DOOF: Decoupled Offline to Online Finetuning via Dynamics Model

Zifeng Zhuang
Zhejiang University, Westlake University
Hangzhou, China
zhuangzifeng@westlake.edu.cn

Xiao He
Westlake Robotics
Hangzhou, China
hexiao18@mails.ucas.ac.cn

Diyuan Shi
Westlake University, Zhejiang University
Hangzhou, China
shidiyuan@westlake.edu.cn

Ting Wang†
Westlake University
Hangzhou, China
wangting@westlake.edu.cn

Donglin Wang†
Westlake University
Hangzhou, China
wangdonglin@westlake.edu.cn

***Abstract*—Offline reinforcement learning (RL) agents, constrained by suboptimal datasets, typically require further online finetuning before deployment. However, this process suffers from severe distribution shift due to unbalanced state coverage in offline data and the inherent conservatism of offline algorithms, often destabilizing policy improvement and causing performance degradation. In this work, we propose a novel *decoupled offline-to-online framework* that separates policy improvement from distribution shift mitigation. Our key insight leverages the dynamics model in model-based RL: 1) During online interaction, *only* the dynamics model is fine-tuned to adapt to the real environment, avoiding direct policy updates under shifting distributions. 2) The policy is then refined *offline* using the updated dynamics, preserving the conservatism of offline RL while eliminating sudden performance drops. This approach reduces the online phase to supervised dynamics learning—a simpler task than policy optimization under distribution shift—while maintaining sample efficiency. Extensive experiments on standard offline RL benchmarks demonstrate that our framework effectively eliminates distribution shift and achieves superior performance with limited online interaction.**

***Index Terms*—Offline to Online Finetuning, Model-based RL**

I. Introduction

Data-driven offline reinforcement learning (RL) [1] has seen significant progress, from traditional algorithms mitigating out-of-distribution (OOD) overestimation [2–4] to sequence modeling [5]. However, limited dataset quality often renders offline-trained policies suboptimal and difficult to deploy. This motivates offline to online finetuning [6], which enhances performance via online interaction. This finetuning faces distribution shifts caused by unbalanced offline datasets [7] and algorithmic conservatism [2]. While exploring OOD regions is necessary for higher returns, it risks sudden performance drops [6, 8] unless distribution shifts are eliminated. Consequently, finetuning must simultaneously address two issues: 1) policy optimization and 2) distribution shift elimination.

Existing methods typically face a trade-off. Less conservative algorithms [6, 8, 9] reduce the extent of distribution shift but cannot address inherently unbalanced offline data. Conversely, constrained methods [10] ensure safer OOD exploration but limit policy improvement. Because tackling both intertwines their objectives, compromises are inevitable. This raises a natural question:

Can we decouple distribution shift elimination from policy improvement during offline to online finetuning?

Doing so would fundamentally avoid conflicts, maximizing the capabilities of both processes independently.

In this paper, we propose **D**ecoupled **O**ffline to **O**nline **F**inetuning (DOOF) using model-based offline algorithms [11–13]. During online interaction, DOOF *solely* mitigates distribution shift by finetuning a more accurate dynamics model via supervised learning. We use the model's uncertainty to encourage exploration in inaccurate OOD regions. Subsequently, policy finetuning occurs entirely offline using this updated model. This simplifies the online phase, avoids distribution-shift-sensitive policy updates, retains inherent conservatism, and fundamentally prevents sudden policy collapse.

Validated on D4RL datasets [7], DOOF achieves significant improvements with only 10k online and 300k offline steps, whereas baselines stagnate or decline. This exceptionally high efficiency stems from the simplified online phase and uncertainty-guided exploration.

II. Preliminary

A. Offline Reinforcement Learning

Reinforcement learning (RL) is formulated by a Markov Decision Process (MDP) [14] $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, r, P_{\mathcal{M}}, d_0, \gamma\}$, with state space $\mathcal{S}$, action space $\mathcal{A}$, scalar reward function $r(s_t, a_t)$, transition dynamics $P_{\mathcal{M}}(s_{t+1}|s_t, a_t)$, initial state distribution

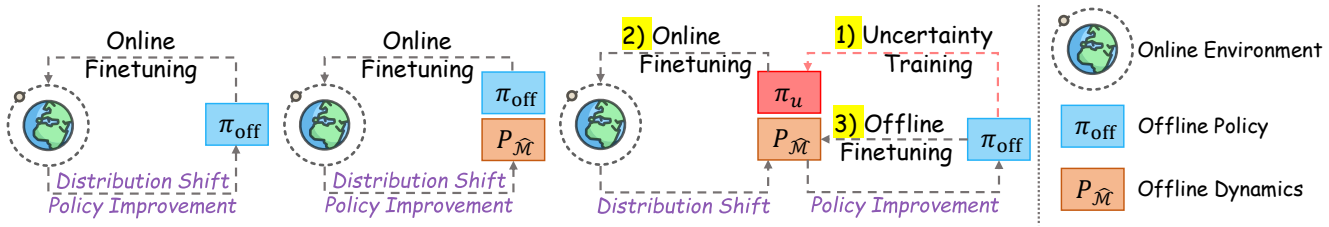

Fig. 1: a) **Model-free** methods directly interact with the environment to finetune the policy, addressing both distribution shift and policy improvement simultaneously. b) **Model-based** methods finetune the policy and dynamics directly, also tackling these two issues. In contrast, c) **DOOF** first finetunes the dynamics through online interaction to overcome distribution shift, and then uses the finetuned dynamics model to finetune the policy in offline manner, achieving policy improvement.

$d_0(s_0)$ and discount factor γ. The objective of RL is to optimize a policy $\pi(a_t|s_t)$ that maximize the expected discounted return

$$J(\pi, \mathcal{M}) = \mathbb{E}_{\tau \sim P_{\pi,\mathcal{M}}(\tau)} \left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t) \right], \tag{1}$$

where $P_{\pi,\mathcal{M}}(\tau) = d_0(s_0) \prod_{t=0}^{T} \pi(a_t|s_t) P_{\mathcal{M}}(s_{t+1}|s_t, a_t)$ is the distribution of trajectory τ generated from the interaction between the policy $\pi(a_t|s_t)$ and the environment $\mathcal{M}$. The value function gives the expected discounted return under policy π when starting from state s in environment $\mathcal{M}$:

$$V^{\pi}_{\mathcal{M}}(s) = \mathbb{E}_{\tau \sim P_{\pi,\mathcal{M}}(\tau|s,a)} \left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t) | s_0 = s \right]. \tag{2}$$

Offline reinforcement learning [1] forbids interaction with the environment $\mathcal{M}$; only a fixed offline dataset $\mathcal{D} = \{(s_t, a_t, r_t, s_{t+1})\}_{t=1}^{N}$ is available. This setting is more challenging since the agent cannot explore the environment or collect feedback, leading to overestimation of out-of-distribution state-action pairs and severely degraded performance.

B. Model-based offline RL algorithms

Existing model-based offline RL methods are built upon model-based policy optimization (MBPO) [11], which contains two stages: transition dynamics pretraining and policy learning. MBPO first estimates a dynamics model[1] $P_{\widehat{\mathcal{M}}}$ from the online buffer or offline dataset $\mathcal{D}$ via maximum likelihood estimation:

$$P_{\widehat{\mathcal{M}}} = \arg\max_{P_{\widehat{\mathcal{M}}}} \mathop{\mathbb{E}}_{(s_t, a_t, s_{t+1}) \sim \mathcal{D}} \left[\log P_{\widehat{\mathcal{M}}}(s_{t+1}|s_t, a_t) \right]. \tag{3}$$

Usually, the dynamics model is considered as predicted Gaussian distribution with K different ensemble models $\{P^k_{\widehat{\mathcal{M}}}(s_{t+1}|s_t, a_t) = \mathcal{N}(\mu^k_\theta(s_t, a_t), \Sigma^k_\phi(s_t, a_t))\}_{k=1}^{K}$. With the learned dynamics model $P_{\widehat{\mathcal{M}}}$, we can construct an estimated MDP $\widehat{\mathcal{M}} = \{\mathcal{S}, \mathcal{A}, r, P_{\widehat{\mathcal{M}}}, d_0, \gamma\}$. Thereafter, MBPO employs the Soft Actor-Critic (SAC) [16] to optimize the policy within the estimated MDP $\widehat{\mathcal{M}}$. An augmented dataset $\mathcal{D} \cup \mathcal{D}_{\widehat{\mathcal{M}}}$ is used to train the policy, where $\mathcal{D}_{\widehat{\mathcal{M}}}$ contains synthetic data generated by performing rollouts in $\widehat{\mathcal{M}}$ starting from states in $\mathcal{D}$. During policy training, mini-batches are sampled from $\mathcal{D} \cup \mathcal{D}_{\widehat{\mathcal{M}}}$, with each data point drawn from the real data $\mathcal{D}$ with the probability p, and from $\mathcal{D}_{\widehat{\mathcal{M}}}$ with probability $1-p$. Model-based offline policy optimization (MOPO) [12] proposes to penalize the reward by the uncertainty $u(s, a)$ of the learned ensemble dynamics models:

$$\hat{r}_t(s, a) = r_t(s, a) - \lambda u(s, a), \tag{4}$$

where penalty coefficient λ is a hyperparameter. While the dynamics uncertainty $u(s, a)$ is usually selected empirically [12], MOBILE [13] theoretically conducts quantification through the inconsistency of Bellman estimations.

C. Offline to Online Finetuning

In this paper, we only consider offline RL with datasets that comprise sub-optimal trajectories rather than optimal ones. If optimal, naive supervised methods such as behavior cloning would sufficient to learn an optimal policy, which is not the issue that offline RL aims to address.

Definition II.1 (Offline to Online Finetuning)**.** *Assume the offline dataset $\mathcal{D}$ is sub-optimal and the offline pretrained policy π_{off} is still sub-optimal.* ***Offline to Online Finetuning*** *aims to improve the performance of π_{off} through the interaction with environment $\mathcal{M}$.*

a) Challenges: The state-action space can be divided into three segments based on the alignment of state-action distributions with offline dataset: a) in-distribution (ID) state-action pairs $(s_{\mathcal{D}}, a_{\mathcal{D}})$ with $s \sim \mathcal{D}, a \sim \mathcal{D}$, b) ID state but out-of-distribution (OOD) action pairs $(s_{\mathcal{D}}, a_{\neg\mathcal{D}})$ with $s \sim \mathcal{D}, a \not\sim \mathcal{D}$, c) totally OOD state-action pairs $(s_{\neg\mathcal{D}}, a)$ with OOD state $s \not\sim \mathcal{D}$ and arbitrary $a \sim \mathcal{A}$.

To ensure the performance of π_{off}, offline algorithms enforce the conservative Q-value on ID state but OOD action pairs $(s_{\mathcal{D}}, a_{\neg\mathcal{D}})$ to prevent selecting the $a_{\neg\mathcal{D}}$ given $s_{\mathcal{D}}$ [2, 4]. Besides, offline datasets $\mathcal{D}$ often have unbalanced state distributions [7]. As a result, offline to online finetuning faces severe **distribution shift** [10] that should be eliminated during online policy improvement.

III. Decoupled Offline to Online Finetuning

Prior offline to online finetuning methods directly finetune π_{off} through online interaction [6, 17]. These approaches aim to eliminate distribution shift while simultaneously improving

[1]We assume the reward function r is known. If not, the reward can be considered as part of the dynamics model $P_{\widehat{\mathcal{M}},r}(s_{t+1}, r_t|s_t, a_t)$. The following theoretical analysis can also be applied to the unknown reward function [12, 13, 15].

policy performance. However, there may be potential conflicts between these two objectives. Unlike previous work, our key insight lies in decoupling the elimination of distribution shift from the policy improvement within the model-based framework. Concretely, our framework first eliminates the distribution shift by online finetuning the dynamics $P_{\widehat{\mathcal{M}}}$ and then finetunes the policy π_{off} in offline manner, without further online interaction.

We first theoretically decouple the offline to online finetuning into two stages and then reveal the relation between online dynamics finetuning and distribution shift elimination. Then we develop our **D**ecoupled **O**ffline to **O**nline **F**inetuning (DOOF) based on offline model-based algorithms. Finally, we discuss the advantages of DOOF, especially on the interaction efficiency.

A. Theoretical Analysis

In the context of model-based approaches, offline to online finetuning should minimize the gap between $J\left(\pi_{\text{off}}, \widehat{\mathcal{M}}\right)$ and $J\left(\pi^*, \mathcal{M}\right)$. This gap encompasses not only a standard RL problem, policy improvement, but also the error introduced by the inaccuracy of the dynamics model $P_{\widehat{\mathcal{M}}}$:

$$J\left(\pi_{\text{off}}, \widehat{\mathcal{M}}\right) - J\left(\pi^*, \mathcal{M}\right) = \tag{5}$$
$$\underbrace{\left[J\left(\pi_{\text{off}}, \widehat{\mathcal{M}}\right) - J\left(\pi_{\text{off}}, \mathcal{M}\right)\right]}_{\text{error of the dynamics model}} + \underbrace{\left[J\left(\pi_{\text{off}}, \mathcal{M}\right) - J\left(\pi^*, \mathcal{M}\right)\right]}_{\text{policy optimization}}.$$

According to the above formulation, our model-based offline to online finetuning algorithm is divided into two phases. First, we aim to reduce the error of the dynamics model that can be viewed as the distributional shift in the context of offline to online finetuning. The relation between this error reduction (also the elimination of the distributional shift) and learning a more accurate $P_{\widehat{\mathcal{M}}}$ during online interaction will be revealed. Secondly, we leverage this refined dynamics model to improve the performance of π_{off} in offline manner without further interaction with the environment.

a) Step I: Elimination of distributional shift: The relation between the first performance difference $J\left(\pi_{\text{off}}, \widehat{\mathcal{M}}\right) - J\left(\pi_{\text{off}}, \mathcal{M}\right)$ and the dynamics distance $d_{\text{TV}}\left(P_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$ can be formulated as follows:

Theorem III.1. *Assume $\mathcal{M}$ and $\widehat{\mathcal{M}}$ are the MDPs with different transition dynamics $P_{\mathcal{M}}$ and $P_{\widehat{\mathcal{M}}}$ but the same reward function r with $|r| \leq r_{\max}$. Then the performance difference $J\left(\pi_{\text{off}}, \widehat{\mathcal{M}}\right) - J\left(\pi_{\text{off}}, \mathcal{M}\right)$ holds with $C_1 = r_{\max} \cdot \gamma / (1-\gamma)$:*

$$\left|J\left(\pi_{\text{off}}, \widehat{\mathcal{M}}\right) - J\left(\pi_{\text{off}}, \mathcal{M}\right)\right| \leq C_1 \mathop{\mathbb{E}}_{(s,a)\sim\rho^{\pi_{\text{off}}}_{\widehat{\mathcal{M}}}} \left[d_{\text{TV}}\left(P_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)\right].$$

$\rho^{\pi_{\text{off}}}_{\widehat{\mathcal{M}}}(s,a) = \pi_{\text{off}}(a|s) \cdot \sum_{t=0}^{T} \gamma^t P\left(s_t = s | \pi_{\text{off}}\right)$ is the discounted unnormalized visitation frequency, where $P\left(s_t = s|\pi_{\text{off}}\right)$ denotes the probability that the t-th state equals s in trajectories generated by policy π_{off} and transition dynamics $P_{\widehat{\mathcal{M}}}$. $d_{\text{TV}}\left(P_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$ is the total variation distance.

Furthermore, the dynamics distance $d_{\text{TV}}\left(P_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$ can be bounded by the state-action frequency:

Algorithm 1 **D**ecoupled **O**ffline to **O**nline **F**inetuning

Input: Offline dataset $\mathcal{D}$; penalty coefficient λ_{off}; exploration coefficient λ_{on}.
(**Offline pretraining**) Obtain the offline pretrained policy π_{off} and offline dynamics model $P_{\widehat{\mathcal{M}}}$;
(**Online finetuning**)
1) Train uncertainty policy π_u via $r_t + \lambda_{\text{on}} \cdot u\left(s_t, a_t\right)$;
2) Collect data with π_u to get the real buffer $\mathcal{D}_{\text{on}}$;
3) Finetune dynamics model $P_{\widehat{\mathcal{M}}}$ on dataset $\mathcal{D} \cup \mathcal{D}_{\text{on}}$;
4) Finetune π_{off} on dataset $\mathcal{D} \cup \mathcal{D}_{\text{on}}$ with finetuned dynamics model $P^*_{\widehat{\mathcal{M}}}$.

Theorem III.2. *Assume $\mathcal{P} = \{P_{\mathcal{M}} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}\}$ and $|\mathcal{P}| < \infty$. Given an exact pair (s,a) exists in $\mathcal{D}$ with $\mathcal{D}_{s,a} = \{s_t, a_t, s_{t+1}\}_{s_t=s, a_t=a}$ and $n(s,a) = |\mathcal{D}_{s,a}|$. For $\delta \in (0,1)$ the dynamics $P_{\widehat{\mathcal{M}}}$ learned by Eq.(3) satisfies*

$$d_{\text{TV}}\left(P_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right) \leq \sqrt{\frac{2\log\left(|\mathcal{P}|/\delta\right)}{n\left(s,a\right)}}, \tag{6}$$

with the probability at least $1-\delta$.

b) Discussion of Theorem III.1 and Theorem III.2: Directly combining the two theorems, we can derive the following inequality with the constant $C_2 = \frac{\gamma \cdot r_{\max}}{1-\gamma}\sqrt{2\log\left(|\mathcal{P}|/\delta\right)}$:

$$\frac{1}{C_2}\left|J\left(\pi_{\text{off}}, \widehat{\mathcal{M}}\right) - J\left(\pi_{\text{off}}, \mathcal{M}\right)\right| \leq \underbrace{\mathop{\mathbb{E}}_{(s_{\mathcal{D}}, a_{\mathcal{D}})}\left[\frac{1}{\sqrt{n\left(s,a\right)}}\right]}_{\text{a) offline dataset}}$$
$$+ \underbrace{\mathop{\mathbb{E}}_{(s_{\mathcal{D}}, a_{\neg\mathcal{D}})}\left[\frac{1}{\sqrt{n\left(s,a\right)}}\right]}_{\text{b) conservative OOD actions}} + \underbrace{\mathop{\mathbb{E}}_{(s_{\neg\mathcal{D}}, a)}\left[\frac{1}{\sqrt{n\left(s,a\right)}}\right]}_{\text{c) unbalanced state distribution}}. \tag{7}$$

On the right-hand side of the inequality, the entire state-action space is divided into three parts based on whether the state or action belongs to the offline dataset $\mathcal{D}$. Obviously, for the estimated dynamics model $P_{\widehat{\mathcal{M}}}$ trained using the offline dataset $\mathcal{D}$, the inequality $n\left(s_{\mathcal{D}}, a_{\mathcal{D}}\right) \gg n\left(s_{\mathcal{D}}, a_{\neg\mathcal{D}}\right) \gg n\left(s_{\neg\mathcal{D}}, a\right)$ holds. This implies that the first term in 7 will be relatively small, while the latter two terms are comparatively larger, especially the third term. During the online interaction, we aim to collect more data to reduce this performance gap caused by the distributional shift. The more efficient approach is to collect b) $(s_{\mathcal{D}}, a_{\neg\mathcal{D}})$ and c) $(s_{\neg\mathcal{D}}, a)$, rather than a) $(s_{\mathcal{D}}, a_{\mathcal{D}})$.

We train a dynamics model on `WK-m` dataset[2] and plot the distribution of the total variation distance across a) $(s_{\mathcal{D}}, a_{\mathcal{D}})$, b) $(s_{\mathcal{D}}, a_{\neg\mathcal{D}})$, c) $(s_{\neg\mathcal{D}}, a)$ in Figure 2. It is evident that the distance satisfies the inequality $d_{\mathcal{F},(s_{\mathcal{D}}, a_{\mathcal{D}})} \ll d_{\mathcal{F},(s_{\mathcal{D}}, a_{\neg\mathcal{D}})} \ll d_{\mathcal{F},(s_{\neg\mathcal{D}}, a)}$, which aligns with our intuitive deductions. The horizontal axis represents the log of distance, indicating the significant distance differences across different state-action

[2]Abbreviations of D4RL datasets are: `halfcheetah` → `HC`, `hopper` → `HP`, `walker2d` → `WK`, `Pen` → `P`,`random` → `r`, `medium` → `m`, `medium-replay` → `mr`, `medium-expert` → `me`, `cloned` → `c`, `human` → `h`.

regions, with even orders of magnitude gaps. This suggests that to reduce the expected dynamics distance, online interaction should minimize the second and third terms.

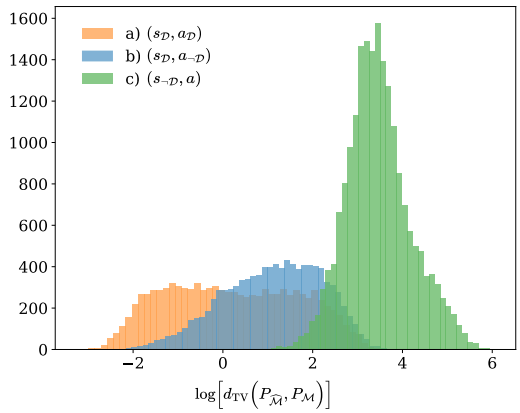

Fig. 2: The distribution of total variation distance on `WK-m` dataset. The log is adopted on the distance.

c) Step II: Offline policy improvement: After online dynamics finetuning, one more accurate dynamics model $P^*_{\widehat{\mathcal{M}}}$ is obtained. Then we finetune the policy π_{off} to get the final policy π^*_{off} in offline manner, the same as the way of training the offline policy π_{off}. Concretely, the offline manner represents optimizing the policy in conservative MDP

$$\widehat{\mathcal{M}}_u = \{\mathcal{S}, \mathcal{A}, r - \lambda_{\text{off}} \cdot u, P_{\widehat{\mathcal{M}}}, d_0, \gamma\}, \tag{8}$$

where $u(s,a) \geq d_{\mathrm{TV}}\left(P_{\widehat{\mathcal{M}}}(s,a), P_{\mathcal{M}}(s,a)\right), \forall s \in \mathcal{S},\ a \in \mathcal{A}$, is the upper bound of the dynamics distance. The performance of policy π obtained from offline policy optimization can be described by the following theorem.

Theorem III.3. *The performance of π, optimized in $\widehat{\mathcal{M}}_u = \{\mathcal{S}, \mathcal{A}, r - \lambda_{\text{off}} \cdot u, P_{\widehat{\mathcal{M}}}, d_0, \gamma\}$, satisfies*

$$J(\pi, \mathcal{M}) \geq J(\pi^*, \mathcal{M}) - 2\lambda_{\text{off}} \cdot \mathop{\mathbb{E}}_{(s,a)\sim\rho^{\pi^*}_{\widehat{\mathcal{M}}}}[u(s,a)]. \tag{9}$$

Here π^* is the optimal policy.

d) Discussion of Theorem III.3: Although the policy is trained under biased dynamics $\hat{\mathcal{M}}$, at test time it is deployed in the true environment $\mathcal{M}$. Consequently, what truly matters is the policy's performance in the real $\mathcal{M}$ rather than $\hat{\mathcal{M}}$. So Theorem 3.3 evaluates the policy's return under $\mathcal{M}$.

Theorem 3.3 reveals that the performance $J(\pi, \mathcal{M})$ is affected by the uncertainty $u(s,a)$, also the distance between the true and estimated dynamics $d_{\mathcal{F}}\left(P_{\widehat{\mathcal{M}}}(s,a), P_{\mathcal{M}}(s,a)\right)$. After online dynamics finetuning, this distance becomes smaller. As a result, the offline finetuning policy π^*_{off} is better $J(\pi^*_{\text{off}}, \mathcal{M}) \geq J(\pi_{\text{off}}, \mathcal{M})$.

B. Practical Implementation

Now we design a practical decoupled model-based offline to online finetuning framework called DOOF motivated by the above analysis, summarized in Alg. 1.

*1) **Offline pretraining**:* We first learn an ensemble dynamics $\left\{P^k_{\widehat{\mathcal{M}}}\right\}_{k=1}^K$ using Eq. (3). All the uncertainty-based offline algorithms are applicable within our decoupled framework, such as MOPO [12], **`count-MORL`** [15] and MOBILE [13]. We choose MOPO to verify DOOF due to its simplicity and universality. Specifically, the uncertainty estimator is the maximum standard deviation of the learned dynamics models $u(s,a) = \max_{k=1}^K \left\|\Sigma^k_\phi(s,a)\right\|_{\mathrm{F}}$. We denote the the policy obtained from offline pretraining as π_{off}.

*2) **Online finetuning**:* According to above analysis, finetuning dynamics model requires data out of the offline dataset distribution, $(s_{\mathcal{D}}, a_{\neg\mathcal{D}})$ and $(s_{\neg\mathcal{D}}, a)$. If we interact with environment via offline pretrained policy π_{off}, the collected data mainly belongs to the distribution $(s_{\mathcal{D}}, a_{\mathcal{D}})$, which contributes little to learn a more accurate dynamics. What we required is a more exploratory and optimistic policy, even if its performance is a little worse than π_{off}.

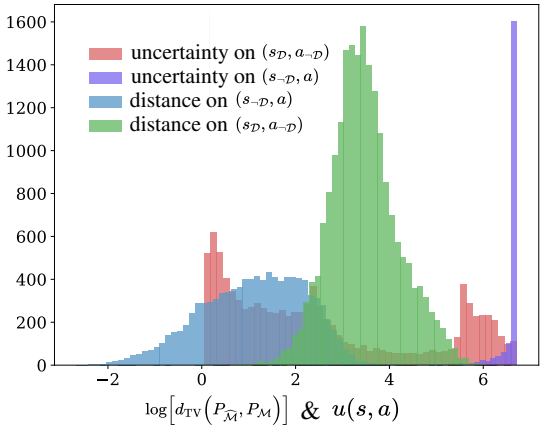

Fig. 3: The distribution of total variation distance and uncertainty on `WK-m` dataset. The reason for the concentration of the uncertainty distribution on the right is the clip operation.

MOPO penalizes OOD state-action pairs using uncertainty $u(s,a)$. In Figure 3, we observe that the state-action region with greater distance $d_{\mathrm{TV}}\left(P_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$ also exhibits higher uncertainty $u(s,a)$. This implies that the uncertainty can assist in identifying which data points are more critical to be collected through online interaction. Therefore, we choose to use the uncertainty as an extrinsic reward to train the policy π_{off}:

$$r^u_t(s_t, a_t) = r_t + \lambda_{\text{on}} \cdot u(s_t, a_t), \tag{10}$$

The exploration coefficient λ_{on} determines the degree of uncertainty guidance. The selection of λ_{on} is proportionally related to the offline penalty coefficient, where the ratio is $\lambda_{\text{on}} : \lambda_{\text{off}} = 0.25, 0.5, 1, 2$. We denote this modified policy as uncertainty policy π_u and, for example, use $\pi_u(0.25)$ to represent the policy is trained with $\lambda_{\text{on}} : \lambda_{\text{off}} = 0.25$. The online buffer $\mathcal{D}_{\text{on}}$ is collected using uncertainty policy π_u through online interaction with environment. And similarly, $\mathcal{D}_{\text{on}}(0.25)$ indicates this online buffer is collected by $\pi_u(0.25)$. Then we finetune the dynamics model on data $\mathcal{D} \cup \mathcal{D}_{\text{on}}$:

$$P^*_{\widehat{\mathcal{M}}} = \arg\min_{P_{\widehat{\mathcal{M}}}} \mathbb{E}_{\mathcal{D}\cup\mathcal{D}_{\text{on}}}\left[-\log P_{\widehat{\mathcal{M}}}(s_{t+1}|s_t, a_t)\right]. \tag{11}$$

Finally, we run MOPO on dataset $\mathcal{D} \cup \mathcal{D}_{\text{on}}$ using the finetuned dynamics model $P^*_{\widehat{\mathcal{M}}}$ to finetune the offline pretrained policy π_{off} without any online interaction.

C. Discussion and Advantages

The online finetuning stage of DOOF is essentially **supervised** dynamics model learning, which is simpler than RL. To recover the true transition dynamics $P_{\widehat{\mathcal{M}}}$ for (s_t, a_t), only data on (s_t, a_t) is required. In contrast, RL inherently relies on a more diverse distribution. When learning the optimal Q-function $Q(s_t, a_t)$, not only data on (s_t, a_t) is needed, but also $(s_{t+1}, a_{t+1}, s_{t+2}, a_{t+2}, \cdots)$. This is why our framework achieves high interaction efficiency. Moreover, the online buffer collected by π_u tends to include state-action pairs with high uncertainty, corresponding to regions where the dynamics is less accurate, which significantly contributes to the distribution shift elimination.

The policy finetuning is conducted in an offline manner without online interaction. If the dynamics is accurate enough, the policy is equivalent to interacting with the true environment, which is the inherent advantage of model-based frameworks. Moreover, the policy training remains conservative, avoiding performance drop.

IV. Related Work

Offline RL methods mainly contains two categories. One is policy constraint, which constrains the learned policy close to behavior policy based on different "closeness" such as batch constrained [18], KL divergence [19], MMD distance [20], MSE constraint [3] and TV distance [4]. The other is value regularization, which regularizes the value function from overestimation on OOD state-action pairs [2]. Besides, Decision Transformer directly maximizes action likelihood, which opens up sequence modeling paradigm. Rein*former* [5] further propose the max-return sequence modeling. Based on the conservatism on different components, offline model-based RL methods derived from MBPO [11] are divided into three categories: MOPO [12] and MOReL [21] propose to penalize the reward function by the uncertainty of the learned dynamics models and MOBILE [13] theoretically conducts uncertainty quantification through the inconsistency of Bellman estimations. COMBO [22] trains a conservative Q-function based on CQL [2]. RAMBO [23] and ARMOR [24] incorporate conservatism by modifying the transition dynamics.

Offline to online finetuning aims to further improve the policy using the offline pretrained policy as initialization, which encompasses distribution shift elimination and policy improvement. The approaches can be divided into two classes. One tries to design less conservative algorithms to reduce the impact of distribution shift during online finetuning [6, 8, 9, 25], while the other applies additional constraints to mitigate the shift [10, 17]. Although MOORe [26], MOTO [27], and FOWM [28] are online finetuning algorithms within the model-based framework, they directly finetune the policy, whereas DOOF decouples dynamics model learning from policy improvement.

V. Experiments

We conduct an extensive evaluation on the decoupled framework using classical offline datasets from D4RL [7]. Our experiments are organized in accordance with the algorithmic workflow:

- Uncertainty policy training: We evaluate the performance of uncertainty policy π_u and illustrate the distribution of the online buffer $\mathcal{D}_{\text{on}}(\pi_u)$. The impact of exploration coefficient λ_{on} on finetuning dynamics and policy is discussed in latter sections.
- Online dynamics finetuning: We focus on the TV distance between the finetuned dynamics model $P^*_{\widehat{\mathcal{M}}}$ and the actual environment, and the distance change before and after finetuning.
- Offline policy finetuning: We plot the typical training curves to analyze the characteristics of DOOF and the impact of λ_{on}. We also demonstrate the exceptional interactive efficiency.

A. Uncertainty policy training

1) ***Performance Comparison****:* The uncertainty policy π_u determines the distribution of collected online buffer $\mathcal{D}_{\text{on}}(\pi_u)$, which directly affects the online dynamics finetuning. When training π_u, we adopt four exploration coefficients λ_{on} with ratio $\lambda_{\text{on}} : \lambda_{\text{off}} = 0.25, 0.5, 1, 2$. We compare the performance between the offline pretrained π_{off} and π_u.

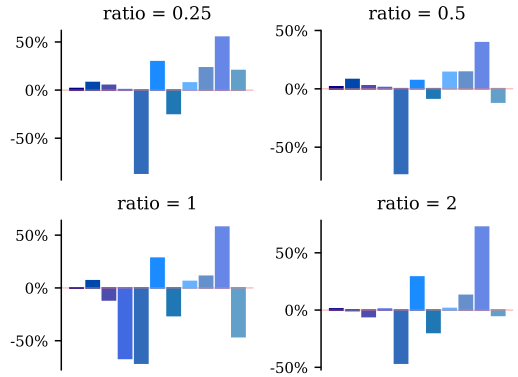

Fig. 5: Performance percentage change of the uncertainty policy π_u over the offline pretrained π_{off}, with different exploration coefficients $\lambda_{\text{on}} : \lambda_{\text{off}} = 0.25, 0.5, 1, 2$ on datasets `HC-r`, `HC-m`, `HC-mr`, `HP-r`, `HP-m`, `HP-mr`, `WK-r`, `WK-m`, `WK-mr`, `P-c`, `P-h`.

In Figure 5, we illustrate the percentage of performance change $\left(\frac{J(\pi_u)}{J(\pi_{\text{off}})} - 1\right) \times 100\%$ on ratio $0.25, 0.5, 1, 2$. Some datasets, such as `HP-m`, exhibits a noticeable performance decline. This is attributed to the uncertainty policy training that alters the inherent conservatism. But surprisingly, the `P-c` achieves performance improvement through simple uncertainty training. In most cases, uncertainty policy training yields stochastic gains, indicating that further policy fine-tuning is still indispensable.

2) ***Distribution Visualization****:* Next, we evaluate the distribution of online buffer $\mathcal{D}_{\text{on}}(\pi_u)$ collected by the uncertainty policy π_u. Ideally, the collected data should consist of state-action regions unseen by the offline pretrained dynamics $P_{\widehat{\mathcal{M}}}$,

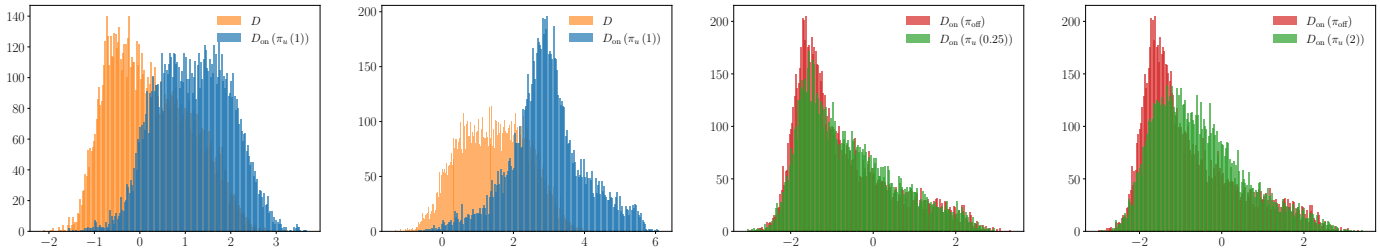

Fig. 4: The left two figures show the distribution of $\log\left[d_{\text{TV}}\left(P_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)\right]$ on the offline dataset $\mathcal{D}$ and the online buffer $\mathcal{D}_{\text{on}}$ collected by the uncertainty policy $\pi_u(1)$ with $\lambda_{\text{on}} = \lambda_{\text{off}}$, across `HC-r` and `WK-r`. The third figure depicts the distance distribution on online buffer collected by the offline pretrained policy π_{off} and uncertainty policy $\pi_u(0.25)$ with $\lambda_{\text{on}} = 0.25\lambda_{\text{off}}$ on `WK-mr`. The last one shows a similar setting, but with $\pi_u(0.25)$ on `WK-mr`.

that is, the distance $d_{\text{TV}}\left(P_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$ on online buffer $\mathcal{D}_{\text{on}}(\pi_u)$ should be larger. In the left two Figure 4, the distance over $\mathcal{D}$ is indeed minimal, while that over $\mathcal{D}_{\text{on}}(\pi_u)$ is substantially larger. This indicates that the dynamics training using $\mathcal{D}_{\text{on}}(\pi_u)$ can effectively mitigate distribution shift.

In the right two figures from Figure 4, the distribution on online buffer collected by pretrained policy π_{off} and the uncertainty π_u are almost overlap. This is because the π_u is finetuned based on π_{off}. But these subtle differences actually have a significant impact on the performance of offline policy finetuning, and we will further discuss this in Section V-C.

B. Online dynamics finetuning

1) ***Distance Comparison****:* Online dynamics finetuning aims to eliminate distribution shift by learning a more accurate dynamics model $P^*_{\widehat{\mathcal{M}}}$. The total variation distance between the dynamics model and the true environment on the uniform state-action distribution, summarized in Table I, can reflect the distribution shift. From the Table I, we observe that the online dynamics finetuning significantly reduces the distance $d_{\text{TV}}\left(P_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$ between the real environment and the uniform state-action distribution, except for `HP-r` with severely biased distribution. Moreover, dynamics finetuning demonstrates robustness with respect to the exploration coefficient λ_{on}.

2) ***Distribution Visualization****:* In Figure 6, we observe that distance distribution $d_{\text{TV}}\left(P_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$, where $P_{\widehat{\mathcal{M}}}$ is pretrained on `HC-r`, exhibit long-tail effect on `HC-me`. This may be attributed to that the dynamics model learned from low-quality datasets is blind on the high-return regions. After finetuning, not only the distance has been significantly reduced, but the long-tail is also mitigated.

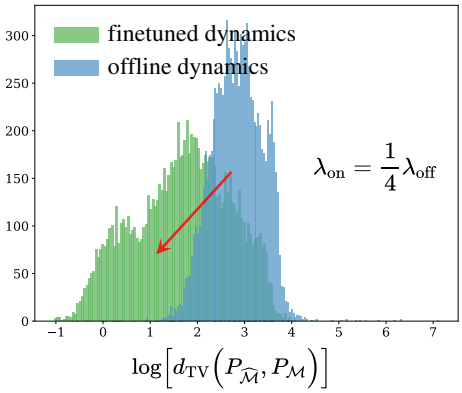

Fig. 6: This figure presents the distance distribution of the dynamics model (trained by `HC-r`) on dataset `HC-me` before and after online dynamics finetuning. The dynamics model is finetuned using online buffer $\mathcal{D}_{\text{on}}(\pi_u(0.25))$.

C. Offline policy finetuning

1) ***Finetuning Performance****:* We first demonstrate the superior finetuning performance and exceptional data efficiency of our framework DOOF. All the previous offline-to-online algorithms directly finetune the policy during the online interaction, hence they plot the return curves during online training for comparison. DOOF decouples the online interaction and policy finetuning, so the performance maintains the same

TABLE I: The total variation distance between the true and estimated dynamics on state-action pairs uniformly sampled from the true MDP. $d_{\text{TV}}\left(P_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$ represents the distance before online dynamics finetuning, also the offline pretrained dynamics model. $d^{(\text{ratio})}_{\text{TV}}\left(P^*_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$ is the distance after finetuning, using the online buffer collected by the π_u trained with $\lambda_{\text{on}} : \lambda_{\text{off}} = 0.25, 0.5, 1, 2$.

Datasets	$d_{\text{TV}}\left(P_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$		$d^{(0.25)}_{\text{TV}}\left(P^*_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$		$d^{(0.5)}_{\text{TV}}\left(P^*_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$		$d^{(1)}_{\text{TV}}\left(P^*_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$		$d^{(2)}_{\text{TV}}\left(P^*_{\widehat{\mathcal{M}}}, P_{\mathcal{M}}\right)$	
	mean	max	mean	max	mean	max	mean	max	mean	max
`HC-r`	118.123	1223.240	119.155	1294.893	**109.950**	**1170.111**	120.915	**2**076.364	119.825	1456.900
`HC-m`	82.339	910.501	**63.708**	969.096	**67.705**	1169.591	**67.813**	1029.322	**67.565**	1116.883
`HC-mr`	121.772	2445.304	**93.484**	**1951.653**	**78.373**	**1500.777**	**89.847**	**1894.1079**	**89.317**	**1641.784**
`HP-r`	39.970	319.989	44.888	425.438	56.680	547.203	56.527	629.646	56.668	672.624
`HP-m`	874.378	5567.265	1156.229	6678.457	950.298	**5421.856**	**766.162**	5719.151	**765.820**	5757.906
`HP-mr`	135.835	824.546	**116.199**	864.601	**112.370**	**688.462**	**105.852**	1401.446	**105.176**	1177.113
`WK-r`	211.203	2073.774	**118.911**	**1149.800**	**92.696**	**793.019**	**145.021**	**1156.193**	**144.939**	**1234.787**
`WK-m`	173.784	1018.120	**156.550**	**973.955**	**157.137**	**909.678**	**149.325**	**988.611**	**148.906**	**997.185**
`WK-mr`	164.754	1505.594	**137.658**	**892.306**	**148.490**	**910.916**	**131.005**	**1104.575**	**130.517**	1765.076

TABLE II: The online (ON-10K) and offline (OFF-300K) finetuning results over 3 seeds. The gray results means the performance has declined after finetuning and the **bold** results represent the best.

	DOOF		IQL			CaL-QL			CQL		
	offline	OFF-300K	*offline*	ON-10K	OFF-300K	*offline*	ON-10K	OFF-300K	*offline*	ON-10K	OFF-300K
HC-r	37.5	**54.9(+17.4)**	14.9	14.2(−0.6)	8.9(−6.0)	25.1	11.2(−14.0)	2.3(−22.9)	23.7	12.3(−11.4)	2.2(−21.5)
HC-m	70.5	**92.5(+22.0)**	48.5	48.3(−0.2)	49.2(+0.8)	47.6	48.6(+ 1.0)	49.1(+ 1.5)	46.6	46.9(+ 0.2)	47.7(+ 1.1)
HC-mr	69.2	**88.3(+19.1)**	44.3	44.2(−0.0)	45.3(+1.0)	46.5	47.4(+ 0.9)	47.4(+ 1.0)	44.9	45.1(+ 0.2)	46.0(+ 1.2)
HP-r	32.0	32.4(+ 0.5)	7.6	7.6(−0.0)	8.17(+0.6)	7.3	4.7(− 2.6)	2.0(− 5.3)	7.8	6.9(− 1.0)	8.6(+ 0.8)
HP-m	68.3	**108.4(+40.1)**	57.3	58.8(+1.4)	59.0(+1.6)	56.8	72.7(+15.9)	73.5(+16.7)	62.3	65.4(+ 3.1)	63.0(+ 0.7)
HP-mr	82.6	**107.3(+24.7)**	97.2	95.7(−1.5)	99.4(+2.2)	97.7	97.3(− 0.5)	97.7(− 0.1)	94.1	95.0(+ 0.9)	99.7(+ 5.7)
WK-r	4.1	**16.9(+12.8)**	3.7	3.7(+0.0)	4.2(+0.6)	3.9	1.5(− 2.4)	0.2(− 3.7)	−0.3	−0.2(+ 0.0)	1.6(+ 1.9)
WK-m	77.7	**93.8(+16.1)**	83.2	80.8(−2.4)	79.0(−4.2)	83.6	83.6(+ 0.0)	83.3(− 0.3)	81.3	83.6(+ 2.3)	80.5(− 0.8)
WK-mr	69.6	**97.8(+28.2)**	77.1	78.1(+1.0)	84.2(+7.1)	79.4	87.2(+ 7.8)	89.2(+ 9.8)	72.1	82.2(+10.2)	87.6(+15.5)
P-c	48.9	54.2(+ 5.3)	69.5	67.9(−1.6)	68.9(−0.6)	−3.5	−4.0(− 0.6)	−4.8(− 1.3)	−3.4	−3.8(− 0.4)	−4.8(− 1.4)
P-h	38.7	**62.0(+23.3)**	56.4	57.6(+1.2)	60.2(+3.8)	7.7	−4.0(−11.6)	−3.4(−11.1)	−3.4	−3.5(− 0.2)	−4.4(− 1.1)

	DOOF		SPOT			PEX			FOWM		
	offline	OFF-300K	*offline*	ON-10K	OFF-300K	*offline*	ON-10K	OFF-300K	*offline*	ON-10K	OFF-300K
HC-r	37.5	**54.9(+17.4)**	8.0	7.8(− 0.2)	5.1(− 2.8)	15.7	16.4(+ 0.7)	8.6(− 7.1)	15.4	20.4(+ 5.0)	20.3(+ 4.9)
HC-m	70.5	**92.5(+22.0)**	45.4	46.2(+ 0.8)	45.1(− 0.3)	48.3	49.5(+ 1.2)	46.0(− 2.3)	44.0	45.4(+ 1.4)	42.9(− 1.1)
HC-mr	69.2	**88.3(+19.1)**	43.8	43.1(− 0.7)	43.2(− 0.6)	44.7	45.7(+ 1.0)	44.2(− 0.5)	47.3	49.8(+ 2.5)	49.7(+ 2.3)
HP-r	31.9	32.4(+ 0.5)	8.5	9.9(+ 1.4)	**32.3(+23.8)**	8.4	7.7(− 0.7)	7.8(− 0.6)	9.2	9.5(+ 0.3)	9.5(+ 0.3)
HP-m	68.3	**108.4(+40.1)**	57.8	59.0(+ 1.2)	60.9(+ 3.1)	59.2	19.4(−39.8)	56.6(− 2.6)	48.5	64.5(+16.0)	52.5(+ 4.0)
HP-mr	82.6	**107.3(+24.7)**	74.2	47.9(−26.4)	85.5(+11.2)	78.3	102.3(+24.0)	73.1(− 5.2)	93.0	88.0(− 5.0)	99.3(+ 6.3)
WK-r	4.1	**16.9(+12.8)**	1.5	6.1(+ 4.6)	5.7(+ 4.2)	8.8	8.6(− 0.2)	9.1(+ 0.3)	4.2	5.6(+ 1.4)	5.5(+ 1.2)
WK-m	77.7	**93.8(+16.1)**	81.7	81.0(− 0.8)	82.5(+ 0.8)	72.1	56.8(−15.3)	81.9(+ 9.8)	0.6	40.0(+39.4)	27.3(+26.8)
WK-mr	69.6	**97.8(+28.2)**	81.1	81.0(− 0.0)	81.8(+ 0.8)	71.3	61.8(− 9.5)	59.7(−11.6)	36.0	54.7(+18.7)	45.3(+ 9.3)
P-c	48.9	54.2(+ 5.3)	2.5	15.5(+13.0)	22.5(+20.0)	33.8	29.8(− 4.0)	46.7(+12.9)	−2.8	51.0(+53.8)	**57.3(+60.1)**
P-h	38.7	**62.0(+23.3)**	25.9	13.7(−12.2)	18.2(− 7.6)	50.1	59.0(+ 8.9)	70.6(+20.5)	−0.0	1.9(+ 1.9)	8.3(+ 8.3)

during online finetuning, which makes the return curves comparison method inapplicable. As a result, we compare our proposed DOOF with representative baselines using the following approaches:

- DOOF first performs 10k online interaction steps with true environment to finetune the dynamics model, and then uses this online finetuned dynamics as estimated environment to offline finetune the policy with 300k steps (OFF-300K);
- For baselines, including model-free IQL [25], Cal-QL [6], CQL [2], SPOT [9], PEX [29] and model-based FOWM [28], they first finetune the policy using 10k online interaction steps (ON-10K) and then finetune the policy 300k gradient steps without online interaction (OFF-300K).

DOOF achieves significant performance gains, approaching and even achieving optimal policy levels. What's more notable is the interaction efficiency; we have achieved remarkable effects with only 10k online interaction steps while the performance of some baselines even drop. In addition to the model-based approach, the decoupled framework and the uncertainty guided data collection to quickly eliminate distribution shift are also crucial.

*2) **Training Curves**:* We further plot several representative curves to illustrate the stable training performance and the relationship between performance and the dynamics distance.

1) In left side of Figure 7, if we use the offline pretrained policy π_{off} to collect the online buffer (the blue curve), the offline finetuning performance may decline. While the performance with uncertainty policy $\pi_u(2)$ is best (the purple curve). This suggests that uncertainty-guided exploration $\pi_u(2)$ is more conducive to collecting data useful for online dynamics finetuning compared with the uncertainty-free policy π_{off}.

2) On the middle figure from Figure 7, the stability of the offline finetuning curves obtained by different data collection policy, π_{off} or π_u, varies. The offline pretrained policy π_{off} is relatively poor (the blue bold curve), while others with uncertainty policy are more stable (for example, the orange curve $\pi_u(0.25)$ is the most stable). This indicates the distribution difference in Figure 4 affects the stability heavily. Besides, this phenomenon also underscores the importance of uncertainty policy.

3) In right side of Figure 7, we conduct twice finetuning process. Dynamics model $P^*_{\widehat{\mathcal{M}}}$ is obtained from the first online dynamics finetuning while $P^{**}_{\widehat{\mathcal{M}}}$ is further finetuned based on $P^*_{\widehat{\mathcal{M}}}$. The distance distributions of these two dynamics models between the true environment are shown in the lower right corner in Figure 7. After the first finetuning, the distance increases, which means this finetuning is a failure that corresponds to a performance decline and severe fluctuations. After the second finetuning, the distance becomes lower than the $P_{\widehat{\mathcal{M}}}$, resulting in a noticeable performance improvement.

VI. Conclusion and Future Work

In this work, we propose a **D**ecoupled model-based **O**ffline to **O**nline **F**inetuning framework called **DOOF**. DOOF decouples the distribution shift elimination from the policy optimization in offline to online finetuning. Specifically, DOOF finetunes the dynamics model during online interaction to eliminate the distribution shift using the online buffer collected by the uncertainty policy. And then the policy is finetuned with the help of the finetuned dynamics in offline manner without further interaction. This decoupled framework not only ensures superior performance but also boasts interaction efficiency.

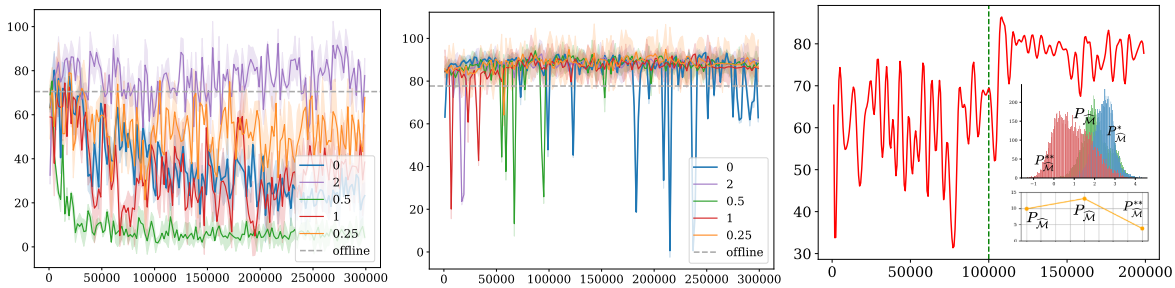

Fig. 7: These figures illustrate the performance curves during policy finetuning on `HC-m` (left) and `WK-m` (middle) with varying values of λ_{on}. The horizontal axis denotes the training steps, not the interaction steps. The blue curve with label '0' is based on the offline pretrained policy π_{off} rather than π_u. The right figure illustrates the performance curve of two rounds of online finetuning. The two smaller plots in the lower right corner show the distribution of distances and the mean distance.

VII. Acknowledgements

This work was supported by the Brain Science and Brain-like Intelligence Technology — National Science and Technology Major Project (Grant No. 2022ZD0208800)

References

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

[2] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1179–1191, 2020.

[3] S. Fujimoto and S. S. Gu, "A minimalist approach to offline reinforcement learning," *Advances in neural information processing systems*, vol. 34, pp. 20 132–20 145, 2021.

[4] Z. Zhuang, K. Lei, J. Liu, D. Wang, and Y. Guo, "Behavior proximal policy optimization," *arXiv preprint arXiv:2302.11312*.

[5] Z. Zhuang, D. Peng, Z. Zhang, D. Wang *et al.*, "Reinformer: Max-return sequence modeling for offline rl," *arXiv preprint arXiv:2405.08740*, 2024.

[6] M. Nakamoto, S. Zhai, A. Singh, M. Sobol Mark, Y. Ma, C. Finn, A. Kumar, and S. Levine, "Cal-ql: Calibrated offline rl pretraining for efficient online fine-tuning," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[7] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4rl: Datasets for deep data-driven reinforcement learning," *arXiv preprint arXiv:2004.07219*, 2020.

[8] J. Lyu, X. Ma, X. Li, and Z. Lu, "Mildly conservative q-learning for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 1711–1724, 2022.

[9] J. Wu, H. Wu, Z. Qiu, J. Wang, and M. Long, "Supported policy optimization for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 31 278–31 291, 2022.

[10] S. Lee, Y. Seo, K. Lee, P. Abbeel, and J. Shin, "Offline-to-online reinforcement learning via balanced replay and pessimistic q-ensemble," in *Conference on Robot Learning*. PMLR, 2022, pp. 1702–1712.

[11] M. Janner, J. Fu, M. Zhang, and S. Levine, "When to trust your model: Model-based policy optimization," *Advances in neural information processing systems*, vol. 32, 2019.

[12] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Y. Zou, S. Levine, C. Finn, and T. Ma, "Mopo: Model-based offline policy optimization," *Advances in Neural Information Processing Systems*, vol. 33, pp. 14 129–14 142, 2020.

[13] Y. Sun, J. Zhang, C. Jia, H. Lin, J. Ye, and Y. Yu, "Model-bellman inconsistency for model-based offline reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2023, pp. 33 177–33 194.

[14] R. S. Sutton, A. G. Barto *et al.*, "Introduction to reinforcement learning," 1998.

[15] B. Kim and M.-h. Oh, "Model-based offline reinforcement learning with count-based conservatism," in *International Conference on Machine Learning*. PMLR, 2023, pp. 16 728–16 746.

[16] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[17] A. Nair, A. Gupta, M. Dalal, and S. Levine, "Awac: Accelerating online reinforcement learning with offline datasets," *arXiv preprint arXiv:2006.09359*, 2020.

[18] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *International conference on machine learning*. PMLR, 2019, pp. 2052–2062.

[19] Y. Wu, G. Tucker, and O. Nachum, "Behavior regularized offline reinforcement learning," *arXiv preprint arXiv:1911.11361*, 2019.

[20] A. Kumar, J. Fu, M. Soh, G. Tucker, and S. Levine, "Stabilizing off-policy q-learning via bootstrapping error reduction," *Advances in Neural Information Processing Systems*, vol. 32.

[21] R. Kidambi, A. Rajeswaran, P. Netrapalli, and T. Joachims, "Morel: Model-based offline reinforcement learning," *Advances in neural information processing systems*, vol. 33, pp. 21 810–21 823, 2020.

[22] T. Yu, A. Kumar, R. Rafailov, A. Rajeswaran, S. Levine, and C. Finn, "Combo: Conservative offline model-based policy optimization," *Advances in neural information processing systems*, vol. 34, pp. 28 954–28 967, 2021.

[23] M. Rigter, B. Lacerda, and N. Hawes, "Rambo-rl: Robust adversarial model-based offline reinforcement learning," *Advances in neural information processing systems*, vol. 35, pp. 16 082–16 097, 2022.

[24] M. Bhardwaj, T. Xie, B. Boots, N. Jiang, and C.-A. Cheng, "Adversarial model for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[25] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit q-learning," *arXiv preprint arXiv:2110.06169*.

[26] Y. Mao, C. Wang, B. Wang, and C. Zhang, "Moore: Model-based offline-to-online reinforcement learning," *arXiv preprint arXiv:2201.10070*, 2022.

[27] R. Rafailov, K. B. Hatch, V. Kolev, J. D. Martin, M. Phielipp, and C. Finn, "Moto: Offline pre-training to online fine-tuning for model-based robot learning," in *Conference on Robot Learning*. PMLR, 2023, pp. 3654–3671.

[28] Y. Feng, N. Hansen, Z. Xiong, C. Rajagopalan, and X. Wang, "Finetuning offline world models in the real world," in *Conference on Robot Learning*. PMLR, 2023, pp. 425–445.

[29] H. Zhang, W. Xu, and H. Yu, "Policy expansion for bridging offline-to-online reinforcement learning," *arXiv preprint arXiv:2302.00935*, 2023.