Active Reinforcement Learning Strategies for Offline Policy Improvement

Ambedkar Dukkipati, Ranga Shaarad Ayyagari*, Bodhisattwa Dasgupta*, Parag Dutta*, Prabhas Reddy Onteru

Department of Computer Science and Automation, Indian Institute of Science
{ambedkar, rangaa, bodhisattwad, paragdutta, prabhasreddy}@iisc.ac.in

Abstract

Learning agents that excel at sequential decision-making tasks must continuously resolve the problem of exploration and exploitation for optimal learning. However, such interactions with the environment online might be prohibitively expensive and may involve some constraints, such as a limited budget for agent-environment interactions and restricted exploration in certain regions of the state space. Examples include selecting candidates for medical trials and training agents in complex navigation environments. This problem necessitates the study of active reinforcement learning strategies that collect minimal additional experience trajectories by reusing existing offline data previously collected by some unknown behavior policy. In this work, we propose an active reinforcement learning method capable of collecting trajectories that can augment existing offline data. With extensive experimentation, we demonstrate that our proposed method reduces additional online interaction with the environment by up to 75% over competitive baselines across various continuous control environments such as **`Gym-MuJoCo`** locomotion environments as well as **`Maze2d`**, **`AntMaze`**, **`CARLA`** and **`IsaacSimGo1`**. To the best of our knowledge, this is the first work that addresses the active learning problem in the context of sequential decision-making and reinforcement learning.

Code — https://github.com/sml-iisc/ActiveRL
Extended version — https://arxiv.org/pdf/2412.13106

1 Introduction

Reinforcement learning (Kaelbling, Littman, and Moore 1996; Sutton and Barto 2018) tackles the problem of sequential decision-making in unknown environments. This involves agents exploring the environment to learn from the interactions online. However, relying solely on online interactions may not be feasible in many practical applications like navigation and clinical trials. To overcome this limitation, recently, offline reinforcement learning (Levine et al. 2020; Fujimoto, Meger, and Precup 2019) has emerged, where agents can learn from offline interactions with unknown policies. Although this presents significant challenges and some advancements have been made, a pertinent question arises: In the presence of offline data, how can one enhance an agent's performance when it is permitted to explore only a limited number of interactions with the environment?

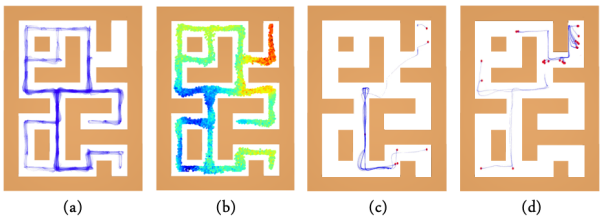

Figure 1: [Best viewed in color] Consider an offline dataset as shown in (a). Our method computes uncertainties in various regions of the environment according to the dataset. As shown in (b), the uncertainties are high in regions where data is not present in the dataset. We collect new trajectories starting from the uncertain regions since that provides more information to the learning algorithm. As can be seen in (c), a simple online trajectory collection policy collects redundant trajectories, while our method focuses on previously unobserved regions, as evident from (d).

For example, in the context of clinical trials, determining the optimal treatment regimen may require access to existing data from related studies while seeking to gather more relevant and informative data by administering treatments to new patients. Each prospective patient presents unique treatment needs and medical histories, often at varying stages of diagnosis. With a constrained budget, it is essential to devise algorithms capable of selecting the optimal subset of participants for treatment, thereby enhancing the utility of the existing data. While this is similar to active learning in a supervised learning setting, it is still an open problem to learn active strategies in reinforcement learning. The main aim of the paper is to address this problem.

Due to budget limitations, there exists a further challenge in deciding whether to pursue a lengthy trajectory or to gather several shorter trajectories across different areas. For instance, in the context of enhancing the navigation capabilities of autonomous vehicles, it is unnecessary to collect data in areas where the existing dataset already contains representative samples. In such instances, any trajectory that enters these regions can be truncated, allowing for the initiation of a new

*These authors contributed equally.

trajectory in a location that offers more informative data.

In the realm of supervised learning, the issue of determining effective methods for data collection is referred to as Active Learning (Cohn, Ghahramani, and Jordan 1996; Settles 2011; Bachman, Sordoni, and Trischler 2017). In this scenario, the agent operates with a limited quantity of labeled data alongside a vast pool of unlabeled data, which incurs significant costs for annotation. The primary goal is to select a small subset of unlabeled data for labeling, thereby enhancing the performance of a model trained on this enriched labeled dataset.

This task becomes increasingly complex in the context of sequential decision-making problems, as the data is represented by samples and trajectories that remain unknown until the agent engages with the environment through an exploration policy. Consequently, rather than merely selecting which data points to label, the agent must decide where, how, and to what extent to explore the environment.

In this work, we develop learning algorithms for enhancing active exploration of the environment utilizing an existing offline dataset. We focus on scenarios where the agent is allocated a limited exploration budget, necessitating the efficient collection of data that can augment the offline data.

Contributions

We consider the problem of active exploration in the context of offline reinforcement learning to minimally augment the offline dataset with informative trajectories to learn an optimal policy.

1. We propose a representation-aware epistemic uncertainty-based method for determining regions of the state space where the agent should collect additional trajectories.
2. We propose an uncertainty-based exploration policy for online trajectory collection that re-uses the representation models.
3. Through extensive experimentation, we empirically demonstrate that our approaches can be widely applicable across a range of continuous control environments. Our active trajectory collection method reduces the need for online interactions by up to 75% when compared to existing fine-tuning approaches.
4. We also perform ablation experiments to demonstrate the importance of each component of our algorithm.

2 Problem Setting

A sequential decision-making problem is formalized by a Markov Decision Process (MDP) defined by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \rho, \gamma)$, where $\mathcal{S}$ is the set of possible states of an environment, $\mathcal{A}$ is the set of actions that can be taken by the agent, $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta\mathcal{S}$ is the transition function, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, ρ is the initial state distribution and $\gamma \in [0, 1)$ is the discount factor. Let $V^\pi(s)$ be the expected sum of discounted rewards obtained by an agent following policy $\pi : \mathcal{S} \to \Delta\mathcal{A}$ from starting state s. The goal of reinforcement learning is to find a policy that maximizes V^π. In offline reinforcement learning, the agent has access only to a fixed set of transitions $\mathcal{D}$ and cannot interact with the environment to collect any additional data. This offline dataset is of the form

$$\mathcal{D} = \{(s_i, a_i, s'_i, r_i, d_i)\}_{1 \le i \le N},$$

where s'_i is a next state sample due to taking action a at state s_i, r_i is the resultant reward, and d_i denotes whether it resulted in episode termination.

First, we explain the problem in a supervised learning setting. Here, given an unlabeled set of data points $\mathcal{D}_{\text{Unl}}$ and a small set of labeled points $\mathcal{D}_{\text{lab}}$, the agent has to choose which points in $\mathcal{D}_{\text{Unl}}$ to label, in the context of what is already available as labeled data. More precisely, consider a supervised learning algorithm Alg that takes a labeled dataset $\mathcal{D}_{\text{lab}}$ and returns a trained model $\pi_{\mathsf{Alg}}(\mathcal{D}_{\text{lab}})$. Further, any model π has to finally perform well on some unknown distribution, quantified by some performance measure V^π. At each stage, the goal of the active learning agent is to choose a point $\tau \in \mathcal{D}_{\text{Unl}}$ to label so as to optimize V of the model learned on this new dataset, i.e.,

$$\tau^* = \operatorname*{argmax}_{\tau \in \mathcal{D}_{\text{Unl}}} V^{\pi_{\mathsf{Alg}}(\mathcal{D}_{\text{lab}} \cup \{\tau\})}.$$

The aim here is to pose this problem in an MDP setting. Due to the sequential nature of the problem, the unlabeled data points in supervised learning correspond to unknown trajectories. Further, a trajectory (which translates to a data point in supervised learning) is not available to be chosen here but can only be observed as a stochastic function of some exploration strategy employed by the active learning agent in the context of constraints imposed by the environment. We formalize these aspects below.

Consider the following MDP $\mathcal{M}_{\text{Act}} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \hat{\rho}, \gamma)$, with a different initial state distribution $\hat{\rho}$ for the active trajectory collection phase. At the start of each episode, a set of candidates initial states $\mathcal{C} = \{s_i \sim \hat{\rho} : 1 \le i \le N\}$ is sampled from $\hat{\rho}$. The agent can choose to start exploring the environment from any subset of these candidates and can also stop exploring at any time, if necessary, to preserve its budget.

Thus, the active learning agent $\mu = (\mathcal{I}, \pi, \beta)$ has three components: (i) an initial state selection function $\mathcal{I} : \mathcal{S} \to \mathbb{R}$ that decides the utility of collecting a trajectory from a given state, (ii) an exploration policy $\pi : \mathcal{S} \to \Delta\mathcal{A}$ that maps states to probability distributions over actions, and (iii) a termination function $\beta : \mathcal{S} \to \{0, 1\}$ that decides whether to terminate the current trajectory.

Such an agent μ induces a distribution $\mathfrak{T}^\mu$ over trajectories due to the stochasticity of the initial state distribution and the exploration policy. The objective of the agent is to collect new samples within a budget in the context of the current dataset $\mathcal{D}$, so as to maximize the performance in the original MDP $\mathcal{M}$ of the policy π_{Alg} trained on this augmented dataset using the offline algorithm Alg, i.e.,

$$\mu = \operatorname*{argmax}_{\mu = (\mathcal{I}, \pi, \beta)} \mathsf{E}_{\tau \sim \mathfrak{T}^\mu} \left[V^{\pi_{\mathsf{Alg}}(\mathcal{D} \cup \{\tau\})} \right].$$

For example, for the medical trials problem discussed in Section 1, candidate initial states are generated by $\hat{\rho}$ corresponding to patients with varied medical history till that point.

Algorithm 1: Active Offline Reinforcement Learning

Input: $\mathcal{D}$: Offline dataset
Alg: Offline RL algorithm
B: Interaction budget
Output: $T = \{\tau_t\}$: Trajectories actively collected from the MDP $\mathcal{M}_{\text{Act}}$
Initialization: $T \leftarrow \phi$;
while $B > 0$ **do**
  LEARN $\mu = (i, \pi, \beta)$ based on $\mathcal{D} \cup T$;
  $\mathcal{C} \sim \hat{\rho}\,(\mathcal{M}_{\text{Act}})$;
  $s_{\text{init}} \leftarrow \underset{1 \le i \le |\mathcal{C}|}{\text{argmax}}\, \mathcal{I}(s_i)$;
  // Collect trajectory from s_{init}
  $s \leftarrow s_{\text{init}}\ \tau = \phi$;
  while $B > 0$ and $\beta(s) = 0$ **do**
    $a \sim \pi(s)$;
    $s' \sim \mathcal{T}(s, a)$;
    $r = r(s, a)$;
    $\tau \leftarrow \tau \cup \{(s, a, s', r)\}$;
    $B \leftarrow B - 1$;
    $s \leftarrow s'$;
  $T \leftarrow T \cup \tau$;
$\pi_{\mathsf{Alg}} \leftarrow \mathsf{Alg}\,(\mathcal{D} \cup T)$

The agent should choose a subset of these based on the state and existing data. Similarly, in the navigation setting, when a known state is reached while exploring the environment, the termination function β stops the current trajectory so as to prevent the collection of redundant samples. The procedure for active exploration is listed in Algorithm 1.

3 Algorithm

A practical implementation of actively collecting trajectories in addition to offline data and learning an optimal agent consists of two components: (i) The base offline reinforcement learning algorithm Alg that learns a policy given a dataset of transitions, and (ii) The active collection strategy $\mu = (\mathcal{I}, \pi, \beta)$. In this work, we consider existing offline algorithms for the first component. For the second active component, the goal is to collect transitions that are diverse and underrepresented in the given offline dataset. To solve this, we propose an epistemic uncertainty-based method, where we learn an ensemble of representation models for encoding states and state-action pairs and use them to estimate the uncertainty of the agent in state and state-action space.

Representation-based Uncertainty Estimation

We consider representation models of the form $\mathcal{E} = (\mathcal{E}^{\mathbf{s}}, \mathcal{E}^{\mathbf{a}})$ to encode state and state-action representations, where $\mathcal{E}^{\mathbf{s}}$ and $\mathcal{E}^{\mathbf{a}}$ encode state and action information respectively. $\mathcal{E}$ has the following modes of operation: (a) given a **state** $\mathbf{s}$ as input, the latent embedding is obtained by passing it through the state encoder, i.e., $\mathcal{E}(\mathbf{s}) \equiv \mathcal{E}^{\mathbf{s}}(\mathbf{s})$, and (b) given a **state-action** pair $(\mathbf{s}, \mathbf{a})$ as input, the latent embedding of the state and action is added after passing the state and action through their respective encoders, i.e., $\mathcal{E}(\mathbf{s}, \mathbf{a}) \equiv \mathcal{E}^{\mathbf{s}}(\mathbf{s}) + \mathcal{E}^{\mathbf{a}}(\mathbf{a})$.

We enforce the following two modeling objectives to align the latent representations learned by $\mathcal{E}$: (i) similar states (or observations) must be clustered in the latent space, and (ii) the embedding of a state-action pair must align with the latent representation of the corresponding next state.

Consider a transition tuple $(\mathbf{s}, \mathbf{a}, \mathbf{s}', \mathbf{r}, \mathbf{d}) \sim \mathcal{D}$. To satisfy the clustering objective, we use $\mathbf{s}$ as the anchor sample, $\mathbf{s}'$ as the positive sample, and any other observation $\mathbf{s}'$ sampled from $\mathcal{D}$ is considered as the negative sample. Embedding vectors $\mathbf{v}$, $\mathbf{v}^+$, and $\mathbf{v}^-$ are obtained by passing $\mathbf{s}$, $\mathbf{s}'$, and $\mathbf{s}''$ respectively through the state encoder $\mathcal{E}^{\mathbf{s}}$. Moreover, to satisfy the transition dynamics modeling objective, we enforce the encoding $\hat{\mathbf{v}}^+ = \mathcal{E}(\mathbf{s}, \mathbf{a})$ to be close to $\mathbf{v}^+$.

We train an ensemble $\{\mathcal{M}_k\}_{k=1}^K$ of such models to maximize the following augmented noise-contrastive loss:

$$\mathrm{L} = \log(\sigma(\mathbf{v} \cdot \mathbf{v}^+)) + \log(1 - \sigma(\mathbf{v} \cdot \mathbf{v}^-)) - \lambda ||\hat{\mathbf{v}}^+ - \mathbf{v}^+||^2,$$

where $\sigma(x) = 1/(1 + \exp(-x))$ is the sigmoid function and λ is a hyper-parameter.

We consider epistemic uncertainty for both initial state selection and exploration. A natural strategy to estimate the same is to use the ensemble $\{\mathcal{E}_k\}_{k=1}^K$ of state representation models and calculate the amount of dissimilarity among the latent representations of said models, i.e., disagreement, for a given state or state-action pair.

Let $\mathbf{v}_k$ denote the latent representation of a state or state-action pair by model $\mathcal{E}_k$. So, for a given state or state-action pair, we have k vectors $\{\mathbf{v}_k\}_{k=1}^K$. We construct a similarity matrix $\mathbb{S}$ as follows:

$$\mathbb{S}_{i,j} = ||\mathbf{v}_i - \mathbf{v}_j||^2 \quad \text{for } i, j = 1, 2, ..., K. \tag{1}$$

We use the value of the largest element in $\mathbb{S}$ as our proxy for epistemic uncertainty of the model w.r.t the environment.

Practical Implementation of Algorithm

In the first phase of our algorithm, we learn the representation models and an RL agent using just the offline dataset. The agent is learned using a suitable offline reinforcement learning algorithm, depending on the environment.

The second active exploration phase consists of two components: (i) Initial state selection and (ii) Trajectory collection from thereon. Given candidate initial states $\mathcal{C} = \{s_i\}$ from $\hat{\rho}$, those initial states are chosen that result in maximum uncertainty for the representation model ensemble $\{\mathcal{E}_k\}_{k=1}^K$, calculated as

$$\text{Uncertainty}(s_i) = \max_{k,k'} ||\mathcal{E}_k^{\mathbf{s}}(s_i) - \mathcal{E}_{k'}^{\mathbf{s}}(s_i)||^2 .$$

For trajectory collection starting at the chosen s_k's, consider π to be the current policy. At the beginning of the active phase, this is just the policy learned on the offline dataset with the appropriate offline RL algorithm. At each step, M actions are sampled from $\pi(.|s)$ for current state s, resulting in M state-action pairs. The policy π could be deterministic, in which case a scaled isotropic Gaussian noise is added to $\pi(s)$ in order to sample multiple actions.

The uncertainty for each pair is calculated using Equation 1, and the action resulting in the most uncertain state-action pair is chosen as the action to execute. This exploration

strategy can be continued until the uncertainty of the current state falls below a certain threshold or the episode terminates.

The degree of exploration is controlled by an ϵ-greedy variant of the exploration policy that explores using the aforementioned environment-aware uncertainty-based procedure with probability ϵ, and simply follows the policy π otherwise. The detailed algorithm is listed in the Appendix (Dukkipati et al. 2024).

Restricted Initial States

In certain environments, it might be infeasible to modify the initial state distribution. The environment could provide us with candidate initial states, albeit restricting us from starting directly at those states. To solve this problem, we propose a modified version of our algorithm in which the agent follows a two-stage policy during active collection. The first stage starts from the default initial state of the environment and goes to the optimal candidate initial state, from which the actual exploration can be done in the second stage. We train two separate policies for these two stages.

We train a goal-based policy on the offline dataset for the first stage. We then create a weighted undirected graph $\mathcal{G}$ from the offline dataset, with each node corresponding to a state and the weight of edge $\{s_i, s_j\}$ being $e^{-||s_i - s_j||}$, and divide the nodes into clusters. The second stage policy is the uncertainty-based exploration policy described previously.

During active collection, the goal-based policy is first used to reach the cluster of states nearest to the identified optimal candidate state, i.e., the candidate state corresponding to the most uncertainty. From here, the agent begins uncertainty-based exploration as described in the previous subsection.

4 Related Work

The use of uncertainty-based methods for actively labeling data points has been studied in the context of supervised learning (Balcan, Beygelzimer, and Langford 2006; Gal, Islam, and Ghahramani 2017).

Similarly, in online reinforcement learning, successful methods for exploring MDPs typically rely on estimates of uncertainty about the Q-values (of state-action pairs) in order to encourage the agent to explore the environment (Osband et al. 2016). Some exploration strategies also rely on uncertainty-based intrinsic rewards or bonuses. Popular approaches include indirect methods for uncertainty estimation such as approximate count (Bellemare et al. 2016), random network distillation (Burda et al. 2019), and curiosity-driven exploration (Pathak et al. 2017). Mai, Mani, and Paull (2022) learn variance ensembles for capturing the uncertainty.

In offline reinforcement learning, both model-free and model-based methods incorporate uncertainty in different ways. MOPO (Yu et al. 2020) and MOREL (Kidambi et al. 2020) are model-based methods in which the epistemic uncertainty of models learned on the offline dataset is explicitly used to induce pessimism in the trained policy. On the other hand, COMBO (Yu et al. 2021) incorporates conservatism without explicitly estimating the uncertainty of the model.

In a model-free setting, UWAC (Wu et al. 2021) approximates the uncertainty through dropout variational inference. EDAC (An et al. 2021) uses the variance of the gradients of an ensemble of Q networks.

The work of Yin et al. (2023) comes closest to our work in terms of application. However, their approach is applicable to the online setting and is primarily constrained to discrete control settings such as Atari 2600. Our approach differs in the following ways: (i) Unlike our approach, they use an ensemble of Q-Networks, and the variance across Q-values defines the uncertainty metric, (ii) they allow resetting to a previously observed state, and (iii) they sample actions from a uniform distribution and use local planning for exploration.

Active Offline Policy Selection (Konyushova et al. 2021) studies a related problem where the goal is to collect additional trajectories for evaluating a given set of policies and determining the best among them. In contrast, our method deals with collecting trajectories for the final goal of learning an optimal policy from the augmented dataset and not just evaluating given policies.

In Go-Explore algorithms (Ecoffet et al. 2021), the agent explores and comes back to already observed states to explore again, which does not work when the environment is largely unexplored. Our method, by contrast, assumes a given set of states and chooses the optimal states from which to start based on the available offline dataset.

5 Experiments

Environments and Datasets

We consider the following continuous control environments for evaluating our representation-aware uncertainty-based active exploration algorithm:

1. **`Maze2d`**: The state is the 2D location on a plane of the agent in the form of a 2D ball. The objective is to navigate towards a goal by adjusting its velocity and direction.
2. **`AntMaze`**: An extension to the **`Maze2d`** environment that includes a virtual ant agent instead that can be manipulated by controlling its joints.
3. **`HalfCheetah`**, **`Hopper`**, and **`Walker2d`**: Locomotion environments in which the objective is to control a 2D stick figure with multiple joints to stably move forward.
4. **`CARLA`**: A self-driving vehicle simulator wherein the agent has to control the acceleration and steering of a vehicle so as to stay in its lane and avoid collisions.
5. **`IsaacSimGo1`**: A GPU-based simulator to control a legged 4×3 DOF quadrupedal robot using proprioceptive measurements along with ego-centric height information of the terrain.

D4RL (Nair et al. 2020) is a collection of offline datasets for training and testing offline RL algorithms. To validate the performance of our active algorithm in the context of limited data, we prune these datasets and create new smaller versions.

We prune the medium and large **`Maze2d`** datasets by removing trajectories near the goal state. Figure 1 shows an example of a pruned dataset. Refer to the Appendix (Dukkipati et al. 2024) for visualization of all the pruned datasets in detail.

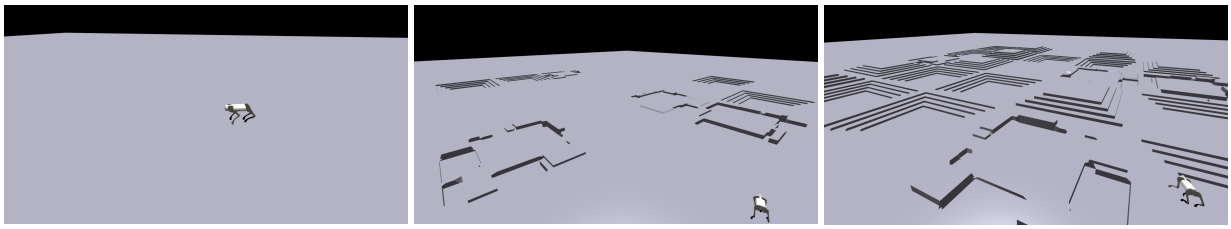

Figure 2: The figures display the terrains for the Unitree Go1 robot experiments in the Nvidia Isaac Simulator. We named the three terrains (from left to right in order) `go1-easy`, `go1-medium` and `go1-hard`. The behavior policy was trained on the `go1-easy` terrain and achieves reasonably high rewards for the locomotion task on the flat surface, as shown. However, we assume that the environment has been modified, and the agent needs to update its policy as quickly as possible in the modified environment. If the agent efficiently uses its exploration budget, then it will be able to generalize the experiences gathered during Active Collection and be able to get high rewards in the `go1-hard` terrain in spite of being given access to `go1-medium` terrain during Active trajectory collection. The accompanying video in the supplementary materials demonstrates the advantage of using our active trajectory collection method.

Additionally, we randomly subsample 30% of the trajectories in the **AntMaze** datasets and the random and medium versions of the locomotion datasets.

For **CARLA**, we use a predefined expert policy to collect the offline dataset. We consider a roundabout with 4 exits. 8 starting "waypoints" are located equidistant from each other throughout the roundabout. The offline dataset is collected with 1 entry and 2 exits. However, the goal exit is not present in the offline dataset. The state space is augmented with the coordinates of the vehicle.

For the **IsaacSim** experiments, we use the `legged_gym` API (Rudin et al. 2022) to simulate Unitree **Go1** robots. Initially, we used the default walking policy[1] for the physical robot to collect an offline dataset, which consists of trajectories on a flat surface. For the final evaluation, the robot is expected to walk on a complex terrain consisting of flights of stairs and discrete obstacles, shown in Figure 2, requiring it to choose suitable starting locations during the active collection phase.

Experimental Settings

For the offline phase of our algorithm, we use (i) TD3+BC (Fujimoto and Gu 2021), (ii) IQL (Kostrikov, Nair, and Levine 2022), (iii) CQL, and (iv) Behavior Cloning, as the base offline RL algorithms. We use TD3+BC in environments such as `maze2d-pruned` and locomotion, Behavior cloning for legged quadrupedal locomotion, and CQL and IQL in **CARLA** and **AntMaze** environments respectively, since TD3+BC does not work in environments where some notion of stitching is required.

In the next (active phase), the policy π obtained offline is improved by using the data collected by our proposed active trajectory collection approach based on the uncertainty estimates induced by the representation model ensemble.

To validate the effectiveness of our proposed active collection, we compare our method with baselines that collect new trajectories without active initial state selection and active exploration. More precisely, the offline phase remains the same, wherein an offline RL algorithm is trained on the given dataset. In the second (fine-tuning) phase (denoted by FT in Table 1), new trajectories are collected starting from the original initial state distribution ρ of the MDP $\mathcal{M}$, using the learned offline policy π as the exploration policy.

Specifically, for the **Maze2d** and locomotion environments, TD3+BC is used as the offline algorithm in the first phase. In the fine-tuning phase, the same training is continued

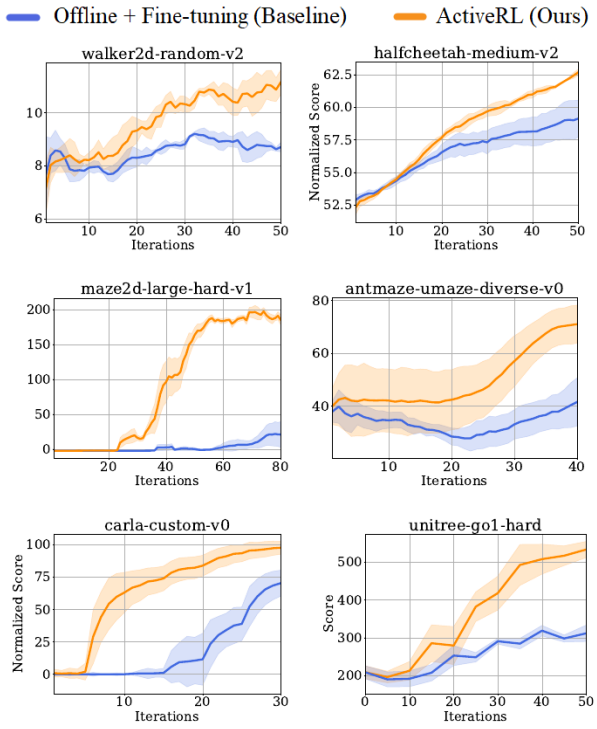

Figure 3: [Best viewed in color] Results of our algorithm compared with the corresponding fine-tuning baseline. In the shaded plots, the results are averaged over multiple random seeds, with the shaded region denoting the standard deviation.

[1]The walking policy is described as Mode 2 in https://unitree-docs.readthedocs.io/en/latest/get_started/Go1_Edu.html

	BC	Offline	Offline + FT	Offline + RND	Offline + AC (Ours)	%age of less interactions
maze2d-medium-easy-v1	-4.5	-4.0	77.5	59.1	**134.3**	62.5
maze2d-large-easy-v1	1.7	-2.0	21.7	10.2	**197.3**	75
maze2d-large-hard-v1	-2.3	-2.0	6.0	1.0	**201.7**	62.5
Maze-pruned total	-5.1	-8.0	105.2	70.3	**533.3**	
antmaze-umaze-v0	62.0	86.7	86.1	81.5	**88.1**	37.5
antmaze-umaze-diverse-v0	54.0	56.0	43.9	39.2	**71.6**	50
antmaze-medium-play-v0	0.0	59.0	68.9	56.8	**73.1**	37.5
antmaze-medium-diverse-v0	1.3	62.3	68.5	62.1	**73.8**	25
antmaze-large-play-v0	0.0	10.3	19.9	14.0	**22.8**	25
antmaze-large-diverse-v0	0.0	9.0	19.8	9.9	**22.9**	37.5
AntMaze-subsampled total	117.3	283.3	307.1	268.2	**352.3**	
halfcheetah-random-v2	2.3	13.5	36.9	41.8	**42.5**	60
hopper-random-v2	4.2	8.2	26.3	23.6	**28.1**	55.5
walker2d-random-v2	2.0	7.9	9.1	10.8	**11.4**	33.3
halfcheetah-medium-v2	42.8	48.3	59.1	58.1	**62.7**	50
hopper-medium-v2	54.0	68.1	93.4	88.4	**96.7**	37.5
walker2d-medium-v2	73.1	83.6	84.9	85.2	**88.5**	-
Locomotion-subsampled total	178.4	229.6	309.7	307.9	**329.9**	
CARLA	0.0	0.0	72.1	88.8	**98.4**	67
unitree-go1-hard	23.1	23.1	34.6	46.7	**59.0**	50
Combined total	313.7	528.0	828.7	781.9	**1372.9**	

Table 1: Results for our active method compared to respective baselines. Mean normalized scores (according to D4RL) are reported across various runs. As can be observed, we consistently improve the performance of the offline trained policy across multiple environments when compared to existing SOTA methods. We also observe a significant reduction in the number of samples required to reach the same performance as the baselines. (We denote inconclusive reduction by '-').

on the newly collected data, with the α value being exponentially decayed to deal with the distribution shift (Beeson and Montana 2022).

For the **`AntMaze`** and **`CARLA`** environments, IQL and CQL, respectively, are used directly for both the offline and online fine-tuning phases, as in Kostrikov, Nair, and Levine (2022); Kumar et al. (2020).

For legged locomotion, we use BPPO (Zhuang et al. 2023) as the offline policy learning algorithm in the active phase, which is perturbed for exploration based on the uncertainty models as described in Section 3.

As an additional trajectory collection baseline for the active phase, we consider Random Network Distillation (Burda et al. 2019), in which the offline learned policy is distilled into an ensemble of smaller networks and used for exploration.

The results are given in Table 1 and Figure 3. Along with the above baselines, we also report the performance of Behavior Cloning (BC) and the base offline algorithm without any additional data collection.

In the final column, we report the percentage of fewer additional interactions with the environment required by our algorithm to reach the best performance of the corresponding Offline + Fine-tuning baseline.

6 Results and Discussion

From Table 1 and Figure 3, one can observe that across the various environments and corresponding datasets, our method demonstrates a significant advantage over the corresponding baselines, both in terms of the rewards obtained as well as the number of samples required to achieve a certain performance. In particular, one can see that our method performs well in scenarios where the behavior policy is sub-optimal and has not learned to explore certain areas of the environment where better rewards are present. Hence, our method augments the offline dataset that does not have good coverage of the state space in a given MDP. For instance, in Table 1, one can

Algorithm	maze2d-large-easy	maze2d-large-hard
BC	1.7	-2.3
Offline	-2.0	-2.0
I+R	45.5	25.0
I+P	0.7	0.2
I+U	51.1	-1.5
A+R	88.1	74.6
A+P	92.9	139.9
A+U	**133.8**	**176.3**

Table 2: Ablation results to understand various components of our approach. For the initial state selection, 'A' denotes active initial state selection, and 'I' denotes usage of the unaltered initial states from the MDP. 'R', 'P', and 'U' denote random policy, offline policy, and uncertainty-based exploration policy, respectively. Active initial state selection followed by an uncertainty-based exploration policy performs best. Further, A and U individually improve performance too.

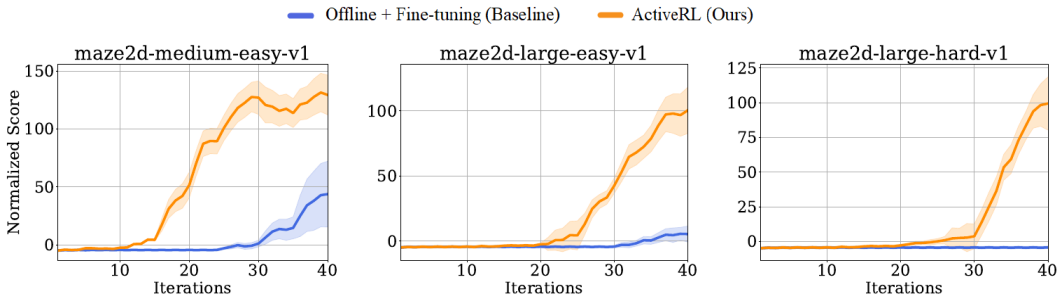

Figure 4: Plots corresponding to experiments where the agent is restricted to start from the original initial state distribution rather than the modified initial state distribution. We use a goal-based policy to reach a state close to the uncertain state and then switch over to our exploration policy.

	TD3	ActiveRL (ours)
maze2d-umaze-v1	142.79	**164.8**
maze2d-medium-v1	148.1	**178.8**
maze2d-large-v1	98.7	**169**
unitree-go1-hard	43.9	**52.8**
Total	433.5	**565.4**

Table 3: Results for ablation with zero initial dataset (analogous to online). It is evident that (using TD3+BC as the base offline algorithm) we perform better than the corresponding online algorithm (TD3).

	var	mean	min	max
maze2d-medium-easy	97.1	108.8	105.6	**110.1**
maze2d-large-easy	111.3	122.5	123.9	**133.8**
maze2d-large-hard	144.6	170.8	159.2	**176.3**

Table 4: Uncertainty metric ablation on **`Maze2d`**.

observe that our method achieves the most performance gain in the pruned `maze2d` datasets where certain regions are missing from the offline dataset.

Additionally, our active method was applied on top of TD3+BC, IQL, and CQL, depending on the environment, showing that it is compatible with multiple offline algorithms.

Ablations To verify the utility of active initial state selection, we perform an ablation by starting exploration only from the given initial state samples from ρ of the original MDP $\mathcal{M}$. The results of this ablation are given in Table 2.

To study the importance of a suitable exploration policy, we conduct an ablation by replacing our uncertainty-based exploration strategy (U) with random exploration (R) and exploration using the learned offline policy (P). The results are shown in Table 2.

From Table 2, we can clearly see that selecting initial states with our method provides an advantage irrespective of the exploration policy used. Conversely, our exploration policy by itself provides an advantage over random and naive strategies. This is true irrespective of how the initial states are chosen.

We also performed ablations to compare our uncertainty estimation technique with other variants. In our method, we take the maximum of the squared difference between estimates by different models in the ensemble. Table 4 shows the performance of our algorithm when this metric is replaced by the variance of the model estimates and the mean, minimum, and maximum of the squared differences, respectively.

Further, we studied the effectiveness of our algorithm for exploring the environment from scratch without any offline dataset. We skip the initial offline policy training step and start from a random policy, collect trajectories, and train on these experiences to simulate online learning. The results are shown in Table 3.

One can see that even in the absence of an initial offline dataset, our exploration strategy gains a significant advantage both in terms of samples used and final performance compared to the corresponding online algorithms.

7 Conclusion

By taking motivation from active learning, which is well-studied in supervised learning settings, we formulated this problem for reinforcement learning in this paper. In the presence of an offline dataset, we proposed active learning strategies with which agents can acquire trajectories of agent-environment interactions to enhance their performance under a limited budget. Our proposed approach consistently performs well across many environments and is compatible with multiple base offline RL algorithms. We compared the performance of our approach with strong baselines and performed ablation studies to understand the role of each component of our method.

Acknowledgements

The authors would like to thank the SERB, Department of Science and Technology, Government of India, for the generous funding towards this work through the IMPRINT Project: IMP/2019/000383.

References

An, G.; Moon, S.; Kim, J.-H.; and Song, H. O. 2021. Uncertainty-based offline reinforcement learning with diversified q-ensemble. In *Advances in Neural Information Processing Systems*, volume 34, 7436–7447.

Bachman, P.; Sordoni, A.; and Trischler, A. 2017. Learning algorithms for active learning. In *International Conference on Machine Learning (ICML)*, 301–310.

Balcan, M.-F.; Beygelzimer, A.; and Langford, J. 2006. Agnostic active learning. In *International Conference on Machine Learning (ICML)*, 65–72.

Beeson, A.; and Montana, G. 2022. Improving TD3-BC: Relaxed Policy Constraint for Offline Learning and Stable Online Fine-Tuning. In *3rd Offline RL Workshop: Offline RL as a "Launchpad"*.

Bellemare, M. G.; Srinivasan, S.; Ostrovski, G.; Schaul, T.; Saxton, D.; and Munos, R. 2016. Unifying Count-Based Exploration and Intrinsic Motivation. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 1479–1487.

Burda, Y.; Edwards, H.; Storkey, A.; and Klimov, O. 2019. Exploration by random network distillation. In *International Conference on Learning Representations (ICLR)*, 1–17.

Cohn, D. A.; Ghahramani, Z.; and Jordan, M. I. 1996. Active learning with statistical models. *Journal of Artificial Intelligence Research*, 4: 129–145.

Dukkipati, A.; Ayyagari, R. S.; Dasgupta, B.; Dutta, P.; and Onteru, P. R. 2024. Active Reinforcement Learning Strategies for Offline Policy Improvement. arXiv:2412.13106.

Ecoffet, A.; Huizinga, J.; Lehman, J.; Stanley, K. O.; and Clune, J. 2021. First return, then explore. *Nature*, 590(7847): 580–586.

Fujimoto, S.; and Gu, S. S. 2021. A Minimalist Approach to Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 34, 20132–20145.

Fujimoto, S.; Meger, D.; and Precup, D. 2019. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning (ICML)*, 2052–2062.

Gal, Y.; Islam, R.; and Ghahramani, Z. 2017. Deep Bayesian Active Learning with Image Data. In *International Conference on Machine Learning (ICML)*, 1183–1192.

Kaelbling, L. P.; Littman, M. L.; and Moore, A. W. 1996. Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research*, 4: 237–285.

Kidambi, R.; Rajeswaran, A.; Netrapalli, P.; and Joachims, T. 2020. MOREL: Model-based offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, 21810–21823.

Konyushova, K.; Chen, Y.; Paine, T.; Gulcehre, C.; Paduraru, C.; Mankowitz, D. J.; Denil, M.; and de Freitas, N. 2021. Active offline policy selection. In *Advances in Neural Information Processing Systems*, volume 34, 24631–24644.

Kostrikov, I.; Nair, A.; and Levine, S. 2022. Offline Reinforcement Learning with Implicit Q-Learning. In *International Conference on Learning Representations (ICML)*.

Kumar, A.; Zhou, A.; Tucker, G.; and Levine, S. 2020. Conservative Q-Learning for Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 33, 1179–1191.

Levine, S.; Kumar, A.; Tucker, G.; and Fu, J. 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*.

Mai, V.; Mani, K.; and Paull, L. 2022. Sample Efficient Deep Reinforcement Learning via Uncertainty Estimation. In *International Conference on Learning Representations (ICLR)*.

Nair, A.; Dalal, M.; Gupta, A.; and Levine, S. 2020. Accelerating Online Reinforcement Learning with Offline Datasets. *CoRR*, abs/2006.09359.

Osband, I.; Blundell, C.; Pritzel, A.; and Van Roy, B. 2016. Deep Exploration via Bootstrapped DQN. In *Advances in Neural Information Processing Systems*, volume 29.

Pathak, D.; Agrawal, P.; Efros, A. A.; and Darrell, T. 2017. Curiosity-Driven Exploration by Self-Supervised Prediction. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 488–489.

Rudin, N.; Hoeller, D.; Reist, P.; and Hutter, M. 2022. Learning to walk in minutes using massively parallel deep reinforcement learning. In *Conference on Robot Learning (CoRL)*, 91–100.

Settles, B. 2011. From Theories to Queries: Active Learning in Practice. In Guyon, I.; Cawley, G.; Dror, G.; Lemaire, V.; and Statnikov, A., eds., *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, volume 16 of *Proceedings of Machine Learning Research*, 1–18. Sardinia, Italy: PMLR.

Sutton, R. S.; and Barto, A. G. 2018. *Reinforcement Learning: An Introduction*. Cambridge: MIT Press, 2nd edition.

Wu, Y.; Zhai, S.; Srivastava, N.; Susskind, J. M.; Zhang, J.; Salakhutdinov, R.; and Goh, H. 2021. Uncertainty Weighted Actor-Critic for Offline Reinforcement Learning. In *International Conference on Machine Learning (ICML)*, 11319–11328.

Yin, D.; Thiagarajan, S.; Lazic, N.; Rajaraman, N.; Hao, B.; and Szepesvari, C. 2023. Sample Efficient Deep Reinforcement Learning via Local Planning. arXiv:2301.12579.

Yu, T.; Kumar, A.; Rafailov, R.; Rajeswaran, A.; Levine, S.; and Finn, C. 2021. COMBO: Conservative offline model-based policy optimization. In *Advances in Neural Information Processing Systems*, volume 34, 28954–28967.

Yu, T.; Thomas, G.; Yu, L.; Ermon, S.; Zou, J. Y.; Levine, S.; Finn, C.; and Ma, T. 2020. MOPO: Model-based offline policy optimization. In *Advances in Neural Information Processing Systems*, volume 33, 14129–14142.

Zhuang, Z.; Kun, L.; Liu, J.; Wang, D.; and Guo, Y. 2023. Behavior Proximal Policy Optimization. In *International Conference on Learning Representations (ICLR)*.