new tasks, BC also perform poorly. *Although our method (DIT) initially has lower returns than DPT and AD, it quickly surpasses them and continues to improve.* Figures 4b and 4c show the results for offline evaluations with expert (high-reward) trajectories and random (low-reward) trajectories. *Despite being pretrained without the optimal action labels, DIT models demonstrate competitive performance to that of DPT*.

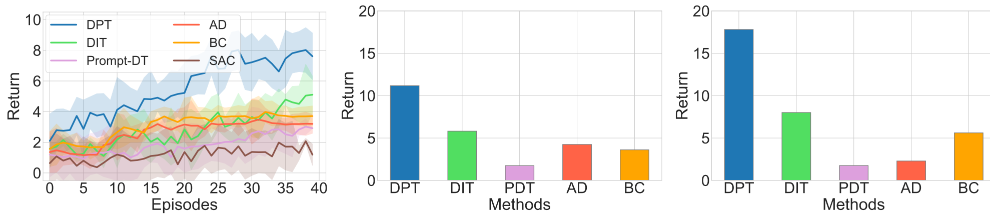


(a) Online (b) Offline Random (c) Offline Expert

Figure 4: Results on **Dark Room** (higher values indicate better performance). **(a)**: Change in return of policies with additional online episodes for (in-context) learning. **(b)** and **(c)**: Offline evaluations with context trajectories sampled from random and expert policies.

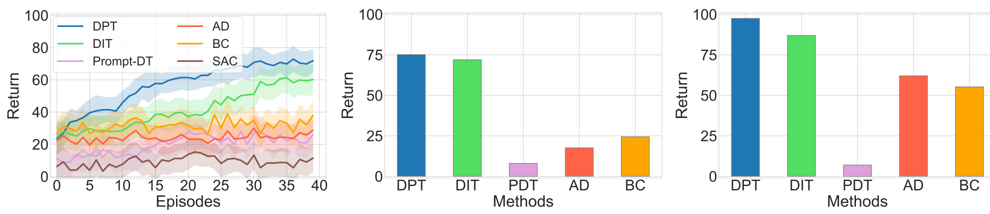
In-context Continuous Control. We explore two complex continuous control tasks, Meta-World (Yu et al., 2020) and Half-Cheetah (Todorov et al., 2012). Meta-World has 20 tasks in total, to evaluate our approach's ability to generate to new RL tasks, we use 15 tasks to train and 5 to test. Similarly, for Half-Cheetah, out of the 40 total tasks, we use 35 tasks to train and 5 to test. The results for Meta-World is presented in Figure 5 and those for Half-Cheetah is presented in Figure 6. We observe that DIT outperforms PDT and BC in all testing scenarios. Moreover, *DIT consistently outperforms AD despite with less information used for pretraining*. It can also be observed that the performance gap between DPT and DIT is larger in the Meta-World environment compared to Half-Cheetah. We believe this is because Meta-World is a more challenging environment than Half-Cheetah. As a result, the *additional set of optimal action labels for out-of-trajectory query states* used by DPT has a greater impact on performance, while DIT can only utilize in-trajectory states and actions as query states with pseudo-optimal labels.

Ablation Study on Weighted Supervised Pretraining. While DIT's significantly improved performance over BC (the unweighted version of DIT) already demonstrates the effectiveness of the proposed weighted pretraining objective, we now conduct experiments in the Miniworld (Chevalier-



(a) Online (b) Offline Random (c) Offline Expert

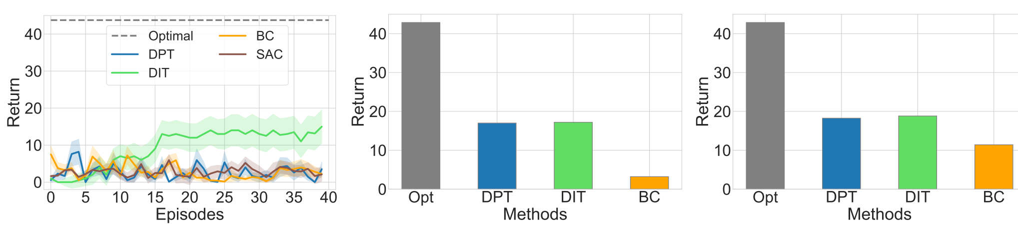
Figure 5: Results on **Meta-World**. **(a)**: Online testing. **(b)** and **(c)**: Offline evaluations.



(a) Online (b) Offline Random (c) Offline Expert

Figure 6: Results on **Half-Cheetah**. **(a)**: Online testing. **(b)** and **(c)**: Offline evaluations.

Boisvert et al., 2023) environment to explore whether DIT reaches the ***limits*** of the weighted pretraining framework. To this end, we compare our model to the *DPT model that uses a pretraining dataset containing only query states that belong to the set of observed states in the pretraining dataset*, along with their associated optimal action labels. In this scenario, the *total number* of pretraining context datasets and optimal action labels for DPT remains the same, but the query states are restricted. This restriction makes the DPT model function as an oracle upper bound for DIT, as all query states used by DIT in the weighted pretraining originate from the observed states. *The significant performance gain of DIT over BC (the unweighted version of DIT) demonstrate the effectiveness of the weighted pretraining framework*. Surprisingly, in the online setting, DPT struggles to perform, while DIT models gradually improve their returns, as shown in Figure 7a. In the offline setting, DIT again demonstrates competitive performance with DPT. These results indicate that *DIT has effectively leveraged the pretraining dataset to a significant extent*.



(a) Online (b) Offline Random (c) Offline Expert

Figure 7: Ablation Study on **Miniworld**.

6 Discussion

We have proposed DIT for pretraining LTMs from suboptimal historical data for in-context RL. DIT has guaranteed policy improvements over the suboptimal behavior policies and thus demonstrated superior empirical performance. Despite these strengths, DIT still requires the behavior policies that collected the historical data from various RL instances to have reasonable rewards. Most historical data typically adheres to this constraint. That said, it is highly unlikely to infer near-optimal actions solely from random trajectories without any information about optimal policies. To this end, we will further explore the limits of the weighted pretraining framework in future work.

Ethics Statement. This work explores pretraining transformer models for in-context reinforcement learning (RL). We do not anticipate any immediate ethical concerns.

Reproducibility Statement. We have provided details about how the datasets used in benchmark environments are generated and how models are trained, including values of all hyperparameters. We have also provided in supplement material implementation of our algorithms in Python.

REFERENCES

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pp. 22–31. PMLR, 2017.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

David Brandfonbrener, Alberto Bietti, Jacob Buckman, Romain Laroche, and Joan Bruna. When does return-conditioned supervised learning work for offline reinforcement learning? *Advances in Neural Information Processing Systems*, 35:1542–1553, 2022.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020a.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020b.

Clément L Canonne. A short note on an inequality between kl and tv. *arXiv preprint arXiv:2202.07198*, 2022.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo de Lazcano, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. *CoRR*, abs/2306.13831, 2023.

Juncheng Dong, Weibin Mo, Zhengling Qi, Cong Shi, Ethan X Fang, and Vahid Tarokh. Pasta: pessimistic assortment optimization. In *International Conference on Machine Learning*, pp. 8276–8295. PMLR, 2023.

Vincent François-Lavet, Peter Henderson, Riashat Islam, Marc G Bellemare, Joelle Pineau, et al. An introduction to deep reinforcement learning. *Foundations and Trends® in Machine Learning*, 11(3-4):219–354, 2018.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

Ying Jin, Zhuoran Yang, and Zhaoran Wang. Is pessimism provably efficient for offline rl? In *International Conference on Machine Learning*, pp. 5084–5096. PMLR, 2021.

Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

Enkelejda Kasneci, Kathrin Seßler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, et al. Chatgpt for good? on opportunities and challenges of large language models for education. *Learning and individual differences*, 103:102274, 2023.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33: 21810–21823, 2020.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

Michael Laskin, Luyu Wang, Junhyuk Oh, Emilio Parisotto, Stephen Spencer, Richie Steigerwald, DJ Strouse, Steven Hansen, Angelos Filos, Ethan Brooks, et al. In-context reinforcement learning with algorithm distillation. *arXiv preprint arXiv:2210.14215*, 2022.

Jonathan Lee, Annie Xie, Aldo Pacchiano, Yash Chandak, Chelsea Finn, Ofir Nachum, and Emma Brunskill. Supervised pretraining can learn in-context reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Wenzhe Li, Hao Luo, Zichuan Lin, Chongjie Zhang, Zongqing Lu, and Deheng Ye. A survey on transformers in reinforcement learning. *Transactions on Machine Learning Research*, 2023a. ISSN 2835-8856. URL `https://openreview.net/forum?id=r30yuDPvf2`. Survey Certification.

Wenzhe Li, Hao Luo, Zichuan Lin, Chongjie Zhang, Zongqing Lu, and Deheng Ye. A survey on transformers in reinforcement learning, 2023b. URL `https://arxiv.org/abs/2301.03044`.

Zuxin Liu, Zijian Guo, Yihang Yao, Zhepeng Cen, Wenhao Yu, Tingnan Zhang, and Ding Zhao. Constrained decision transformer for offline safe reinforcement learning. In *International Conference on Machine Learning*, pp. 21611–21630. PMLR, 2023.

Tatsuya Matsushima, Hiroki Furuta, Yutaka Matsuo, Ofir Nachum, and Shixiang Gu. Deployment-efficient reinforcement learning via model-based offline optimization. *arXiv preprint arXiv:2006.03647*, 2020.

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, Mohammad 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, Jan 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 H. 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 B. 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, 2024.

Rafael Figueiredo Prudencio, Marcos ROA Maximo, and Esther Luna Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

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

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL http://jmlr.org/papers/v22/20-1364.html.

Paria Rashidinejad, Banghua Zhu, Cong Ma, Jiantao Jiao, and Stuart Russell. Bridging offline reinforcement learning and imitation learning: A tale of pessimism. *Advances in Neural Information Processing Systems*, 34:11702–11716, 2021.

Daniel J Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, Zheng Wen, et al. A tutorial on thompson sampling. *Foundations and Trends® in Machine Learning*, 11(1):1–96, 2018.

John Schulman. Trust region policy optimization. *arXiv preprint arXiv:1502.05477*, 2015.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, et al. Large language models encode clinical knowledge. *Nature*, 620(7972):172–180, 2023.

Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. Large language models in medicine. *Nature medicine*, 29(8):1930–1940, 2023.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033. IEEE, 2012.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. 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, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu 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. *ArXiv*, abs/2307.09288, 2023. URL `https://api.semanticscholar.org/CorpusID:259950998`.

Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial intelligence review*, 18:77–95, 2002.

Qing Wang, Jiechao Xiong, Lei Han, Han Liu, Tong Zhang, et al. Exponentially weighted imitation learning for batched historical data. *Advances in Neural Information Processing Systems*, 31, 2018.

Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. Bloomberggpt: A large language model for finance. *ArXiv*, abs/2303.17564, 2023a. URL `https://api.semanticscholar.org/CorpusID:257833842`.

Tianyu Wu, Shizhu He, Jingping Liu, Siqi Sun, Kang Liu, Qing-Long Han, and Yang Tang. A brief overview of chatgpt: The history, status quo and potential future development. *IEEE/CAA Journal of Automatica Sinica*, 10(5):1122–1136, 2023b.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning, 2019.

Chenjun Xiao, Yifan Wu, Jincheng Mei, Bo Dai, Tor Lattimore, Lihong Li, Csaba Szepesvari, and Dale Schuurmans. On the optimality of batch policy optimization algorithms. In *International Conference on Machine Learning*, pp. 11362–11371. PMLR, 2021.

Mengdi Xu, Yikang Shen, Shun Zhang, Yuchen Lu, Ding Zhao, Joshua Tenenbaum, and Chuang Gan. Prompting decision transformer for few-shot policy generalization. In *international conference on machine learning*, pp. 24631–24645. PMLR, 2022.

Taku Yamagata, Ahmed Khalil, and Raul Santos-Rodriguez. Q-learning decision transformer: Leveraging dynamic programming for conditional sequence modelling in offline rl. In *International Conference on Machine Learning*, pp. 38989–39007. PMLR, 2023.

Hongyang Yang, Xiao-Yang Liu, and Christina Dan Wang. Fingpt: Open-source financial large language models, 2023.

Mengjiao Yang, Dale Schuurmans, Pieter Abbeel, and Ofir Nachum. Dichotomy of control: Separating what you can control from what you cannot. *arXiv preprint arXiv:2210.13435*, 2022.

Ming Yin and Yu-Xiang Wang. Towards instance-optimal offline reinforcement learning with pessimism. *Advances in neural information processing systems*, 34:4065–4078, 2021.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pp. 1094–1100. PMLR, 2020.

William Yuan, Jiaxing Chen, Shaofei Chen, Lina Lu, Zhenzhen Hu, Peng Li, Dawei Feng, Furong Liu, and Jing Chen. Transformer in reinforcement learning for decision-making: A survey. *Authorea Preprints*, 2023.

Qinqing Zheng, Amy Zhang, and Aditya Grover. Online decision transformer. In *international conference on machine learning*, pp. 27042–27059. PMLR, 2022.

A Baselines

A.1 Bandit Algorithms

Empirical Mean (EMP). We follow Lee et al. (2024) to consider a strengthened version of EMP which, in the offline setting, only chooses from actions that have been observed at least once in the offline dataset while, in the online setting, at least choosing every action once. At every time step, EMP chooses actions as

$$\widehat{a} \in \arg\max_{a \in \mathcal{A}} \{\widehat{\mu}_a\},$$

where $\widehat{\mu}_a$ is the average observed reward for action a.

Upper Confidence Bound (UCB). Motivated by the Hoeffding's Inequality, at each time step, UCB chooses actions as

$$\widehat{a} \in \arg\max_{a \in \mathcal{A}} \left\{\widehat{\mu}_a + C \cdot \sqrt{1/n_a}\right\},$$

where C is a hyperparameter and n_a is the number of times a has been chosen. For unseen actions, $\widehat{\mu}_a$ is set to 0 and n_a is set to 1. We follow Lee et al. (2024) to set C to be 1 as it demonstrates the best empirical performance.

Lower Confidence Bound (LCB). LCB is on the contrary of UCB. In the offline setting, LCB only chooses from observed actions in the offline dataset. Specifically, it chooses actions as

$$\widehat{a} \in \arg\max_{a \in \mathcal{A}} \left\{\widehat{\mu}_a - C \cdot \sqrt{1/n_a}\right\},$$

where C is a hyperparameter and n_a is the number of times a has been chosen. Similar to hyperparameter of UCB, the hyperparameter C for LCB is also set to 1 due to its strong empirical performance.

Thompson Sampling (TS). We use Gaussian TS Russo et al. (2018) with a Gaussian prior. The mean and variance of the prior are set to the true mean and variance of the pretraining tasks: 0 for mean and 1 for variance.

A.2 RL Baselines

Decision-Pretrained Transformer (DPT). The Decision-Pretrained Transformer (DPT) is designed to perform in-context learning for reinforcement learning (RL) tasks by leveraging a supervised pretraining approach. The core idea is to train a transformer model to predict optimal actions given a query state and a corresponding in-context dataset, which contains interactions from a variety of tasks. These interactions are represented as transition tuples consisting of states, actions, and rewards, offering context for decision-making. During pretraining, DPT samples a distribution of tasks. For each task T_i, an in-context dataset D_i is constructed to include sequences of state-action-reward interactions that represent past experience with that task. Additionally, a query state s^* is sampled from the MDP's state distribution, and the model is trained to predict the optimal action based on this query state and the context D_i. Formally, the training objective is to minimize the expected loss over the sampled task distribution by predicting a distribution over actions given the state and context.

Prompt-based Decision Transformer (Prompt-DT). Prompt-DT arranges its data to facilitate few-shot policy generalization by using trajectory prompts. For each task T_i, a prompt τ_i^* of length K^* is constructed from few-shot demonstration data P_i, containing tuples of state, action, and reward-to-go $(s^*, a^*, \hat{G}^*)$. This prompt encodes task-specific context necessary for policy adaptation. Additionally, the recent trajectory history τ_i of length K, sampled from an offline dataset D_i, is appended to the prompt to form the full input sequence τ_{input}. Formally, this input sequence is represented as $\tau_{\text{input}} = (\tau_i^*, \tau_i) = (\hat{r}_1^*, s_1^*, a_1^*, \ldots, \hat{r}_{K^*}^*, s_{K^*}^*, a_{K^*}^*, \hat{r}_{K^*+1}, s_{K^*+1}, a_{K^*+1}, \ldots, \hat{r}_{K^*+K}, s_{K^*+K}, a_{K^*+K})$. This sequence contains $3(K^* + K)$ tokens, following the state-action-reward format. The full sequence τ_{input} is then passed through a Transformer model, which autoregressively predicts actions at the heads corresponding to each state token. We follow Prompt-DT's setting and set $k = 20$.

Algorithm Distillation (AD). Algorithm Distillation (AD) transforms the process of reinforcement learning (RL) into an in-context learning task by training a transformer model to predict optimal actions based on a cross-episodic trajectory. AD gathers trajectories from training episodes, where each trajectory T of length H encodes the states, actions, and rewards observed over multiple episodes. Instead of training via traditional gradient updates, AD models the training history to predict actions for subsequent episodes, effectively distilling the behavior of RL algorithms like SAC into the transformer. This enables the model to learn directly from context, facilitating quick adaptation to new tasks and improving learning efficiency.

Behavior Cloning (BC). Behavior Cloning (BC) is a supervised learning approach for imitation learning, where the goal is to learn to mimic the behavior of a policy by mapping states to actions. Specifically, the objective is to minimize the discrepancy between the actions predicted by the learned policy π_θ and the target policy's actions, often through a loss function such as mean squared error or cross-entropy for continuous or discrete action spaces, respectively: $J(\theta) = \mathbb{E}_{(s_t,a_t)\sim D}[\ell(\pi_\theta(s_t), a_t)]$, where D is the dataset of state-action pairs collected from the target policy's demonstrations, s_t is the state at time step t, and a_t is the corresponding target action.

Soft Actor-Critic (SAC). Soft Actor-Critic (SAC) is an off-policy deep reinforcement learning algorithm that balances exploration and exploitation by maximizing a trade-off between expected reward and entropy. The core objective of SAC is to learn a policy that not only maximizes cumulative rewards but also encourages exploration by maximizing the entropy of the policy's actions. SAC uses an actor network to predict actions, and two critic networks to estimate the Q-values of state-action pairs. The training objective involves learning the parameters of the policy to maximize a soft objective function: $J(\pi) = \sum_t \mathbb{E}_{(s_t,a_t)\sim D}[Q(s_t, a_t) - \alpha \log \pi(a_t|s_t)]$, where $Q(s_t, a_t)$ is the Q-value estimated by the critics, α is a temperature parameter controlling the trade-off between reward and entropy, and $\pi(a_t|s_t)$ is the action probability distribution given the state. SAC is trained by sampling mini-batches of transitions from a replay buffer to update the policy (actor) and Q-value estimates (critics). For model and training settings, we use the default implementation from Stable Baselines3 Raffin et al. (2021).

B Theoretical Results

B.1 Proof of Proposition 4.1

Consider the following optimization problem:

$$\max_\pi J(\pi) = \mathbb{E}_{\tau\sim p_\tau, s\sim d_\tau(s), a\sim\pi^b_\tau(a|s)}\left[\underbrace{A^b_\tau(s,a)}_{\text{(I)}} - \eta \cdot \underbrace{D_{\mathrm{KL}}(\pi(\cdot|s;\tau)\|\pi^b_\tau(\cdot|s))}_{\text{(II)}}\right], \tag{11}$$

where D_{KL} is the Kullback–Leibler (KL) divergence, and let $\pi^\star \in \arg\max_\pi J(\pi)$ be its optimizer. Then we have for any policy $\pi(a|s;\tau)$,

$$\begin{aligned}&\mathbb{E}_{\tau\sim p(\tau), s\sim d_\tau(s)}\left[D_{\mathrm{KL}}\left(\pi^\star(\cdot|s;\tau)\|\pi(\cdot|s;\tau)\right)\right]\\ &= -\mathbb{E}_{p(\tau), s\sim d_\tau(s), a\sim\pi^b_\tau(a|s)}\left[\frac{1}{Z_\tau(s)}\exp\left(\frac{A^b_\tau(s,a)}{\eta}\right)\log\pi(a|s;\tau)\right] + C,\end{aligned} \tag{12}$$

where C is a constant independent of π and $Z_\tau(s) = \sum_a \pi^b_\tau(a|s)\exp(A^b_\tau(s,a)/\eta)$.

Proof of Proposition 4.1. For any task τ and fixed state s, we have

$$\begin{aligned}
&\max_{\pi} \mathbb{E}_{a\sim\pi(a|s;\tau)} \left[A^b_\tau(s,a) - \eta \cdot D_{\mathrm{KL}}(\pi(\cdot|s;\tau)\|\pi^b_\tau(\cdot|s))\right] \\
&= \min_{\pi} \mathbb{E}_{a\sim\pi(a|s;\tau)}[\log \frac{\pi(a|s;\tau)}{\pi^b_\tau(a|s)} - \frac{1}{\eta} A^b_\tau(s,a)] \\
&= \min_{\pi} \mathbb{E}_{a\sim\pi(a|s;\tau)} \left[\log \frac{\pi(a|s;\tau)}{\pi^b_\tau(a|s)\exp(A^b_\tau(s,a)/\eta)}\right] \\
&= \min_{\pi} \mathbb{E}_{a\sim\pi(a|s;\tau)} \left[\log \frac{\pi(a|s;\tau)}{\pi^b_\tau(a|s)\exp(A^b_\tau(s,a)/\eta)/Z_\tau(s)} - \log Z_\tau(s)\right] \\
&= \min_{\pi} \mathbb{E}_{a\sim\pi(a|s;\tau)} \left[\log \frac{\pi(a|s;\tau)}{\pi^b_\tau(a|s)\exp(A^b_\tau(s,a)/\eta)/Z_\tau(s)}\right] \quad (Z_\tau(s) \text{ is independent of } \pi) \\
&= \min_{\pi} D_{\mathrm{KL}}(\pi(\cdot|s;\tau)\|\pi^\star_\tau),
\end{aligned}$$

where $\pi^\star_\tau(a|s) = \pi^b_\tau(\cdot|s)\exp(A^b_\tau(s,a)/\eta)/Z_\tau(s)$. Note that the optimum π for a fixed s and task τ is obtained at $\pi = \pi^\star_\tau$, which is unique by the uniqueness property of KL divergence, i.e., $D_{\mathrm{KL}}(\pi\|\pi^\star_\tau) = 0$ if and only if $\pi = \pi^\star_\tau(a|s)$. Thus, the optimal task-conditioned policy is

$$\pi^\star(a|s;\tau) = \pi^\star_\tau = \pi^b_\tau(a|s)\exp(A^b_\tau(s,a)/\eta)/Z_\tau(s).$$

Thus, we further have

$$\begin{aligned}
&\mathbb{E}_{\tau\sim p(\tau), s\sim d_\tau(s)}\left[D_{\mathrm{KL}}\left(\pi^\star(\cdot|s;\tau)\|\pi(\cdot|s;\tau)\right)\right] \\
&= \mathbb{E}_{\tau\sim p(\tau), s\sim d_\tau(s), a\sim\pi^\star(a|s;\tau)}\left[\log \frac{\pi^\star(a|s;\tau)}{\pi(a|s;\tau)}\right] \\
&= \mathbb{E}_{\tau\sim p(\tau), s\sim d_\tau(s)}\left[\sum_a \pi^b_\tau(a|s)\exp(A^b_\tau(s,a)/\eta)/Z_\tau(s)\log \frac{\pi^\star(a|s;\tau)}{\pi(a|s;\tau)}\right] \\
&= -\mathbb{E}_{\tau\sim p(\tau), s\sim d_\tau(s), a\sim\pi^b_\tau(a|s)}\left[\exp(A^b_\tau(s,a)/\eta)/Z_\tau(s)\log \pi(a|s;\tau)\right] + C,
\end{aligned}$$

where $C = \mathbb{E}_{\tau\sim p(\tau), s\sim d_\tau(s), a\sim\pi^b_\tau(a|s)}\left[\exp(A^b_\tau(s,a)/\eta)/Z_\tau(s)\log \pi^\star(a|s;\tau)\right]$. □

B.2 Proof of Proposition 4.2

Let $\pi^\star$ be the policy that optimizes Equation (4). For any task τ and policy π, let $G_\tau(\pi) = \mathbb{E}[\sum_{h=0}^{\infty}\gamma^h r_h|\pi,\tau]$ represent the expected reward of π for τ. Let $\pi^\star_\tau$ denote $\pi^\star(a|s;\tau)$. Then

$$\mathbb{E}_{\tau\sim p_\tau}[G_\tau(\pi^\star_\tau) - G_\tau(\pi^b_\tau)] \geq \frac{\eta}{1-\gamma}\mathbb{E}_{\tau\sim p_\tau}[C^D_\tau] - \frac{2\gamma}{(1-\gamma)^2}\mathbb{E}_{\tau\sim p_\tau}\left[C^A_\tau\sqrt{C^D_\tau/2}\right], \tag{13}$$

where $C^D_\tau = \mathbb{E}_{s\sim d_\tau(s)}[D_{\mathrm{KL}}(\pi^\star(\cdot|s;\tau)\|\pi^b_\tau(\cdot|s))]$ and $C^A_\tau = \max_s|\mathbb{E}_{a\sim\pi^\star(a|s;\tau)}A^b_\tau(s,a)|$.

Proof of Proposition 4.2. First consider any fixed task τ. From Corollary 1 in Achiam et al. (2017), we have

$$G_\tau(\pi^\star_\tau) - G_\tau(\pi^b_\tau) \geq \frac{1}{1-\gamma}\sum_s d_\tau(s)\sum_a \pi^\star(a|s;\tau)A^b_\tau(s,a) - \frac{2\gamma C^A_\tau}{(1-\gamma)^2}\mathbb{E}_{s\sim d_\tau(s)}\|\pi^\star(\cdot|s;\tau) - \pi^b_\tau(\cdot|s)\|_{TV}, \tag{14}$$

where $C^A_\tau = \max_s|\mathbb{E}_{a\sim\pi^\star(a|s;\tau)}A^b_\tau(s,a)|$ and $\|\cdot\|_{TV}$ is the total variation distance between two distributions. In the proof of Propsition 4.1, we observe that: for any τ and s,

$$\pi^\star(\cdot|s;\tau) \in \arg\max_\pi \mathcal{L}(\pi,s) = \mathbb{E}_{a\sim\pi(a|s;\tau)}\left[A^b_\tau(s,a) - \eta\cdot D_{\mathrm{KL}}(\pi(\cdot|s;\tau)\|\pi^b_\tau(\cdot|s))\right].$$

Thus, $\mathcal{L}(\pi^\star_\tau, s) \geq \mathcal{L}(\pi^b_\tau, s)$, which implies that

$$\mathbb{E}_{a\sim\pi^\star(a|s;\tau)}\left[A^b_\tau(s,a) - \eta\cdot D_{\mathrm{KL}}(\pi^\star(\cdot|s;\tau)\|\pi^b_\tau(\cdot|s))\right] \geq \mathbb{E}_{a\sim\pi^b_\tau(a|s;\tau)}\left[A^b_\tau(s,a)\right] = 0.$$

Hence, we have

$$\mathbb{E}_{s\sim d_\tau(s), a\sim\pi^\star(a|s;\tau)}\left[A^b_\tau(s,a)\right] \geq \eta\mathbb{E}_{s\sim d_\tau(s)}[D_{\mathrm{KL}}(\pi^\star(\cdot|s;\tau)\|\pi^b_\tau(\cdot|s))]. \tag{15}$$

Moreover, from Pinsker's inequality (Canonne, 2022),

$$\mathbb{E}_{s\sim d_\tau(s)}\|\pi^\star(\cdot|s;\tau)-\pi_\tau^b(\cdot|s)\|_{TV} \le \mathbb{E}_{s\sim d_\tau(s)}\sqrt{\frac{1}{2}D_{\mathrm{KL}}(\pi^\star(\cdot|s;\tau)\|\pi_\tau^b(\cdot|s))} \tag{16}$$

$$\le \sqrt{\frac{1}{2}\mathbb{E}_{s\sim d_\tau(s)}[D_{\mathrm{KL}}(\pi^\star(\cdot|s;\tau)\|\pi_\tau^b(\cdot|s))]}, \tag{17}$$

where the last inequality comes from Jensen's Inequality. Pluging (16) and (15) into (14), we have

$$G_\tau(\pi_\tau^\star)-G_\tau(\pi_\tau^b) \ge \frac{\eta}{1-\gamma}\mathbb{E}_{s\sim d_\tau(s)}[D_{\mathrm{KL}}(\pi^\star(\cdot|s;\tau)\|\pi_\tau^b(\cdot|s))] \tag{18}$$

$$-\frac{2\gamma C_\tau}{(1-\gamma)^2}\sqrt{\frac{1}{2}\mathbb{E}_{s\sim d_\tau(s)}[D_{\mathrm{KL}}(\pi^\star(\cdot|s;\tau)\|\pi_\tau^b(\cdot|s))]}. \tag{19}$$

Taking expectation with respect to τ concludes the proof:

$$\mathbb{E}_{\tau\sim p_\tau}[G_\tau(\pi_\tau^\star)-G_\tau(\pi_\tau^b)] \ge \frac{\eta}{1-\gamma}\mathbb{E}_{\tau\sim p_\tau}[C_\tau^D] - \frac{2\gamma}{(1-\gamma)^2}\mathbb{E}_{\tau\sim p_\tau}\left[C_\tau^A\sqrt{C_\tau^D/2}\right], \tag{20}$$

where $C_\tau^D = \mathbb{E}_{s\sim d_\tau(s)}[D_{\mathrm{KL}}(\pi^\star(\cdot|s;\tau)\|\pi_\tau^b(\cdot|s))]$ and $C_\tau^A = \max_s|\mathbb{E}_{a\sim\pi^\star(a|s;\tau)}A_\tau^b(s,a)|$. □

B.3 Justification for the identity $Z_\tau(s) = 1$

Assume that $|A_\tau^b(s,a)/\eta| \ll |\log\pi_\tau^b(a|s)|$. Note that this can always be satisfied through reward normalization. Then

$$Z_\tau(s) = \sum_a \pi_\tau^b(a|s)\exp(A_\tau^b(s,a)/\eta) = \mathbb{E}_{a\sim\pi_\tau^b(a|s)}[\exp(A_\tau^b(s,a)/\eta)]$$
$$= \mathbb{E}_{a\sim\pi_\tau^b(a|s)}[1 + A_\tau^b(s,a)/\eta + o((A_\tau^b(s,a)/\eta)^2)] \quad \text{(by Taylor expansion)}.$$

Moreover, by definition of the advantage function, we have

$$\mathbb{E}_{a\sim\pi_\tau^b(a|s)}[A_\tau^b(s,a)] = \mathbb{E}_{a\sim\pi_\tau^b(a|s)}[Q_\tau^b(s,a)] - V_\tau^b(s) = 0.$$

Thus,

$$Z_\tau(s) = 1 + \mathbb{E}_{a\sim\pi_\tau^b(a|s)}[o((A_\tau^b(s,a)/\eta)^2)] \approx 1.$$

C MDP Environments

Dark Room. The agent is randomly placed in a room of 10×10 grids, and there is an *unknown* goal location on one of the grid. Thus, there are $10x10 = 100$ goals. The agent's observation is its current position/grid in the room, i.e., $\mathcal{S} = [10]\times[10]$. The agent needs to move to the goal location by choosing from 5 actions: to move in one of the 4 directions (up, down, left, right) or stay still. The agent receives a reward of 1 only when it is at the goal; otherwise, it receives 0. The horizon for Dark Room is 100. We follow Lee et al. (2024) to use the tasks on 80 out of the 100 goals for pretraining, and reserve the rest 20 goals for testing our models' in-context RL capability for unseen tasks. The optimal actions are defined as: move up or down until the agent is on the same vertical position as the goal; otherwise move left or right until the agent reaches the goal.

Miniworld. The agent is placed in a room with four boxes of different colors, one of which being the target box. The goal is to reach a box of a specific color in the room. The agent receives a $(25\times 25\times 3)$ color image and its 2-D direction as input, and can choose from four possible actions: to turn left/right, move straight forward, or stay still. Similar to Dark Room, it receives a reward of 1 only when it is near the target box while the horizon is 50. The optimal actions are defined as follows: turn left/right towards the correct box if the agent's front is not within 15 degrees of the correct box; otherwise move forward and stay if the agent is near the box.

Meta-World. The agent needs to control a robotic arm to pick up an object and place it at a designated target location. In each task, the state space is in 39 dims including the gripper's position and state (open or closed), the 3D position of the object to be manipulated, and the coordinates of the target location. The agent operates in a continuous action space, where it can adjust the gripper's 3D position and control the open/close state to enable successful grasping and releasing of the object. It provides partial rewards for moving the gripper towards the object, grasping it correctly, transporting it to the target location, and successfully releasing it there. The task goal is to learn an optimal policy that efficiently achieves the sequence of actions required to pick up and accurately place the object at the specified location. Each task has a different goal position. We train in 15 tasks and test in 5 tasks.

Half-Cheetah. The agent needs to control a 2D half-cheetah robot to achieve and maintain varying target velocities, which change across episodes. The state space contains the cheetah's motion, including joint angles, velocities, body velocity, and position. These observations enable the agent to learn intricate movement patterns and maintain balance while running. The action controls the torques applied to each joint of the cheetah, thus dictating its locomotion and stability. The reward is designed to align with the core task objective: matching the agent's velocity to the target velocity. Each task has different target velocity, and we use 35 tasks to train and 5 to test.

D Pretraining Dataset

Pretraining Datasets for Dark Room and Miniworld. To ensure coverage of optimal actions (so that optimal policies can be inferred), at every step, with probability p (respectively $1 - p$) we use optimal policy (respectively random policy) to choose action. We choose p so that the average reward of the trajectories in the pretraining dataset is less than 30% of that of the optimal trajectories, reflecting the challenging yet common scenarios. For Dark Room, to test whether DIT models can generalize to unseen RL problems in context, we collect context datasets from only 80 out of the total 100 goals and reserves the rest 20 for testing. For each training goal, we follow the setting of DPT to collect 1k context datasets, leading to a total of 80k context datasets in the pretraining dataset (64k for training and 16k for validation). For Miniworld, we collect 40k context datasets (32k for training and 8k for validation), 10k datasets for each of the four tasks corresponding to four possible box colors.

Pretraining Datasets for Meta-World and Half-Cheetah. We construct the pretraining datasets using historical trajectories generated by agents trained with *Soft Actor Critic* (SAC). Specifically, SAC is trained until convergence for each task, then we sample from its learning trajectories to build the dataset. Our SAC model training follows the settings outlined in Haarnoja et al. (2018). For the Meta-World environment, we use its built-in deterministic policy as the optimal policy; for Half-Cheetah, we use the optimal SAC policy. In Meta-World, we used 15 tasks to train and 5 to test. Similarly, for Half-Cheetah, we used 35 tasks to train and 5 to test.

E Training Parameters.

For all methods, we use the AdamW optimizer with a weight decay of $1e - 4$, a learning rate of $1e - 3$, and a batch size of 128.

F Model Details

Decision Transformer Architecture. Our model is based on a causal GPT-2 architecture Radford et al. (2019). It consists of 6 attention layers, each with a single attention head, and an embedding size of 256. To separately encode state, action, and reward pairs, we employ three fully connected layers. We use a single fully connected layer to decode from the transformer's output.

Value Function Transformer Architecture. The architecture of the value function transformer mirrors that of the decision transformer.

G Computation Requirements

Our experiments can be conducted on a single A6000 GPU. It typically takes less than one hour to generate the required dataset for training in parallel. For PPO, training usually takes less than 10 minutes per task. For the other methods, we observe that the transformer model converges within 50 epochs.

H Pseudocodes

Algorithm 1 Pretraining of Decision Importance Transformer

1: **Input:** Pretraining Dataset $\mathcal{D} = \{D^i\}$; transformer models $T_\theta, \widehat{Q}_\zeta, \widehat{V}_\phi$.
2: `// In-context Estimation of Advantage Functions`
3: Randomly initialize and train $\widehat{Q}_\zeta$ and $\widehat{V}_\phi$ by optimizing the loss in Equation (9).
4: Construct the in-context advantage estimator as:

$$\widehat{A}_b = \widehat{Q}_\zeta - \widehat{V}_\phi.$$

5: `// Weighted Pretraining`
6: Randomly initialize T_θ.
7: With trained $\widehat{A}_b$ and $\mathcal{D}$, train T_θ by optimizing the loss in Equation (10).

Algorithm 2 Deployment of In-Context RL Models

1: **Input:** Pretrained transformer Model T_θ; Horizon of episodes H; Number of episodes N for online testing; Offline dataset $D_{\text{off}} = \{(s_h, a_h, s_{h+1}, r_h)\}_h$, consisting of transitions collected by a behavioral policy.
2: `// Offline Testing`
3: **for** every time step $h \in \{1, \dots, H\}$ **do**
4: Observe state s_h
5: Sample action with T_θ:

$$a_h \sim T_\theta\left(\cdot | s_h, D_{\text{off}}\right)$$

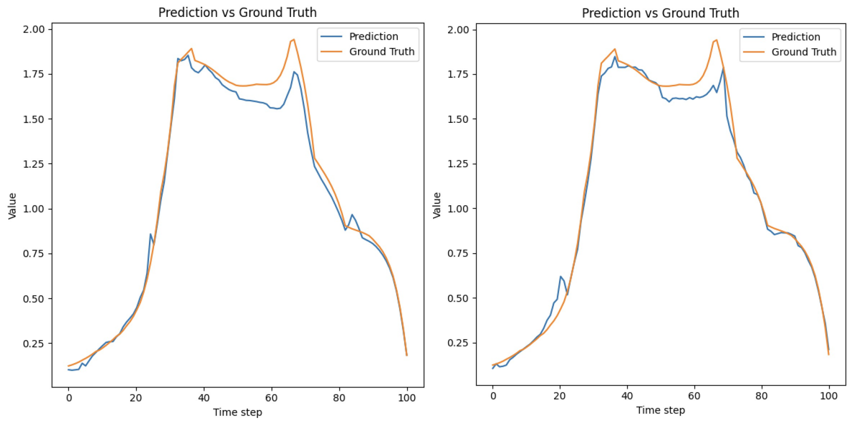
6: Collect reward r_h
7: **end for**
8: `// Online Testing`
9: Initialize an empty online data buffer $D_{\text{on}} = \{\}$
10: **for** every online trial $n \in \{1, \dots, N\}$ **do**
11: **for** every time step $h \in \{1, \dots, H\}$ **do**
12: Observe state s_h
13: Sample action with T_θ:

$$a_h \sim T_\theta\left(\cdot | s_h, D_{\text{on}}\right)$$

14: Collect reward r_h
15: **end for**
16: Append the collected transitions $\{(s_h, a_h, s_{h+1}, r_h)\}_h$ into D_{on}
17: **end for**

I Estimation of Advantage function

Figure 8 illustrates the performance of our value function estimators. Notably, the ground truth labels represent the cumulative rewards empirically sampled using Monte Carlo, rather than the in-trajectory cumulative rewards. From the two graphs, we observe that our function estimator effectively learns the empirical distribution of cumulative rewards. Furthermore, the difference between the Q-function and V-function estimators provides the advantage function.



(a) Q function (b) V function

Figure 8: Performance of Q and V function estimator. On the x-axis is time step of horizon; on the y-axis is the model predictions or ground truth values.

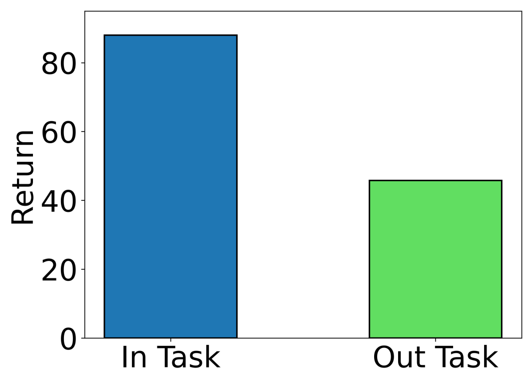


Figure 9: Performance of DIT when the in-context trajectory is aligned (In Task) or misaligned (Out Task) with the current task goal.

J Effectiveness of in-context trajectory

Figure 9 illustrates the effectiveness of the in-context trajectory for DIT. Since DIT predicts actions based on the current state and the historical states in the in-context trajectory, it is crucial to ensure that the task goal of the in-context trajectory aligns with the current task that DIT is predicting. Here, ”In Task” refers to cases where the in-context trajectory is sampled from the same task as the current task, while ”Out Task” indicates that the in-context trajectory is sampled from a different task.

From Figure 9, we observe that alignment between the in-context trajectory and the current task goal is critical for effective performance. This finding also validates that DIT relies heavily on the in-context trajectory for action prediction, as misalignment with the current task goal leads to a significant decrease in performance.