Adaptive Policy Selection and Fine-Tuning under Interaction Budgets for Offline-to-Online Reinforcement Learning

Alper Kamil Bozkurt BOZKURTA@VCU.EDU
Virginia Commonwealth University, Richmond, VA

Xiaoan Xu XIAOAN.XU@DUKE.EDU
Duke University, Durham, NC

Shangtong Zhang XDM2BT@VIRGINIA.EDU
University of Virginia, Charlottesville, VA

Miroslav Pajic MIROSLAV.PAJIC@DUKE.EDU
Duke University, Durham, NC

Yuichi Motai YMOTAI@VCU.EDU
Virginia Commonwealth University, Richmond, VA

Editors: G. Sukhatme, L. Lindemann, S. Tu, A. Wierman, N. Atanasov

Abstract

In offline-to-online reinforcement learning (O2O-RL), policies are first safely trained offline using previously collected datasets and then further fine-tuned for tasks via limited online interactions. In a typical O2O-RL pipeline, candidate policies trained with offline RL are evaluated via either off-policy evaluation (OPE) or online evaluation (OE). The policy with the highest estimated value is then deployed and continually fine-tuned. However, this setup has two main issues. First, OPE can be unreliable, making it risky to deploy a policy based solely on those estimates, whereas OE may identify a viable policy with substantial online interaction, which could have been used for fine-tuning. Second–and more importantly–it is also often not possible to determine a priori whether a pretrained policy will improve with post-deployment fine-tuning, especially in non-stationary environments. As a result, procedures committing to a single deployed policy are impractical in many real-world settings. Moreover, a naive remedy that exhaustively fine-tunes all candidates would violate interaction budget constraints and is likewise infeasible. In this paper, we propose a novel adaptive approach for policy selection and fine-tuning under online interaction budgets in O2O-RL. Following the standard pipeline, we first train a set of candidate policies with different offline RL algorithms and hyperparameters; we then perform OPE to obtain initial performance estimates. We next adaptively select and fine-tune the policies based on their predicted performance via an upper-confidence-bound approach thereby making efficient use of online interactions. We demonstrate that our approach improves upon O2O-RL baselines with various benchmarks.

Keywords: Offline-to-Online Reinforcement Learning, Adaptive Policy Selection & Fine-Tuning

1. Introduction

Dynamic physical systems that employ reinforcement learning (RL) are becoming central to complex missions across civilian, industrial, and defense domains. Examples include autonomous vehicles (Feng et al., 2023; Tian et al., 2024) and robotic platforms (Singh et al., 2022; Campanaro et al., 2024; Chang et al., 2025), which increasingly operate in dynamic, nonstationary environments. By deriving decisions and control directly from onboard sensing and perception, these systems can substantially reduce operator workload and, in turn, lower the incidence of human error (Zhang

et al., 2022). Deep RL (Tang et al., 2025) has been widely adopted to synthesize control policies for high-dimensional, nonlinear physical systems where manual controller design is infeasible. Despite the flexibility and power of this framework, standard RL methods typically require extensive direct interaction with the physical environment for exploration (Ladosz et al., 2022) and are therefore impractical for training policies from scratch when such interactions are costly, risky, or time-consuming (Dulac-Arnold et al., 2021).

Offline RL (Levine et al., 2020; Prudencio et al., 2023) has emerged as an alternative to online RL, enabling the training of policies from large, existing datasets. These datasets are usually collected under safe, controlled conditions (Zhou et al., 2023); often, though not exclusively, by human operators. A central challenge in offline RL is that the performance of a pretrained policy becomes unpredictable as its behavior diverges from that of the policy that collected the dataset (Kostrikov et al., 2021). Due to this distributional shift, a policy pretrained via offline RL can perform arbitrarily poorly in the real environment (Qin et al., 2022). To partially mitigate this issue, offline policy selection, usually via off-policy evaluation (OPE) (Paine et al., 2020; Uehara et al., 2025), is used to identify high-performing policies among candidates pretrained under different hyperparameters. However, OPE estimates are mostly not reliable enough to determine which policy to deploy, primarily since they are also vulnerable to distributional shift (Brandfonbrener et al., 2021), a difficulty further exacerbated by nonstationary real-world environments that demand online adaptation (Julian et al., 2021).

The necessity of online evaluation and fine-tuning of policies pretrained via offline RL has motivated the offline-to-online RL (O2O-RL) paradigm, which treats the entire pipeline as a single joint problem. By combining offline pretraining with a small amount of online interaction, O2O-RL can efficiently yield high-performing policies, enabling broader deployment of RL in physical domains. As a result, O2O-RL has attracted substantial attention and has led to the development of numerous methods. Most O2O-RL approaches (e.g., Nair et al., 2020; Lee et al., 2022; Ball et al., 2023) focus on improving the offline stage to produce pretrained policies that adapt more effectively to real environments via fine-tuning. While these methods improve performance overall, they do not address the fundamental sensitivity of policy performance to hyperparameters and environments. Recently, several methods have examined policy selection in O2O-RL (e.g., Konyushova et al., 2021; Kurenkov and Kolesnikov, 2022), aiming to identify the best candidate by evaluating pretrained policies under a limited budget of online interactions. Yet, these methods do not incorporate fine-tuning into policy selection and overlook the trade-offs that fine-tuning imposes on the same interaction budget.

In this work, we address the problem of obtaining high-performing, deployable policies in the context of O2O-RL from a broader and more practical perspective. Prior work and our experiments show that pretrained policies can perform arbitrarily poorly in real environments, and that fine-tuning may require substantial interaction to improve performance and can even cause regressions, with outcomes varying across algorithms, hyperparameters, and environments. To manage this volatility in performance under interaction budgets, we propose, to our knowledge, the first approach (Figure 1) that jointly combines adaptive policy selection with fine-tuning. In the offline stage of the approach, we first train a diverse set of candidate policies with offline RL, spanning a representative range of algorithms and hyperparameters, and then use OPE to obtain initial performance estimates, similar to Konyushova et al. (2021). In the novel online stage, we actively select and fine-tune candidate policies using an upper-confidence-bound (UCB) criterion on their predicted future performance until the interaction budget is exhausted. After each online episode,

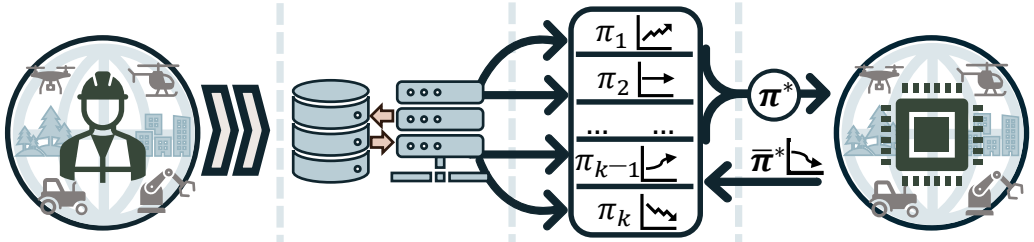

Figure 1: **Proposed O2O-RL Framework.** (a) The datasets are typically collected in controlled, task-agnostic settings. (b) Offline RL trains a diverse set of candidate policies across algorithms and hyperparameters. (c) A linear model predicts the future performance of each policy with a UCB. (d) The policy with the highest UCB is selected and fine-tuned; its predicted performance and UCB are updated during fine-tuning; whenever its UCB falls below that of another candidate, the selected policy is replaced by that candidate.

we evaluate the performance of the policy, then update the prediction and the confidence bound for the selected policy via a linear autoregressive model. We adaptively switch policies when the bound for the selected policy falls below that of another candidate policy. We evaluate our approach on multiple benchmarks and show consistent improvements over O2O-RL baselines.

2. Related Work

2.1. Offline Reinforcement Learning

Basic Approaches. A naive approach to offline RL is to train policies via standard online, off-policy algorithms such as deep deterministic policy gradient (DDPG, Lillicrap et al., 2016), twin delayed DDPG (TD3, Fujimoto et al., 2018), and soft actor-critic (SAC, Haarnoja et al., 2018) on batches drawn from previously collected datasets. However, such policies can fail catastrophically, as they may lead to states outside the dataset due to substantially inaccurate value estimates arising from distributional shift. In contrast, behavior cloning (BC; e.g., Torabi et al., 2018; Florence et al., 2022), which aims to imitate the dataset behavior and thereby largely prevent distributional shift, often cannot surpass the performance of the behavior policy, which is undesirable for non-expert datasets.

Offline RL for Distributional Shift. The central objective in offline RL is to improve upon the behavior policy of the dataset while avoiding catastrophic failures caused by distributional shift. Most offline RL methods therefore balance this trade-off by permitting off-policy updates with additional regularization or constraints. Some approaches, such as batch-constrained deep Q-learning (BCQ, Fujimoto et al., 2019), bootstrapping error accumulation reduction (BEAR, Kumar et al., 2019), and policy in the latent action space (PLAS, Zhou et al., 2021), focus on constraining actions to prevent policies from taking actions outside the dataset. Other approaches, such as conservative Q-learning (CQL, Kumar et al., 2020) and TD3+BC (Fujimoto and Gu, 2021), incorporate, respectively, Q-value and BC regularization terms into the policy updates. We refer readers to Levine et al. (2020) and Prudencio et al. (2023) for comprehensive reviews of offline RL approaches.

2.2. Offline-to-Online Reinforcement Learning

Offline RL for Fine-Tuning. The fundamental limitation that the performance of offline RL methods is upper-bounded by dataset quality necessitates fine-tuning pretrained policies via online interactions. This insight has motivated methods that treat the entire offline-to-online framework as a unified problem, where the main objective is to improve the final performance of policies after fine-tuning. Advantage weighted actor critic (AWAC, Nair et al., 2020) improves the efficiency of fine-tuning with off-policy data via dynamic programming. Implicit Q-learning (IQL, Kostrikov et al., 2021) utilizes a policy-extraction step during training that aids fine-tuning. Calibrated Q-learning (CalQL, Nakamoto et al., 2023), building on CQL, learns conservative Q-values better suited for fine-tuning. Revisited behavior regularized actor-critic (ReBRAC, Tarasov et al., 2023) extends TD3+BC with simple hyperparameter choices that benefit both offline training and fine-tuning. Hybrid RL (Hy-Q, Song et al., 2023) augments offline and online data and applies a fitted Q-iteration procedure. In this work, we do not attempt to improve any specific offline RL algorithm. Instead, we treat the choice of algorithm as a high-level hyperparameter when training candidate policies and fine-tuning, since no single offline RL method is uniformly best in all environments.

Online Policy Selection. A recent line of work, Konyushova et al. (2021) and Kurenkov and Kolesnikov (2022), studied the policy selection problem in O2O-RL. In a setting similar to ours, these studies use a limited online-interaction budget to identify the best-performing pretrained policies, rather than relying on OPE estimates as in prior offline RL approaches (Paine et al., 2020; Uehara et al., 2025). Since evaluating all candidates online would quickly exhaust the budget, they propose active selection strategies to decide which policies to evaluate, thereby allocating the budget efficiently. In contrast, our focus is to allocate online interactions to identify the policies that will perform best after fine-tuning under the given budget.

3. Adaptive Policy Selection and Fine-Tuning

3.1. Problem Formulation

Following the standard RL framework, we model the interaction with an environment to perform tasks as Markov decision processes (MDPs). Formally, an MDP $\mathcal{M}$ is a tuple $(S, A, P, p_0, R, \gamma)$ where S is the state set, A is the action set, P and p_0 are the transition and the initial-state distributions, R is the reward function, and γ is the discount factor. The ultimate objective in RL is to obtain a policy $\pi : S \mapsto A$ maximizing the expected cumulative discounted reward; i.e., $\pi^* \coloneqq \operatorname{argmax}_\pi \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$. In offline RL, this objective must be achieved using a dataset $\mathcal{D}$ of transitions $\{(s_i, a_i, r_i, s'_i)\}_{i=1}^{M}$ collected by a typically unknown behavior policy π_b in $\mathcal{M}$ without further interaction with the environment.

In this work, we extend the offline RL setting by allowing a limited budget N of online interactions in addition to the dataset $\mathcal{D}$; i.e., a set of at most N new transitions may be collected from the environment. Unlike Konyushova et al. (2021) and Kurenkov and Kolesnikov (2022), in our O2O-RL setting these online interactions can be used not only for evaluation but also for fine-tuning, introducing a novel trade-off. Given an MDP $\mathcal{M}$, a dataset $\mathcal{D}$ collected from $\mathcal{M}$, and an interaction budget N, our aim is to yield the highest performing policy achievable, in the sense of minimizing regret. We formally define our specific problem as follows:

Problem 1 *Consider a set of K candidate policies trained offline on a dataset using different hyperparameter settings, and let π_j^i denote the policy that is obtained from the ith pretrained policy after j fine-tuning iterations, and let $v_j^i := \mathbb{E}_{\pi_j^i}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right]$ denote its corresponding value. For a given interaction budget N, the optimal selection is then a policy $\pi_{j^*}^{i^*}$ such that $i^*, j^* := \underset{1<i<K, 0<j\leq N}{\operatorname{argmax}} v_j^i$. Our objective is to devise a procedure that fine-tunes each pretrained policy π_0^i for $\bar{j}_i$ iterations and estimates the values $\{\hat{v}_0^i, \ldots, \hat{v}_{\bar{j}_i}^i\}$ under the constraint $\sum_{i=1}^{K}\bar{j}_i \leq N$ such that the value of the selected policy $\hat{v} = \max_{1<i<K}\max_{0<j\leq\bar{j}_i}\hat{v}_j^i$ minimizes regret $= v_{j^*}^{i^*} - \hat{v}$.*

3.2. Main Approach

Our overall procedure is shown in Algorithm 1. We now explain our approach in detail below.

Offline Initialization. We follow the standard O2O pipeline. First, we train a diverse pool of K candidate policies $\{\pi_0^i, \ldots, \pi_0^K\}$ with offline RL on the given dataset $\mathcal{D}$ by sweeping across multiple algorithmic families and hyperparameter ranges to cover the design space suitable for the environment $\mathcal{M}$. We then perform OPE using off-the-shelf methods to estimate the value $\hat{v}_{\text{OPE}}^i$ of each pretrained policy π_0^i using off-the-shelf OPE methods. As we show in our experiments, these OPE estimates could be substantially inaccurate but are often correlated with the actual values. Thus, we use these estimates to rank the policies for initial policy selection and do not directly incorporate them into our fine-tuning value estimation. Instead, we initialize the estimates $\hat{v}_0^i$ with the value estimate $\hat{v}_B$ of the behavior policy π_B calculated using the transitions in $\mathcal{D}$.

Fine-Tuning Value Evolution Model. As we demonstrate in Figure 2, the values of policies during fine-tuning do not always monotonically increase; they may stall or regress, exhibit sudden drops or jumps, and pass through both high- and low-variance regimes. We model this nonstationarity in value during fine-tuning with an autoregressive (AR) process combined with a conditionally heteroscedastic (ARCH) (Engle, 1982) process. In particular, we adopt a linear AR(2) model with two lags and the standard ARCH(1) model with only one lag to capture the short-term upward/downward trends and time-varying variance, respectively. While we favor this minimalist design to highlight the key aspects, other richer models such as AR with (integrated) moving-average (ARMA, ARIMA, Box et al., 2015), or generalized ARCH (GARCH, Bollerslev, 1986) models with higher lags can also be employed, depending on the application. Formally, we consider the following model:

$$\mu_t = \beta_0 + \beta_1 \hat{v}_{t-1}^i + \beta_2 \hat{v}_{t-2}^i \tag{1}$$

$$\hat{v}_t^i = \mu_t + \epsilon_t \tag{2}$$

$$\epsilon_t = \sigma_t \eta_t \tag{3}$$

$$\sigma_t^2 = \alpha_0 + \alpha_1 \epsilon_{t-1}^2 \tag{4}$$

where $\eta_t \overset{\text{iid}}{\sim} N(0,1)$ are identically and independently distributed standard normal residuals. Here, $\beta_0, \beta_1, \beta_2$, and α_0, α_1 are mean and variance parameters to be estimated from observed values. The standard methods such as ordinary least squares (OLS) or quasi-maximum likelihood (QML) can be used to estimate these variables using off-the-shelf tools. To mitigate the initial identification problem due to the lack of early observations, we start with $\underline{t}$ pseudo-observations $(\hat{v}_{-\underline{t}}^i, \ldots, \hat{v}_0^i)$,

Algorithm 1: Adaptive Policy Selection and Fine-Tuning

Input : an MDP $\mathcal{M}$, a dataset $\mathcal{D}$, an interaction budget N
Output: a policy π^*

`# Offline Stage`
Estimate behavior policy value $\hat{v}_B$ using $\mathcal{D}$
Train K candidate policies $\{\pi_0^i\}_{i=1}^K$ on $\mathcal{D}$ with diverse offline RL algorithms/hyperparameters
Estimate the initial policy values $\{\hat{v}_{\text{OPE}}^i\}_{i=1}^K$ via OPE
Argsort the policies based on $\{\hat{v}_{\text{OPE}}^i\}_{i=1}^K$ to obtain OPE ranks $\{r^i\}_{i=1}^K$
Initialize value list $\hat{\mathbf{v}}^i \leftarrow (\hat{v}_{-\underline{t}}^i, \ldots, \hat{v}_0^i)$ with pseudo-observations $\hat{v}_B$ for each π_0^i
heap $\leftarrow$ PriorityQueue$(\{(\pi_0^i, \hat{v}_B, r^i)\}_{i=1}^K)$ `# prioritizes values over OPE ranks`
$\Pi \leftarrow \{\}$ `# policy storage`

`# Online Stage`
for $j = 1$ **to** N **do**
  `# Fine-Tuning & Evaluation`
  $\pi_t^i \leftarrow$ heap.pop()
  $\pi_{t+1}^i \leftarrow \text{finetune}(\mathcal{M}, \pi_t^i)$
  $\hat{v}_{t+1}^i \leftarrow \text{evaluate}(\mathcal{M}, \pi_{t+1}^i)$
  Π.append$((\hat{v}_{t+1}^i, \pi_{t+1}^i))$
  $\mathbf{v}^i$.append$(\hat{v}_{t+1}^i)$
  `# Value Forecast`
  Fit an AR(2)-ARCH(1) model m to $\mathbf{v}^i$
  $\tau \leftarrow \{t{+}2, \ldots, t{+}1{+}N{-}j\}$ `# forecast iteration numbers`
  Simulate values $\{(\tilde{v}_{t'}^{(s)})_{t' \in \tau}\}_{s=1}^R$ using the fitted m
  Compute the UCB (the 95th percentile) value $\bar{v}_{t'}^i$ from $\{\tilde{v}_{t'}^{(s)}\}_{s=1}^R$ for all $t' \in \tau$
  $\bar{v}_*^i \leftarrow \max_{t' \in \tau} \bar{v}_{t'}^i$
  heap.push$((\pi_{t+1}^i, \bar{v}_*^i, 0))$ `# zero implies higher priority than OPE orders`
end
$\pi^* \leftarrow \max\{\Pi\}$ `# max based on value estimates`
return π^*

each set to $\hat{v}_B$, with $\underline{t}$ chosen according to model complexity. We refer readers to Box et al. (2015) and Francq and Zakoian (2019) for methodological estimation details.

Fine-Tuning Value Forecast. After each fine-tuning iteration, we re-estimate the parameters of AR(2)-ARCH(1) accounting for the new observed value estimate and then use these estimated parameters to forecast the future values. Although the mean and the variance in (1) and (4) admit closed-form multi-step forecasts, we rely on simulation to keep our framework more compatible with general models. Concretely, for a fine-tuned policy π_t^i and a remaining budget n, we simulate R sequences of values $\{(\tilde{v}_{t+1}^{(s)}, \ldots, \tilde{v}_{t+n}^{(s)})\}_{s=1}^R$ with residuals drawn from a standard normal distribution. For each future fine-tuning iteration $t' \in \{t{+}1, \ldots, t{+}n\}$, we compute the 95th percentile of $\{\tilde{v}_{t'}^{(1)}, \ldots, \tilde{v}_{t'}^{(R)}\}$, and use these percentiles as UCBs, denoted $\bar{v}_{t'}^i$, on the forecasted values. We then calculate the maximum UCB (max-UCB), $\bar{v}_*^i = \max_{t'=t+1,\ldots,t+n} \bar{v}_{t'}^i$, and push the policy π_t^i with its max-UCB $\bar{v}_*^i$ to a max-priority queue data structure.

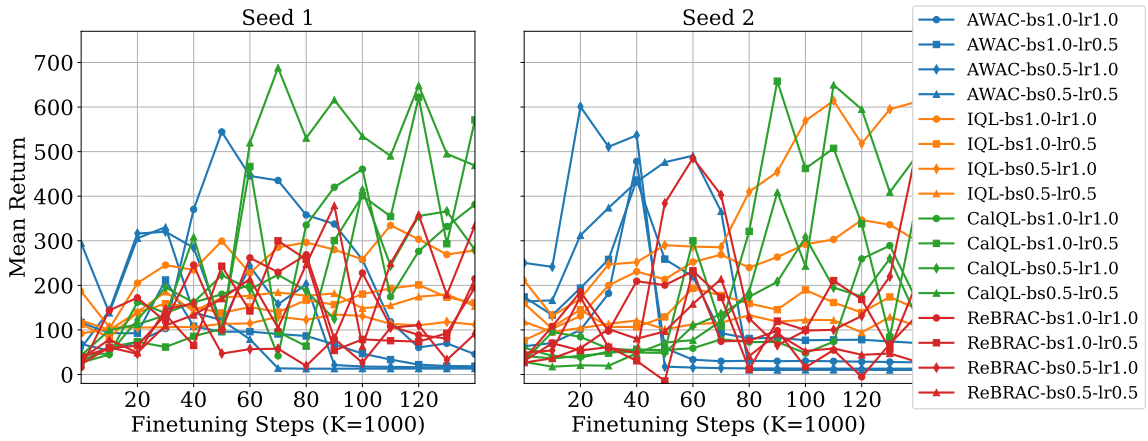

Figure 2: Evolution of the mean return values of pretrained policies during fine-tuning on WALKER-RANDOM for two random seeds. Policies are pretrained offline for 200 K steps using default and half-default batch sizes (bs) and learning rates (lr). Values are obtained by averaging returns over 100 rollouts. The value curves are highly irregular: they may improve (e.g., CalQL), regress after initial improvement (e.g., AWAC), or exhibit high variance. Even with the same algorithms and hyperparameters, changing the seed can significantly alter the outcome of offline training and the progression of the value curves, which necessitates an adaptive policy selection approach.

Policy Selection for Fine-Tuning. The online stage proceeds as an iterative fine-tuning loop. At each iteration, we pop from a priority queue the policy π_t^i with the largest max-UCB, fine-tune it to obtain π_{t+1}^i, evaluate π_{t+1}^i via online interaction, and store π_{t+1}^i with its estimated value $\hat{v}_{t+1}^i$ in the policy set Π. We then forecast the values, compute the updated max-UCB $\bar{v}{*}^i$, and push π_{t+1}^i back into the queue with $\bar{v}{*}^i$. Once the interaction budget is exhausted, we return the policy with the largest estimated value from the policy set Π. To incorporate OPE estimates into selection, we rank the OPE scores and initialize the queue by pushing them alongside $\hat{v}_B$, with the queue ordering prioritizing max-UCBs over OPE ranks. This allows OPE to guide early choices, since all policies initially share the pseudo max-UCB $\hat{v}_B$. Fine-tuned policies are pushed with the highest OPE rank to maintain consistency and to prioritize them over policies that have not yet been fine-tuned whenever their max-UCB is greater than or equal $\hat{v}_B$.

4. Experiments

4.1. Implementation Details

We evaluate our approach on four standard legged-robot locomotion tasks from the D4RL offline RL benchmark (Fu et al., 2020), implemented in PYBULLET (Coumans and Bai, 2021): HOPPER, (HALF)CHEETAH, WALKER(2D), and ANT, where each non-terminated trajectory contains 1,000 transitions. For each environment, we use four datasets (16 in total), covering increasing levels of expertise, namely, RANDOM, MEDIUM, MEDIUM-REPLAY, and MEDIUM-EXPERT, each containing 1,000 K (K=1,000) transitions. See (Farama, 2021) for details on how these datasets are generated.

We employ four representative training algorithms designed for offline pretraining followed by online fine-tuning: AWAC (Nair et al., 2020), IQL (Kostrikov et al., 2021), CalQL (Nakamoto et al., 2023), and ReBRAC (Tarasov et al., 2023). We adopt the implementations of these algorithms provided by the offline RL library d3rlpy (Seno and Imai, 2022). For each algorithm, we consider four hyperparameter settings from the cartesian product of two batch sizes (default and half-default) and two learning rates (default and half-default). We pretrain a candidate policy for each setting for 200 K iterations on datasets, with all remaining hyperparameters fixed to the d3rlpy defaults, yielding 16 pretrained candidate policies in total. We then utilize the Fitted Q-Evaluation (FQE, Le et al., 2019) implementation from d3rlpy to obtain OPE estimates of these pretrained policies, alongside the estimates produced by their respective critic networks. We fine-tune the pretrained policies with online interactions with the environments using the same hyperparameters used to pretrain them. We use the Python library ARCH (Sheppard, 2021) to estimate the AR(2)-ARCH(1) model parameters and forecast the future values of policies.

4.2. Main Results

We compare our adaptive policy selection and fine-tuning approach with several standard O2O-RL baselines. **`OPE`**: The pretrained policy with the highest OPE estimate is selected without fine-tuning. **`Best`**: The pretrained policy with the highest true value is selected without fine-tuning (hypothetical upper bound for reference). **`OE`** (Online Evaluation): All pretrained policies are evaluated online by sharing the interaction budget equally across them; the policy with the highest estimated value is then selected without fine-tuning. **`FT`**: The policy selected by **`OPE`** is fine-tuned using the entire interaction budget. **`Ours`**: See Algorithm 1. The value lists are initialized with $\underline{t}$=5 pseudo-observations. In each fine-tuning iteration, 10 K online transitions are used for fine-tuning and another 10 K for evaluation, for a total cost of 20 K transitions per iteration. The values are forecast and the UCBs are computed with $R = 100$ simulations.

Our main results for online interaction budgets of 160 K and 320 K are presented in Tables 1 and 2, respectively. We compare O2O-RL approaches by evaluating the policies they select, using the mean return over 100 rollouts. For each environment, we rescale the mean returns using min-max normalization, with the minimum set to the value estimate of a random policy on the dataset, and the maximum set to the value of the highest-performing policy ($v^{i^*}_{j^*}$; see Problem 1) that would be obtained if future fine-tuning values were known a priori. The experiments are repeated with four seeds and scores are reported as mean values with associated standard deviations, expressed as percentages.

Across all environments and datasets, the `OPE` scores are typically lower and thus entail higher opportunity loss (regret) relative to the maximum achievable score (i.e., 100%) with fine-tuning, which underscores the importance of leveraging the online interaction budget. The `OE` scores are higher mainly because the budget is large enough to obtain reliable and unbiased value estimates for each pretrained policy through online evaluation. However, `OE` could struggle when the number of candidate policies is large and the budget is insufficient; in such cases, adaptive evaluation approaches (e.g., Konyushova et al. (2021)) can be employed. Nevertheless, the maximum scores attainable by any policy selection approach without fine-tuning–online or offline–is upper-bounded by the values of the best pretrained policies, shown in the `Best` column. By contrast, as `FT` scores indicate, straightforward fine-tuning does not substantially improve upon offline algorithms, since some policies may stagnate or regress during fine-tuning, dragging down the average score.

Table 1: Results for O2O-RL baselines and our approach under a 160 K online interaction budget.

Environment		Offline		Online		
		OPE	Best	OE	FT	Ours
HOPPER	RANDOM	0.6 ± 0.6	3.8 ± 1.1	3.8 ± 1.1	57.6 ± 8.7	$\mathbf{62.1 \pm 20.9}$
	MEDIUM-REPLAY	27.0 ± 4.0	72.6 ± 13.0	72.1 ± 13.6	35.2 ± 4.9	$\mathbf{83.6 \pm 8.7}$
	MEDIUM	30.4 ± 4.9	86.0 ± 2.9	$\mathbf{85.9 \pm 2.9}$	29.5 ± 9.6	84.7 ± 5.6
	MEDIUM-EXPERT	12.2 ± 4.9	74.0 ± 36.4	73.9 ± 36.6	33.6 ± 3.1	$\mathbf{77.2 \pm 26.3}$
	Average	*17.6 ± 3.6*	*59.1 ± 13.3*	*58.9 ± 13.5*	*39.0 ± 6.6*	***76.9 ± 15.4***
CHEETAH	RANDOM	55.8 ± 2.7	91.9 ± 5.1	$\mathbf{91.8 \pm 5.2}$	76.5 ± 5.9	91.7 ± 1.3
	MEDIUM-REPLAY	68.4 ± 7.3	84.7 ± 7.6	84.7 ± 7.6	75.4 ± 5.3	$\mathbf{95.8 \pm 2.5}$
	MEDIUM	46.9 ± 6.6	90.0 ± 5.9	89.8 ± 6.0	56.9 ± 6.3	$\mathbf{92.0 \pm 2.6}$
	MEDIUM-EXPERT	40.3 ± 3.2	98.7 ± 0.9	$\mathbf{97.8 \pm 1.6}$	51.0 ± 6.9	97.4 ± 0.8
	Average	*52.9 ± 4.9*	*91.3 ± 4.9*	*91.0 ± 5.1*	*65.0 ± 6.1*	***94.2 ± 1.8***
WALKER	RANDOM	13.6 ± 2.6	45.1 ± 4.5	41.2 ± 10.3	30.4 ± 4.6	$\mathbf{47.1 \pm 17.2}$
	MEDIUM-REPLAY	34.1 ± 8.1	95.8 ± 3.1	95.3 ± 3.8	28.7 ± 6.8	$\mathbf{96.3 \pm 2.0}$
	MEDIUM	24.9 ± 2.3	92.9 ± 2.9	$\mathbf{91.9 \pm 3.7}$	20.3 ± 2.1	89.9 ± 2.8
	MEDIUM-EXPERT	22.7 ± 3.9	100.0 ± 0.0	$\mathbf{100.0 \pm 0.0}$	6.1 ± 2.1	97.5 ± 2.3
	Average	*23.8 ± 4.2*	*83.4 ± 2.6*	*82.1 ± 4.4*	*21.4 ± 3.9*	***82.7 ± 6.1***
ANT	RANDOM	9.0 ± 6.8	82.1 ± 10.4	81.4 ± 10.8	5.5 ± 9.0	$\mathbf{83.4 \pm 6.7}$
	MEDIUM-REPLAY	29.1 ± 8.9	77.4 ± 13.1	77.4 ± 13.1	37.6 ± 24.1	$\mathbf{81.8 \pm 11.2}$
	MEDIUM	33.6 ± 5.3	69.7 ± 3.4	69.7 ± 3.4	22.8 ± 1.5	$\mathbf{82.3 \pm 3.4}$
	MEDIUM-EXPERT	47.8 ± 9.9	99.1 ± 1.5	$\mathbf{98.7 \pm 1.4}$	19.4 ± 1.9	96.6 ± 2.7
	Average	*29.9 ± 7.7*	*82.1 ± 7.1*	*81.8 ± 7.2*	*21.3 ± 9.1*	***86.0 ± 6.0***
	Overall Average	*31.0 ± 5.1*	*79.0 ± 7.0*	*78.5 ± 7.6*	*36.7 ± 6.4*	***85.0 ± 7.3***

Our approach combines online evaluation and fine-tuning through an adaptive mechanism, effectively taking the best of both worlds. For example, datasets without good transitions such as HOPPER-RANDOM can yield weak pretrained policies resulting in very low scores, whereas fine-tuning can severely deteriorate high-performing pretrained policies as in WALKER-MEDIUM-EXPERT. Our method monitors whether selected policies are improving with fine-tuning and allocates budget accordingly. If a selected policy is improving, the budget is used for further fine-tuning, avoiding the mistake of spending the entire budget on evaluation, as OE does for HOPPER-RANDOM. If a selected policy is not improving, other candidate policies are selected, resulting in the evaluation of more policies. In this way, the budget is devoted to identifying a high-performing pretrained policy rather than to degrading performance with unnecessary fine-tuning. As a result, by avoiding these pitfalls, our approach achieves the best average scores for each environment and overall.

Finally, increasing the online interaction budget (e.g., from 160 K to 320 K; see Table 2) predictably increases the regret of purely offline approaches, since additional fine-tuning raises the achievable maximum score. However, the differences in scores after doubling the budget are modest, suggesting that offline RL with small interaction budgets is often sufficient to obtain high-performing policies, provided the budget is allocated adaptively and effectively as in our approach.

5. Conclusion

We propose, to our knowledge, the first adaptive O2O-RL framework that jointly performs policy selection and fine-tuning under an online interaction budget. We begin by training a diverse set of candidate policies with offline RL across algorithms and hyperparameters, then use OPE to obtain

Table 2: Results for O2O-RL baselines and our approach under a 320 K online interaction budget.

Environment		Offline		Online		
		OPE	Best	OE	FT	Ours
HOPPER	RANDOM	0.5 ± 0.5	2.8 ± 1.5	2.8 ± 1.5	61.5 ± 9.9	**63.3 ± 13.2**
	MEDIUM-REPLAY	24.8 ± 3.0	66.6 ± 10.0	66.1 ± 10.3	61.4 ± 6.0	**80.3 ± 5.6**
	MEDIUM	28.8 ± 6.2	80.8 ± 7.6	80.8 ± 7.6	44.6 ± 2.3	**84.3 ± 8.4**
	MEDIUM-EXPERT	11.6 ± 4.6	70.7 ± 35.1	70.5 ± 35.3	52.7 ± 2.5	**75.4 ± 22.5**
	Average	*16.4 ± 3.6*	*55.2 ± 13.5*	*55.0 ± 13.7*	*55.0 ± 5.2*	***75.8 ± 12.4***
CHEETAH	RANDOM	46.0 ± 8.5	75.1 ± 9.6	75.1 ± 9.6	76.4 ± 9.4	**77.1 ± 10.8**
	MEDIUM-REPLAY	64.1 ± 9.6	79.3 ± 10.1	79.3 ± 10.1	75.8 ± 7.8	**90.7 ± 6.8**
	MEDIUM	46.7 ± 6.7	89.7 ± 5.5	89.6 ± 5.6	72.8 ± 5.0	**91.8 ± 2.3**
	MEDIUM-EXPERT	39.7 ± 3.4	97.1 ± 3.7	**97.1 ± 3.7**	60.4 ± 4.9	95.5 ± 3.5
	Average	*49.1 ± 7.1*	*85.3 ± 7.2*	*85.3 ± 7.2*	*71.4 ± 6.8*	***88.8 ± 5.8***
WALKER	RANDOM	10.4 ± 0.5	35.1 ± 3.8	35.1 ± 3.8	**50.4 ± 2.7**	40.7 ± 4.6
	MEDIUM-REPLAY	33.8 ± 8.3	94.9 ± 3.9	**94.9 ± 3.9**	56.4 ± 2.2	92.1 ± 6.8
	MEDIUM	24.4 ± 2.5	90.9 ± 3.9	**90.0 ± 4.9**	31.4 ± 2.1	89.6 ± 2.8
	MEDIUM-EXPERT	22.7 ± 3.9	100.0 ± 0.0	**100.0 ± 0.0**	28.0 ± 3.5	96.7 ± 3.2
	Average	*22.8 ± 3.8*	*80.2 ± 2.9*	***80.0 ± 3.2***	*41.5 ± 2.6*	*79.7 ± 4.3*
ANT	RANDOM	8.7 ± 6.5	80.1 ± 9.9	79.4 ± 10.3	46.9 ± 3.9	**85.1 ± 4.6**
	MEDIUM-REPLAY	26.6 ± 6.1	73.3 ± 17.2	73.3 ± 17.2	54.7 ± 5.6	**78.9 ± 11.4**
	MEDIUM	33.0 ± 4.8	68.7 ± 2.0	68.7 ± 2.0	48.6 ± 7.1	**82.5 ± 2.1**
	MEDIUM-EXPERT	47.8 ± 9.9	99.1 ± 1.5	**99.1 ± 1.5**	58.2 ± 9.9	97.2 ± 2.9
	Average	*29.1 ± 6.8*	*80.3 ± 7.7*	*80.1 ± 7.7*	*52.1 ± 6.6*	***85.9 ± 5.2***
	Overall Average	*29.4 ± 5.3*	*75.3 ± 7.8*	*75.1 ± 7.9*	*55.0 ± 5.3*	***82.6 ± 7.0***

initial rankings. Acknowledging that pretrained policies can perform arbitrarily poorly and that fine-tuning may stall or regress, depending on the algorithm, hyperparameters, environment, or even random seeds, we allocate scarce online interactions adaptively via a UCB criterion on predicted future values. After each fine-tuning iteration, we evaluate the selected policy, update a linear autoregressive model to refine the value predictions and confidence bounds, and switch when the UCB of another policy exceeds the current one, thereby using the budget efficiently. Across multiple benchmarks, our adaptive selection and fine-tuning procedure consistently improves over O2O-RL baselines.

Although it outperforms strong baselines, our approach leaves ample room for improvement. One limitation is that, after each fine-tuning iteration, it requires a considerable amount of online interaction to evaluate the fine-tuned policy. A future direction could be to leverage exploration rollouts to monitor progress during fine-tuning with little or no explicit online evaluation. Another direction is to incorporate policy similarities, as suggested by Konyushova et al. (2021) to improve efficiency. An orthogonal direction is to develop OPE methods tailored to O2O-RL that focus more on ranking and on the fine-tuning potential of pretrained policies. A further direction is to extend the problem setting to align with the concept of deployment complexity (Huang et al., 2022; Su et al., 2022). Lastly, beyond benchmark evaluation, ablation studies, theoretical analysis, and connections to other paradigms such as Hybrid RL (Song et al., 2023) could provide further understanding and guide future improvement. We believe our framework will contribute to offline RL research by taking one more step toward practical, deployable recipes for real-world systems where online interactions are costly or risky.

Acknowledgments

This work is sponsored in part by the NSF under NAIAD Award 2332744, the National AI Institute for Edge Computing Leveraging Next Generation Wireless Networks under Grant CNS-2112562, as well as the Commonwealth Cyber Initiative's Central Virginia Node under the awards VV-1Q26-001, HV-2025-035, and HC-2025-033. We are deeply grateful to all the reviewers for their thoughtful and constructive feedback.

References

Philip J Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning*, pages 1577–1594. PMLR, 2023.

Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327, 1986.

George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.

David Brandfonbrener, Will Whitney, Rajesh Ranganath, and Joan Bruna. Offline rl without off-policy evaluation. *Advances in neural information processing systems*, 34:4933–4946, 2021.

Luigi Campanaro, Siddhant Gangapurwala, Wolfgang Merkt, and Ioannis Havoutis. Learning and deploying robust locomotion policies with minimal dynamics randomization. In *6th Annual Learning for Dynamics & Control Conference*, pages 578–590. PMLR, 2024.

Eric T Chang, Peter Ballentine, Zhanpeng He, Do-Gon Kim, Kai Jiang, Hua-Hsuan Liang, Joaquin Palacios, William Wang, Pedro Piacenza, Ioannis Kymissis, et al. Spikeatac: A multimodal tactile finger with taxelized dynamic sensing for dexterous manipulation. *arXiv preprint arXiv:2510.27048*, 2025.

Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2021. Accessed 10 November 2025.

Gabriel Dulac-Arnold, Nir Levine, Daniel J Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, 110(9):2419–2468, 2021.

Robert F Engle. Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the econometric society*, pages 987–1007, 1982.

Foundation Farama. Dataset reproducibility guide. https://github.com/Farama-Foundation/d4rl/wiki/Dataset-Reproducibility-Guide, 2021. Accessed 10 November 2025.

Shuo Feng, Haowei Sun, Xintao Yan, Haojie Zhu, Zhengxia Zou, Shengyin Shen, and Henry X Liu. Dense reinforcement learning for safety validation of autonomous vehicles. *Nature*, 615(7953): 620–627, 2023.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Conference on robot learning*, pages 158–168. PMLR, 2022.

Christian Francq and Jean-Michel Zakoian. *GARCH models: structure, statistical inference and financial applications*. John Wiley & Sons, 2019.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

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

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pages 2052–2062. PMLR, 2019.

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*, pages 1861–1870. PMLR, 2018.

Jiawei Huang, Jinglin Chen, Li Zhao, Tao Qin, Nan Jiang, and Tie-Yan Liu. Towards deployment-efficient reinforcement learning: Lower bound and optimality. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=ccWaPGl9Hq.

Ryan Julian, Benjamin Swanson, Gaurav Sukhatme, Sergey Levine, Chelsea Finn, and Karol Hausman. Never stop learning: The effectiveness of fine-tuning in robotic reinforcement learning. In *Conference on Robot Learning*, pages 2120–2136. PMLR, 2021.

Ksenia Konyushova, Yutian Chen, Thomas Paine, Caglar Gulcehre, Cosmin Paduraru, Daniel J Mankowitz, Misha Denil, and Nando de Freitas. Active offline policy selection. *Advances in Neural Information Processing Systems*, 34:24631–24644, 2021.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *Deep RL Workshop NeurIPS*, 2021.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019.

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.

Vladislav Kurenkov and Sergey Kolesnikov. Showing your offline reinforcement learning work: Online evaluation budget matters. In *International Conference on Machine Learning*, pages 11729–11752. PMLR, 2022.

Pawel Ladosz, Lilian Weng, Minwoo Kim, and Hyondong Oh. Exploration in deep reinforcement learning: A survey. *Information Fusion*, 85:1–22, 2022.

Hoang Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. In *International Conference on Machine Learning*, pages 3703–3712. PMLR, 2019.

Seunghyun Lee, Younggyo Seo, Kimin Lee, Pieter Abbeel, and Jinwoo Shin. Offline-to-online reinforcement learning via balanced replay and pessimistic q-ensemble. In *Conference on Robot Learning*, pages 1702–1712. PMLR, 2022.

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.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

Mitsuhiko Nakamoto, Simon Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. *Advances in Neural Information Processing Systems*, 36:62244–62269, 2023.

Tom Le Paine, Cosmin Paduraru, Andrea Michi, Caglar Gulcehre, Konrad Zolna, Alexander Novikov, Ziyu Wang, and Nando de Freitas. Hyperparameter selection for offline reinforcement learning. *arXiv preprint arXiv:2007.09055*, 2020.

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*, 35(8):10237–10257, 2023.

Rong-Jun Qin, Xingyuan Zhang, Songyi Gao, Xiong-Hui Chen, Zewen Li, Weinan Zhang, and Yang Yu. Neorl: A near real-world benchmark for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:24753–24765, 2022.

Takuma Seno and Michita Imai. d3rlpy: An offline deep reinforcement learning library. *Journal of Machine Learning Research*, 23(315):1–20, 2022. URL http://jmlr.org/papers/v23/22-0017.html.

Kevin Sheppard. Univariate volatility modeling, bootstrapping, multiple comparison procedures and unit root tests. https://github.com/bashtage/arch, 2021. Accessed 10 November 2025.

Bharat Singh, Rajesh Kumar, and Vinay Pratap Singh. Reinforcement learning in robotic applications: a comprehensive survey. *Artificial Intelligence Review*, 55(2):945–990, 2022.

Yuda Song, Yifei Zhou, Ayush Sekhari, Drew Bagnell, Akshay Krishnamurthy, and Wen Sun. Hybrid RL: Using both offline and online data can make RL efficient. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=yyBis80iUuU.

DiJia Su, Jason D. Lee, John Mulvey, and H. Vincent Poor. MURO: Deployment constrained reinforcement learning with model-based uncertainty regularized batch optimization, 2022. URL https://openreview.net/forum?id=eWNpRVcfzi.

Chen Tang, Ben Abbatematteo, Jiaheng Hu, Rohan Chandra, Roberto Martín-Martín, and Peter Stone. Deep reinforcement learning for robotics: A survey of real-world successes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 28694–28698, 2025.

Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36:11592–11620, 2023.

Zhen Tian, Dezong Zhao, Zhihao Lin, David Flynn, Wenjing Zhao, and Daxin Tian. Balanced reward-inspired reinforcement learning for autonomous vehicle racing. In *6th Annual Learning for Dynamics & Control Conference*, pages 628–640. PMLR, 2024.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. In *International Joint Conference on Artificial Intelligence*, pages 4950–4957, 2018.

Masatoshi Uehara, Chengchun Shi, and Nathan Kallus. A review of off-policy evaluation in reinforcement learning. *Statistical Science*, 2025.

Rong Zhang, Qibing Lv, Jie Li, Jinsong Bao, Tianyuan Liu, and Shimin Liu. A reinforcement learning method for human-robot collaboration in assembly tasks. *Robotics and Computer-Integrated Manufacturing*, 73:102227, 2022.

Gaoyue Zhou, Liyiming Ke, Siddhartha Srinivasa, Abhinav Gupta, Aravind Rajeswaran, and Vikash Kumar. Real world offline reinforcement learning with realistic data source. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7176–7183. IEEE, 2023.

Wenxuan Zhou, Sujay Bajracharya, and David Held. Plas: Latent action space for offline reinforcement learning. In *Conference on Robot Learning*, pages 1719–1735. PMLR, 2021.