Journal of Machine Learning Research 24 (2023) 1-35 Submitted 5/22; Revised 4/23; Published 6/23

Adaptation Augmented Model-based Policy Optimization

Jian Shen[†*] ROCKYSHEN@APEX.SJTU.EDU.CN
Hang Lai[†*] LAIHANG99@SJTU.EDU.CN
Minghuan Liu[†] MINGHUANLIU@SJTU.EDU.CN
Han Zhao[‡] HANZHAO@ILLINOIS.EDU
Yong Yu[†] YYU@SJTU.EDU.CN
Weinan Zhang[†⋆] WNZHANG@SJTU.EDU.CN
[†]*Department of Computer Science, Shanghai Jiao Tong University*
[‡]*Department of Computer Science, University of Illinois, Urbana-Champaign*

Editor: Laurent Orseau

Abstract

Compared to model-free reinforcement learning (RL), model-based RL is often more sample efficient by leveraging a learned dynamics model to help decision making. However, the learned model is usually not perfectly accurate and the error will compound in multi-step predictions, which can lead to poor asymptotic performance. In this paper, we first derive an upper bound of the return discrepancy between the real dynamics and the learned model, which reveals the fundamental problem of distribution shift between simulated data and real data. Inspired by the theoretical analysis, we propose an adaptation augmented model-based policy optimization (AMPO) framework to address the distribution shift problem from the perspectives of feature learning and instance re-weighting, respectively. Specifically, the feature-based variant, namely FAMPO, introduces unsupervised model adaptation to minimize the integral probability metric (IPM) between feature distributions from real and simulated data, while the instance-based variant, termed as IAMPO, utilizes importance sampling to re-weight the real samples used to train the model. Besides model learning, we also investigate how to improve policy optimization in the model usage phase by selecting simulated samples with different probability according to their uncertainty. Extensive experiments on challenging continuous control tasks show that FAMPO and IAMPO, coupled with our model usage technique, achieves superior performance against baselines, which demonstrates the effectiveness of the proposed methods.

Keywords: Model-based reinforcement learning, distribution shift, occupancy measure, Integral Probability Metric, importance sampling

1. Introduction

Reinforcement learning (RL) algorithms can be roughly divided into two categories according to whether they utilize an environmental dynamics model: model-free RL (MFRL) and model-based RL (MBRL). MFRL methods, which directly learn a value function or a policy (or both), have achieved great success on a wide range of tasks such as video games (Mnih

∗. Equal contribution.
⋆. Corresponding author.

et al., 2015; Hessel et al., 2018) and robotic control (Gu et al., 2017; Haarnoja et al., 2018). However, MFRL is notoriously sample-inefficient and requires a tremendous number of interactive samples to learn a good policy. In many high-stakes real-world applications, *e.g.*, autonomous driving, online education, etc., it is often expensive, or even infeasible, to collect such large-scale datasets. In contrast, MBRL methods, which learn a dynamics model first and use it to alleviate sampling cost, are widely considered to be an appealing alternative (Sun et al., 2018; Langlois et al., 2019).

Despite the higher sample efficiency, model-based methods tend to have poor asymptotic performance compared to their model-free counterparts due to the vulnerability to model errors (Nagabandi et al., 2018; Chua et al., 2018). To be precise, despite being equipped with a high-capacity model, such errors still exist due to the potential *distribution shift* between the training and generating phases, *i.e.*, the state-action input distribution used to train the model is different from the one generated by the model (Talvitie, 2014). Specifically, the training data is usually collected by the behavior policies in the real environment but the model is required to make predictions on the data collected by the target policy in the model. When an imperfect model is used for multi-step rollouts, the error in one-step prediction is inclined to accumulate in the next steps, also known as the multi-step compounding error challenge (Asadi et al., 2018).

In light of the distribution shift problem in MBRL, in the literature there are many efforts devoted to tackle this problem by improving the approximation accuracy of model learning, or by designing careful strategies when using the model for simulation. For model learning, different architectures (Asadi et al., 2018, 2019; Chua et al., 2018) and loss functions (Farahmand, 2018; Wu et al., 2019) have been proposed to mitigate overfitting or to improve multi-step predictions so that the distributions of the simulated data generated by the model are closer to the real ones. Besides, Talvitie (2014, 2017) also proposed a *self-correct* mechanism that uses the predicted states as the input to train the model along with the real data to gradually bridge the gap between the simulated data and the real ones. On the other hand, for model usage, delicate rollout schemes (Janner et al., 2019; Pan et al., 2020) have been adopted to stop the model rollouts before the simulated data deviate too much from the real distribution. Although these existing methods help alleviate the distribution shift to some extent, this problem has not been explicitly addressed.

In this paper, we investigate how to better address the distribution shift problem explicitly in a principled way. To begin with, we first derive an upper bound of the return disparity between the real dynamics and the learned model, which naturally inspires a bound minimization algorithm. To this end, we propose our AMPO (Adaptation augmented Model-based Policy Optimization) framework upon the existing MBPO (Janner et al., 2019) method with two variants, dubbed as FAMPO and IAMPO, respectively. More specifically, FAMPO introduces a model adaptation procedure which encourages the model to learn invariant feature representations by minimizing the integral probability metric (IPM) between the feature distributions of real data and simulated data. In addition to aligning the feature distributions, we also try to handle the distribution shift problem on the data level and propose an instance-based model adaptation method, IAMPO. The intuition behind IAMPO is that the model should focus more on the state-action pairs that are more likely to appear in the simulated data distributions. The design principle of IAMPO is simple and straightforward: optimizing the dynamics model by minimizing the return discrepancy via gradient descent. In practice, IAMPO re-weights all the training samples by the importance scores when learning the dynamics model, where the importance score is defined as the density ratio between the occupancy measures of the simulated data and the real data.

While model adaptation helps to achieve better generalization, some inaccurate predictions can still catastrophically affect the performance due to the imperfect approximation. For this reason, during the model usage phase, we adopt a weighted sampling strategy, which samples the simulated data with different probabilities according to the model uncertainty, to reduce the proportion of uncertain samples in the simulated data for policy optimization. We evaluate our algorithms on a range of continuous control benchmark tasks, which demonstrate that FAMPO and IAMPO, coupled with the weighted sampling strategy, achieves higher sample efficiency and better asymptotic performance compared to various baselines.

2. Preliminaries

We first introduce the notation used throughout the paper and the problem setup of RL. Then, we briefly discuss the concepts related to integral probability metric. Finally, we present a classical MBRL method, MBPO, which will be the underlying framework for our algorithm.

2.1 Reinforcement Learning

We consider an infinite-horizon Markov Decision Process (MDP), which is defined by the tuple $(\mathcal{S}, \mathcal{A}, T, r, \nu_0, \gamma)$, where $\mathcal{S}$ is the state space and $\mathcal{A}$ is the action space. Throughout the paper, we assume the state and action spaces are continuous. We use $\gamma \in (0,1)$ to denote the discount factor, and $T(s'|s,a)$ to mean the transition density of state s' given action a made under state s. The initial state distribution is denoted as ν_0, and the reward function is denoted as $r(s,a)$. We assume the reward function is bounded by $r_{\max} := \sup_{s,a} |r(s,a)| < \infty$. The agent maintains a policy $\pi(a|s)$ that determines the probability of choosing an action a at a given state s. The goal in reinforcement learning (RL) is to find the optimal policy π^* that maximizes the expected return (sum of discounted rewards), denoted by η_T:

$$\pi^* := \arg\max_{\pi} \eta_T(\pi) = \arg\max_{\pi} \mathbb{E}_{\pi}\Big[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 \sim \nu_0\Big], \tag{1}$$

where $s_{t+1} \sim T(s|s_t, a_t)$ and $a_t \sim \pi(a|s_t)$. In general the true transition $T(s'|s,a)$ is unknown and MBRL methods often learn an approximate model $\hat{T}(s'|s,a)$ of the transition dynamics, using samples collected from interactions with the MDP. Different from the previous works (Luo et al., 2018; Chua et al., 2018), in this paper we also assume the reward function $r(s,a)$ to be unknown, and an agent needs to learn the reward function simultaneously.

Given a policy π and a transition function T, we denote the density of being in state s at time step t as $P^{\pi}_{T,t}(s) = P(s_t = s \mid \pi, T, s_0 \sim \nu_0)$. We then define the normalized occupancy measure (Ho and Ermon, 2016) of policy π under the dynamics T as

$$\rho^{\pi}_T(s,a) = (1-\gamma) \cdot \pi(a|s) \sum_{t=0}^{\infty} \gamma^t P^{\pi}_{T,t}(s).$$

Similarly, $\rho^{\pi}_{\hat{T}}(s,a)$ represents the normalized occupancy measure of policy π under the approximate dynamics model $\hat{T}$.

2.2 Integral Probability Metric

Integral probability metric (IPM) is a family of discrepancy measures between two distributions over the same space (Müller, 1997; Sriperumbudur et al., 2009). Specifically, given two probability distributions $\mathbb{P}$ and $\mathbb{Q}$ over $\mathcal{X}$, the $\mathcal{F}$-IPM is defined as

$$d_{\mathcal{F}}(\mathbb{P},\mathbb{Q}) := \sup_{f\in\mathcal{F}} \ \mathbb{E}_{x\sim\mathbb{P}}[f(x)] - \mathbb{E}_{x\sim\mathbb{Q}}[f(x)], \tag{2}$$

where $\mathcal{F}$ is a class of witness functions $f : \mathcal{X} \to \mathbb{R}$. Following Bińkowski et al. (2018), we assume IPM is symmetric. That is, if $f \in \mathcal{F}$, we also have $-f \in \mathcal{F}$. By choosing different function class $\mathcal{F}$, IPM reduces to many well-known distance metrics between probability distributions. In particular, the Wasserstein-1 distance (Villani, 2008) is defined using the 1-Lipschitz functions $\{f : \|f\|_L \le 1\}$, where the Lipschitz semi-norm $\|\cdot\|_L$ is defined as $\|f\|_L = \sup_{x\neq y} |f(x) - f(y)|/|x - y|$. Furthermore, total variation is also a kind of IPM and we use $d_{\mathrm{TV}}(\cdot,\cdot)$ to denote it.

2.3 Model-based Policy Optimization

We briefly summarize the model-based policy optimization (MBPO) (Janner et al., 2019) algorithm, on top of which we build our algorithm. MBPO uses a bootstrapped ensemble of probabilistic dynamics models $\hat{T}_\theta(s'|s,a)$, parameterized by θ. Each individual dynamics model is a probabilistic neural network which outputs a Gaussian distribution with diagonal covariance. The model ensemble is trained on the real data via minimizing the negative log-likelihood loss:

$$\mathcal{L}^i_{\hat{T}}(\theta) = \sum_{n=1}^{N} \left[\mu^i_\theta(s_n,a_n) - s_{n+1}\right]^\top {\Sigma^i_\theta}^{-1}(s_n,a_n)\left[\mu^i_\theta(s_n,a_n) - s_{n+1}\right] + \log\det\Sigma^i_\theta(s_n,a_n)\,. \tag{3}$$

The learned model $\hat{T}_\theta(s'|s,a)$ is used to generate k-step rollouts starting from the states sampled from the real data buffer $\mathcal{D}_{\mathrm{env}}$ with the actions taken by the current policy π_ϕ. The generated simulated data is then added to a separate buffer $\mathcal{D}_{\mathrm{model}}$. Finally, the policy π_ϕ is trained on both real and simulated data from $\mathcal{D}_{\mathrm{env}} \cup \mathcal{D}_{\mathrm{model}}$ in a fixed ratio using Soft Actor-Critic (SAC) (Haarnoja et al., 2018), which trains a stochastic policy with entropy regularization in actor-critic architecture by minimizing the expected KL-divergence:

$$\mathcal{L}_\pi(\phi) = \mathbb{E}_s[D_{\mathrm{KL}}(\pi_\phi(\cdot|s) \parallel \exp(Q(s,\cdot) - V(s))], \tag{4}$$

where the Q and V functions are estimated via the soft Bellman backup operator following Haarnoja et al. (2018).

3. Feature-based Adaptation-augmented MBPO

In this section, we first propose a feature-based adaptation approach to explicitly mitigate the distribution shift problem in MBRL, which is inspired by the return discrepancy upper bound derived as follows.

3.1 An Upper Bound for Return Discrepancy

One of the main benefits of model-based RL methods is that we can use it to simulate data to replace the real environment once the model is learned. Imagine that if the dynamics model is perfect, we do not need the real environment anymore. However, if the dynamics model is extremely erroneous, the policy learned on the model may perform worse in the real environment, which can lower sample efficiency instead. Therefore, it is necessary in MBRL to derive an upper bound of the discrepancy between the expected return in the real environment $\eta_T(\pi)$ and the expected return in the model $\eta_{\hat{T}}(\pi)$ with the same policy π in the following form (Luo et al., 2018; Janner et al., 2019):

$$\eta_{\hat{T}}(\pi) - \eta_T(\pi) \leq C. \tag{5}$$

Usually, the dynamics model $\hat{T}$ will be learned from experiences (s, a, s') collected by a behavioral policy π_D in the real environment dynamics T. Typically, in an online MBRL method with iterative policy optimization, the behavioral policy π_D represents a collection of past policies. Once we have derived this bound, we can naturally design a model-based framework to optimize the RL objective by maximizing the surrogate return $\eta_{\hat{T}}(\pi)$ and minimizing the discrepancy C simultaneously. Specifically, previous works have derived the following lemma to give a precise return discrepancy (Luo et al., 2018; Yu et al., 2020).

Lemma 1 *(Luo et al. (2018), Lemma 4.3; Yu et al. (2020), Lemma 4.1; Shen et al. (2020), Lemma E.1) Let two MDPs share the same reward function $r(s, a)$, but with different dynamics $T(\cdot|s, a)$ and $\hat{T}(\cdot|s, a)$ respectively. Define $V_T^\pi(s) := \mathbb{E}_{\pi,T}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s\right]$ as the expected discounted return under π starting from state s, and $G_{\hat{T}}^\pi(s, a) := \mathbb{E}_{s'\sim\hat{T}}\left[V_T^\pi(s') \mid s, a\right] - \mathbb{E}_{s'\sim T}\left[V_T^\pi(s') \mid s, a\right]$. For any policy π, we have that*

$$\eta_{\hat{T}}(\pi) - \eta_T(\pi) = \kappa \mathbb{E}_{(s,a)\sim\rho_{\hat{T}}^\pi}[G_{\hat{T}}^\pi(s, a)], \tag{6}$$

where $\kappa = \gamma(1 - \gamma)^{-1}$.

For the sake of completeness, we provide a proof of Lemma 1, which is almost the same as in Yu et al. (2020). The only difference is that our occupancy measure is normalized.
Proof Let W_j be the expected return when executing π on $\hat{T}$ for the first j steps, then switching to T for the remainder. That is,

$$W_j = \mathbb{E}_{a_t\sim\pi,\ t<j: s_{t+1}\sim\hat{T},\ t\geq j: s_{t+1}\sim T}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 \sim \nu_0\right].$$

Note that $W_0 = \eta_T(\pi)$ and $W_\infty = \eta_{\hat{T}}(\pi)$, so

$$\eta_{\hat{T}}(\pi) - \eta_T(\pi) = \sum_{j=0}^{\infty}(W_{j+1} - W_j).$$

And we have

$$W_j = R_j + \mathbb{E}_{(s_j,a_j)\sim\pi,\hat{T}}\Big[\mathbb{E}_{s_{j+1}\sim T}\left[\gamma^{j+1} V_T^\pi(s_{j+1}) \mid s_j, a_j\right]\Big],$$

$$W_{j+1} = R_j + \mathbb{E}_{(s_j,a_j)\sim\pi,\hat{T}}\Big[\mathbb{E}_{s_{j+1}\sim\hat{T}}\left[\gamma^{j+1} V_T^\pi(s_{j+1}) \mid s_j, a_j\right]\Big],$$

where R_j is the expected return of the first j time steps, which are taken with respect to $\hat{T}$. Then

$$\begin{aligned} W_{j+1} - W_j &= \gamma^{j+1}\mathbb{E}_{(s_j,a_j)\sim\pi,\hat{T}}\left[\mathbb{E}_{s_{j+1}\sim\hat{T}}\left[V_T^\pi(s_{j+1}) \mid s_j, a_j\right] - \mathbb{E}_{s_{j+1}\sim T}\left[V_T^\pi(s_{j+1})\right] \mid s_j, a_j\right] \\ &= \gamma^{j+1}\mathbb{E}_{(s_j,a_j)\sim\pi,\hat{T}}\left[G_{\hat{T}}^\pi(s_j,a_j)\right]. \end{aligned}$$

Thus

$$\begin{aligned} \eta_{\hat{T}}(\pi) - \eta_T(\pi) &= \sum_{j=0}^{\infty}(W_{j+1} - W_j) \\ &= \sum_{j=0}^{\infty}\gamma^{j+1}\mathbb{E}_{(s_j,a_j)\sim\pi,\hat{T}}\left[G_{\hat{T}}^\pi(s_j,a_j)\right] \\ &= \frac{\gamma}{1-\gamma}\mathbb{E}_{(s,a)\sim\rho_{\hat{T}}^\pi}\left[G_{\hat{T}}^\pi(s,a)\right], \end{aligned}$$

as desired. ■

Lemma 1 states that the discrepancy between the return in the model and in the real environment can be rewritten as the discrepancy between the expected value of the next states output by the model and the real dynamic. Intuitively, it means that the expected return discrepancy will be small when the output distributions of the model and real dynamic are close.

Based on this theoretical result, Yu et al. (2020) built an upper bound as $\eta_{\hat{T}}(\pi) - \eta_T(\pi) \leq 2\gamma(1-\gamma)^{-2}r_{\max}\cdot\mathbb{E}_{(s,a)\sim\rho_{\hat{T}}^\pi}d_{\mathrm{TV}}(T(\cdot|s,a),\hat{T}(\cdot|s,a))$. However, it is impractical to design an algorithm to optimize this upper bound, since given the (s,a) sampled from $\rho_{\hat{T}}^\pi$, we could not directly obtain the real s' from $T(\cdot|s,a)$ in practice. Luo et al. (2018) used the opposite direction of the return discrepancy to obtain $\eta_T(\pi) - \eta_{\hat{T}}(\pi) \leq \kappa L\mathbb{E}_{(s,a)\sim\rho_T^\pi}[\|T(s,a) - \hat{T}(s,a)\|]$ assuming the real dynamics and the dynamics model are deterministic and the value function $V_{\hat{T}}^\pi$ is L-Lipschitz. However, to optimize this bound, it requires to collect samples from ρ_T^π for every iterate π. Their solution is to constrain the policy to be in the neighborhood of a reference policy π_{ref} while using the data collected by π_{ref}. Instead, in what follows we show that $\mathbb{E}_{\rho_{\hat{T}}^\pi}[G_{\hat{T}}^\pi(s,a)]$ can be further decomposed to construct an upper bound for the expected return discrepancy, which inspires our feature-based model adaptation method without introducing any further constraints or assumptions as in the previous works.

Theorem 2 *Let $\mathcal{F}_1$ be a collection of functions from $\mathcal{S}\times\mathcal{A}$ to $\mathbb{R}$. With Lemma 1, under the assumption that $G_{\hat{T}}^\pi(s,a)\in\mathcal{F}_1$, the expected return discrepancy admits the following bound:*

$$\eta_{\hat{T}}(\pi) - \eta_T(\pi) \leq \gamma(1-\gamma)^{-1}d_{\mathcal{F}_1}(\rho_T^{\pi_D}, \rho_{\hat{T}}^\pi) + \gamma(1-\gamma)^{-2}r_{\max}\cdot\mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}\sqrt{2D_{\mathrm{KL}}(T(\cdot|s,a),\hat{T}(\cdot|s,a))}. \quad (7)$$

Proof Let $\mathcal{F}_2$ be a collection of functions from $\mathcal{S}$ to $\mathbb{R}$ that $\mathcal{F}_2 := \left\{f : \|f\|_\infty \leq \frac{r_{\max}}{1-\gamma}\right\}$, and since $r_{\max} := \sup_{s,a}|r(s,a)|$, we have $V_T^\pi(s')\in\mathcal{F}_2$. Using Lemma 1, we can rewrite the expected return discrepancy as

$$\begin{aligned}\eta_{\hat{T}}(\pi) - \eta_T(\pi) =&\kappa \mathbb{E}_{(s,a)\sim\rho_{\hat{T}}^{\pi}}[G_{\hat{T}}^{\pi}(s,a)] - \kappa \mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}[G_{\hat{T}}^{\pi}(s,a)] + \kappa \mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}[G_{\hat{T}}^{\pi}(s,a)]\\ \le& \kappa \sup_{f\in\mathcal{F}_1}\Big(\mathbb{E}_{(s,a)\sim\rho_{\hat{T}}^{\pi}}[f(s,a)] - \mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}[f(s,a)]\Big)\\ &+ \kappa \mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}\Big[\sup_{g\in\mathcal{F}_2}\mathbb{E}_{s'\sim\hat{T}}\left[g(s')\mid s,a\right] - \mathbb{E}_{s'\sim T}\left[g(s')\mid s,a\right]\Big]\\ =&\kappa d_{\mathcal{F}_1}(\rho_T^{\pi_D},\rho_{\hat{T}}^{\pi}) + \kappa \mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}[d_{\mathcal{F}_2}(\hat{T}(\cdot|s,a),T(\cdot|s,a))],\end{aligned} \tag{8}$$

and since $\mathcal{F}_2 := \left\{f : \|f\|_\infty \le \frac{r_{\max}}{1-\gamma}\right\}$, the second term reduces to the total variation distance:

$$\begin{aligned}\eta_{\hat{T}}(\pi) - \eta_T(\pi) \le& \kappa d_{\mathcal{F}_1}(\rho_T^{\pi_D},\rho_{\hat{T}}^{\pi}) + 2\kappa(1-\gamma)^{-1}r_{\max}\cdot \mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}} d_{\mathrm{TV}}(T(\cdot|s,a),\hat{T}(\cdot|s,a))\\ \le& \kappa d_{\mathcal{F}_1}(\rho_T^{\pi_D},\rho_{\hat{T}}^{\pi}) + \kappa(1-\gamma)^{-1}r_{\max}\cdot \mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}\sqrt{2D_{\mathrm{KL}}(T(\cdot|s,a),\hat{T}(\cdot|s,a))},\end{aligned} \tag{9}$$

where the last inequality holds due to Pinsker's inequality. At last, replacing κ with $\gamma(1-\gamma)^{-1}$ completes the proof. ■

Remark Theorem 2 gives an upper bound on the discrepancy between the return in the model and the real environment. There are two terms in this bound: the last term corresponds to the model estimation error on real data, since the Kullback–Leibler divergence measures the average quality of current model estimation. The first term is the integral probability metric between the (s,a) distributions $\rho_T^{\pi_D}$ and $\rho_{\hat{T}}^{\pi}$, which exactly corresponds to the distribution shift problem between model learning and model usage.

To maximize $\eta_T(\pi)$, we would like to minimize the upper bound and maximize $\eta_{\hat{T}}(\pi)$ jointly over the policy and the dynamics model. In practice, we usually omit model optimization in the term $\eta_{\hat{T}}(\pi)$ for simplicity, following the practice in previous work (Luo et al., 2018). Then optimizing $\eta_{\hat{T}}(\pi)$ over the policy and optimizing the last term of upper bound over the model together becomes the standard principle of Dyna-style MBRL approaches, which may suffer catastrophic performance deterioration if the first term $d_{\mathcal{F}_1}(\rho_T^{\pi_D},\rho_{\hat{T}}^{\pi})$ is too large. Therefore, the key is to minimize the first term, *i.e.*, the occupancy measure divergence, which is intuitively reasonable since the dynamics model will predict simulated (s,a) samples close to its training data with high accuracy. To optimize this term over the policy, we can use imitation learning methods on the dynamics model, such as GAIL (Ho and Ermon, 2016), where the real samples are viewed as the expert demonstrations. However, optimizing this term over the policy is unnecessary, which can further reduce the efficiency of the whole training process. For example, one may optimize the policy excessively over this term instead of the $\eta_{\hat{T}}(\pi)$ term. So in this paper, we mainly focus on how to optimize this occupancy measure divergence term over the model.

3.2 Practical FAMPO Algorithm

To optimize the occupancy measure divergence term over the model, we first tackle it explicitly on feature level from the perspective of unsupervised domain adaptation (Ben-David

Algorithm 1 FAMPO

```
1: Initialize policy $\pi_\phi$, dynamics model $\hat{T}_\theta$, environment buffer $\mathcal{D}_{\text{env}}$, model buffer $\mathcal{D}_{\text{model}}$
2: repeat
3:     Perform $G_1$ gradient steps to train the model $\hat{T}_\theta$ with samples from $\mathcal{D}_{\text{env}}$
4:     for $F$ model rollouts do
5:         Sample a state $s$ uniformly from $\mathcal{D}_{\text{env}}$
6:         Use the policy $\pi_\phi$ to perform a $k$-step rollout on the model $\hat{T}_\theta$ starting from $s$; add
           to $\mathcal{D}_{\text{model}}$
7:     end for
8:     Perform $G_2$ gradient steps to train the feature extractor with samples $(s, a)$ from both
       $\mathcal{D}_{\text{env}}$ and $\mathcal{D}_{\text{model}}$ by the model adaptation loss $\mathcal{L}_{\text{WD}}$
9:     for $E$ timesteps in the real environment do
10:        Use the policy $\pi_\phi$ to take an action in the real environment; add the sample$(s, a, s', r)$
           to $\mathcal{D}_{\text{env}}$
11:        Use the model $\hat{T}_\theta$ to estimate the uncertainty for each sample in $\mathcal{D}_{\text{model}}$
12:        Perform $G_3$ gradient steps to train the policy $\pi_\phi$ with real data uniformly sampled
           from $\mathcal{D}_{\text{env}}$ and simulated data sampled from $\mathcal{D}_{\text{model}}$ according to the uncertainty
13:    end for
14: until certain number of real samples have been collected
```

et al., 2010; Zhao et al., 2019), which aims at generalizing a learner on unlabeled data with labeled data from a different distribution. One promising solution to this problem is to find invariant feature representations by incorporating an additional objective of feature distribution alignment (Ben-David et al., 2007; Ganin et al., 2016). This motivates a model adaptation procedure to encourage the dynamics model to learn the features that are invariant to the real state-action data and the simulated one.

Simultaneously, model adaptation can also be incorporated into existing Dyna-style MBRL methods, including those by reducing the distribution shift problem. Thereafter, in this paper, we adopt MBPO (Janner et al., 2019) as our baseline backbone framework due to its remarkable success in practice. We dub the integrated framework FAMPO and detail the step-by-step algorithm in Algorithm 1, where we defer the discussion on the weighted sampling strategy (line 11 to 12) to Section 5.

3.2.1 Incorporating Unsupervised Model Adaptation

For brevity, we will only introduce how to train individual dynamics model here, which can be easily extended to other models in the ensemble. Since our models are implemented by a neural network, we can define the shallow layers as the feature extractor f_g with corresponding parameters θ_g and the remaining layers as the decoder f_d with parameters θ_d. Thus, we have $\hat{T} = f_d \circ f_g$ and $\theta = \{\theta_g, \theta_d\}$. We integrate a model adaptation loss over the output of feature extractor, which encourages such a conceptual division as the feature encoder and the decoder. The main idea of model adaptation is to adjust the feature extractor f_g in order to align the two feature distributions of real samples and simulated ones as input, so that the induced feature distributions from real and simulated samples can be close in the feature space.

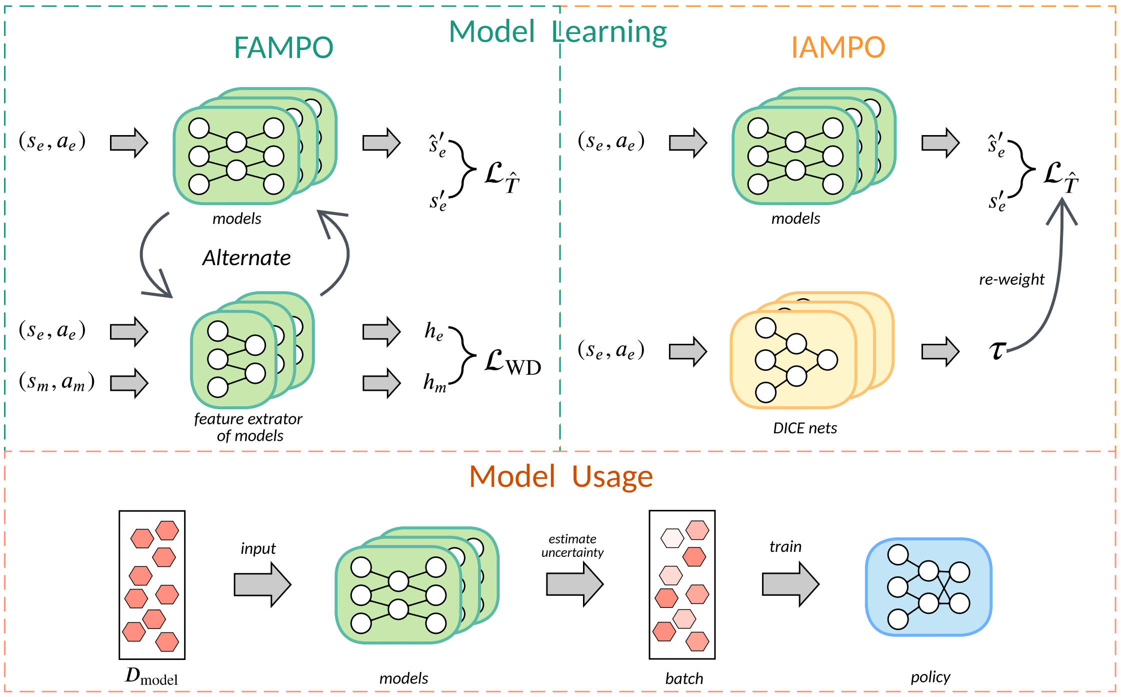


Figure 1: Illustration of FAMPO and IAMPO algorithms. For model learning, FAMPO alternates between minimizing the standard MLE loss and aligning the latent feature distributions of models, while IAMPO trains the DICE nets to estimate the occupancy measure density ratio τ, which is then used to re-weight the real samples for model training. For model usage, when forming a batch to train the policy, the simulated data is selected with different probabilities based on the estimated uncertainty, which is represented by the shade of color.

To incorporate unsupervised model adaptation into MBPO, we adopt alternative optimization between model training and model adaptation as illustrated in Figure 1. At every iteration (line 3 to 8 in Algorithm 1), when the dynamics model is trained, we use it to generate simulated rollouts which will then be used for model adaptation and policy optimization. As for the detailed adaptation strategy, instead of directly sharing the parameter weights of the feature extractor between real data and simulated data (Ganin et al., 2016), we choose to adopt the asymmetric feature mapping strategy (Tzeng et al., 2017), which has been shown to outperform the weight-sharing variant in domain adaptation due to more flexible feature mappings. To be specific, the asymmetric feature mapping strategy unties the shared weights between two domains and learns individual feature extractors for real data and simulated data respectively. Thus in FAMPO, after the model adaptation at one iteration is finished, we will use the weight parameters for simulated data to initialize the model training for the next iteration. Through such an alternative optimization between model training and model adaptation, the feature representations learned by the feature extractor will be informative for the decoder to predict real samples, and more importantly it can generalize to the simulated samples.

3.2.2 Model Adaptation via Wasserstein-1 Distance

Specifically, given real samples (s_e, a_e) from the environment buffer $\mathcal{D}_{\text{env}}$ and the simulated samples (s_m, a_m) from the model buffer $\mathcal{D}_{\text{model}}$, the two separate feature extractors map them to feature representations $h_e = f_g^e(s_e, a_e)$ and $h_m = f_g^m(s_m, a_m)$. To achieve model adaptation, we minimize one kind of IPM between the two feature distributions $\mathbb{P}_{h_e}$ and $\mathbb{P}_{h_m}$ according to the lower bound in Theorem 2. In this paper, we choose the Wasserstein-1 distance as the divergence measure in model adaptation, which is validated to be effective in domain adaptation (Shen et al., 2018). In Appendix C, we also provide a variant that uses Maximum Mean Discrepancy (MMD).

Wasserstein-1 distance corresponds to an instance of IPM where the witness function satisfies the 1-Lipschitz constraint. To estimate the Wasserstein-1 distance, we use a critic network f_c with parameters ω as introduced in Wasserstein GAN (Arjovsky et al., 2017). The critic maps a feature representation to a real number, and then according to Eq. (2) the Wasserstein-1 distance can be estimated by maximizing the following objective function over the critic:

$$\mathcal{L}_{\text{WD}}(\theta_g^e, \theta_g^m, \omega) = \frac{1}{N_e}\sum_{i=1}^{N_e} f_c(h_e^i) - \frac{1}{N_m}\sum_{j=1}^{N_m} f_c(h_m^j). \tag{10}$$

In the meanwhile, the parameterized family of critic functions $\{f_c\}$ should satisfy the 1-Lipschitz constraint according to the IPM formulation of Wasserstein-1 distance. In order to properly enforce the 1-Lipschitz constraint, we choose the gradient penalty loss (Gulrajani et al., 2017) for the critic

$$\mathcal{L}_{gp}(\omega) = \mathbb{E}_{\mathbb{P}_{\hat{h}}}[(\|\nabla f_c(\hat{h})\|_2 - 1)^2], \tag{11}$$

where $\mathbb{P}_{\hat{h}}$ is obtained by uniformly sampling along straight lines between pairs of points sampled from $\mathbb{P}_{h_e}$ and $\mathbb{P}_{h_m}$ as suggested in Gulrajani et al. (2017).

After the critic is trained to approximate the Wasserstein-1 distance, we optimize the feature extractor to minimize the estimated Wasserstein-1 distance to learn features invariant to the real data and simulated data. To sum up, model adaptation though Wasserstein-1 distance can be achieved by solving the following minimax objective

$$\min_{\theta_g^e, \theta_g^m} \max_{\omega} \quad \mathcal{L}_{\text{WD}}(\theta_g^e, \theta_g^m, \omega) - \alpha \cdot \mathcal{L}_{gp}(\omega), \tag{12}$$

where θ_g^e and θ_g^m are the parameters of the two feature generators for real data and simulated data respectively, and α is the balancing coefficient. For model adaptation at each iteration, we alternate between training the critic to estimate the Wasserstein-1 distance and training the feature extractor of the dynamics model to learn transferable features.

4. Instance-based Adaptation-augmented MBPO

Apart from explicitly mitigating the distribution shift by learning invariant features as in Section 3, in this section, we additionally propose an instance-based model adaptation method to tackle this problem on data level, which is motivated by the following derivation.

4.1 Derivative of Return Discrepancy

Recall that we presented a precise return discrepancy in Lemma 1, and further decomposed it to derive Theorem 2. Instead, we can also directly optimize the dynamics model by taking the gradient of the return discrepancy. To take a further step, according to the definition of integral probability metric (IPM) (Müller, 1997), let $\mathcal{F}$ be a collection of functions from $\mathcal{S}$ to $\mathbb{R}$, under the assumption that $V_T^\pi \in \mathcal{F}$, we have

$$\begin{aligned} G_{\hat{T}}^\pi(s,a) &\le \sup_{f\in\mathcal{F}} \mathbb{E}_{s'\sim\hat{T}}\left[f(s') \mid s,a\right] - \mathbb{E}_{s'\sim T}\left[f(s') \mid s,a\right] \\ &=: d_{\mathcal{F}}(\hat{T}(\cdot|s,a), T(\cdot|s,a)), \end{aligned} \tag{13}$$

where $d_{\mathcal{F}}$ is the IPM defined by the function class $\mathcal{F}$. Denoting the dynamics model as $\hat{T}_\theta$ parameterized by θ, we have

$$\begin{aligned} &\nabla_\theta \mathbb{E}_{(s,a)\sim\rho_{\hat{T}_\theta}^\pi}[d_{\mathcal{F}}(T(\cdot|s,a),\hat{T}_\theta(\cdot|s,a))] = \nabla_\theta \int_{s,a} \rho_{\hat{T}_\theta}^\pi d_{\mathcal{F}}(T(\cdot|s,a),\hat{T}_\theta(\cdot|s,a))\,\mathrm{d}s\,\mathrm{d}a \\ &= \int_{s,a} (\rho_{\hat{T}_\theta}^\pi \nabla_\theta d_{\mathcal{F}}(T(\cdot|s,a),\hat{T}_\theta(\cdot|s,a)) + d_{\mathcal{F}}(T(\cdot|s,a),\hat{T}_\theta(\cdot|s,a))\nabla_\theta \rho_{\hat{T}_\theta}^\pi)\,\mathrm{d}s\,\mathrm{d}a \\ &= \mathbb{E}_{\rho_{\hat{T}_\theta}^\pi}[\nabla_\theta d_{\mathcal{F}}(T(\cdot|s,a),\hat{T}_\theta(\cdot|s,a))] + \mathbb{E}_{\rho_{\hat{T}_\theta}^\pi}[d_{\mathcal{F}}(T(\cdot|s,a),\hat{T}_\theta(\cdot|s,a))\nabla_\theta \log\rho_{\hat{T}_\theta}^\pi]. \end{aligned} \tag{14}$$

A justification of the interchange of the gradient and integration in Equation 14 is provided in Appendix B. There are two terms in this equation: one is the derivative of the IPM between the real transition T and the dynamics model $\hat{T}_\theta$, and the other is the derivative of the occupancy measure of policy π in the dynamics model $\hat{T}_\theta$.

The first term in Eq. 14 suggests us to minimize the IPM under the expectation over the occupancy measure $\rho_{\hat{T}_\theta}^\pi$. To derive an optimization objective according to this term, we need to tackle two questions. First, which specific IPM form should we use? Second, how to compute the IPM when we cannot access the ground-truth of $T(\cdot|s,a)$ for the data drawn from $\rho_{\hat{T}_\theta}^\pi$?

To answer the first question, as in Theorem 2, we have $\|V_T^\pi\|_\infty \le (1-\gamma)^{-1} r_{\max}$ and then the IPM reduces to the total variation distance $(1-\gamma)^{-1} r_{\max} d_{\mathrm{TV}}(T,\hat{T}_\theta)$. Then according to Pinsker's inequality, we have $d_{\mathrm{TV}}(T,\hat{T}_\theta) \le \sqrt{\frac{1}{2} D_{\mathrm{KL}}(T(\cdot|s,a)\|\hat{T}_\theta(\cdot|s,a))}$. By discarding the constants, the loss function can be written as

$$\mathcal{L}_{\hat{T}}(\theta) = \mathbb{E}_{(s,a)\sim\rho_{\hat{T}_\theta}^\pi}[D_{\mathrm{KL}}(T(\cdot|s,a)\|\hat{T}_\theta(\cdot|s,a))]. \tag{15}$$

Eq. 15 justifies the use of maximum likelihood estimation (MLE) since MLE is the minimizer of the empirical approximation of the Kullback–Leibler divergence. In practice, following MBPO (Janner et al., 2019), we use an ensemble of probabilistic networks to represent the model and train the model ensemble via maximum likelihood. However, since the first component in KL requires the output of ground-truth dynamics T for model generated data $\rho_{\hat{T}_\theta}^\pi$, this loss function cannot be directly optimized.

In an online MBRL method with iterative policy improvement, we denote the data-collecting policy as π_D, which represents a collection of policies in previous iterations. Hence, we will have the transitions (s, a, s') collected by the behavioral policy π_D in the real environment dynamics T, which are stored in the buffer $\mathcal{D}_{\text{env}}$. Therefore, we use importance sampling to deal with the second question:

$$\mathcal{L}_{\hat{T}}(\theta) = \mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}\Big[\perp\frac{\rho_{\hat{T}_\theta}^{\pi}}{\rho_T^{\pi_D}}\, D_{\text{KL}}(T(\cdot|s,a)\|\hat{T}_\theta(\cdot|s,a))\Big], \tag{16}$$

where the stop-gradient operator $\perp$ means the importance ratio $\tau^*(s,a) = \rho_{\hat{T}_\theta}^{\pi}(s,a)/\rho_T^{\pi_D}(s,a)$ is treated as a constant when taking derivatives of this objective. The proposed instance-based model learning technique uses the real data from $\mathcal{D}_{\text{env}}$ to train the model as previous works, but the loss function for each data point is re-weighted by the occupancy measure density ratio.

4.2 Importance Ratio Estimation

In order to apply the above importance-ratio scheme, we need to estimate the importance ratio $\tau^*(s,a)$ for each instance (s,a) in the real data buffer $\mathcal{D}_{\text{env}}$. In practice, the importance ratio can be estimated either by the classical binary classification (Zadrozny, 2004) or by the DICE (DIstribution Correction Estimation) framework. We utilize DICE to estimate the importance ratio by default, and also provide the comparison of these two ratio estimation techniques in Appendix C.

4.2.1 Binary Classification

One direct way to estimate this ratio is constructing a binary classification problem where the real data in $\mathcal{D}_{\text{env}}$ is labeled positive and the simulated data in $\mathcal{D}_{\text{model}}$ is labeled negative (Zadrozny, 2004). To be more specific, let $\zeta(s,a)$ be the output of the classifier on some state-action pair (s,a), then the importance ratio can be estimated by

$$\tau(s,a) = \psi(1-\zeta(s,a))/\zeta(s,a), \tag{17}$$

where ψ is the ratio of positive and negative samples. In practical implementation, the classifier can be modeled as feed-forward neural networks and trained to minimize the cross-entropy loss.

4.2.2 Importance Ratio Estimation via DICE

Although binary classification is simple to implement, in MBRL the number of real data and the simulated data is not balanced since we hope to simulate much more simulated data to help improve the policy when real data is limited. This imbalanced classification problem may cause the learning process biased (Krawczyk, 2016) and lead to poor estimation. To sidestep the above obstacle, we can turn to use the DICE framework to estimate the ratio by exploiting the Markov property.

In off-policy evaluation where we want to estimate the performance of a target policy from data generated by a behavioral policy, one promising solution is to re-weight the reward by the occupancy measure density ratio. Recently several works have proposed to estimate this ratio such as DualDICE (Nachum et al., 2019), GenDICE (Zhang et al., 2020a) and GradientDICE (Zhang et al., 2020b). In this paper, we adapt the GradientDICE architecture to estimate the importance ratio in Eq. 16. We will briefly introduce the whole framework. For more detailed introduction of this framework, we refer the readers to Zhang et al. (2020b). To begin with, the occupancy measure satisfies the following equation:

$$\rho^{\pi}_{\hat{T}}(s',a') = (1-\gamma)\nu_0(s')\pi(a'|s') + \gamma \int \rho^{\pi}_{\hat{T}}(s,a)\hat{T}(s'|s,a)\pi(a'|s')\, \mathrm{d}s\, \mathrm{d}a, \tag{18}$$

where ν_0 is the initial state distribution. Then using $\rho^{\pi_D}_{T}(s,a)\tau(s,a)$ to replace $\rho^{\pi}_{\hat{T}}(s,a)$ and minimizing some divergence between the LHS and RHS of Eq. 18 over the function τ with an additional constraint can finally estimate the importance ratio $\tau^*(s,a)$. Denoting $\delta(s',a') = \gamma \int \rho^{\pi_D}_{T}(s,a)\tau(s,a)\hat{T}(s'|s,a)\pi(a'|s')\, \mathrm{d}s\, \mathrm{d}a + (1-\gamma)\nu_0(s')\pi(a'|s') - \rho^{\pi_D}_{T}(s',a')\tau(s',a')$, we have the following objective function

$$\mathcal{L}(\tau) = \frac{1}{2}\mathbb{E}_{\rho^{\pi_D}_{T}}\Big[\Big(\frac{\delta(s,a)}{\rho^{\pi_D}_{T}(s,a)}\Big)^2\Big] + \frac{\lambda}{2}(\mathbb{E}_{\rho^{\pi_D}_{T}}[\tau(s,a)] - 1)^2 \ , \tag{19}$$

where $\lambda > 0$ is a constant coefficient. By further applying Fenchel conjugate (Rockafellar, 1970) and the interchangeability principle as in Zhang et al. (2020a), optimizing the final objective of GradientDICE is a minimax problem as

$$\begin{aligned}\min_{\tau}\max_{f,\beta} \mathcal{L}(\tau,\beta,f) =& (1-\gamma)\mathbb{E}_{s\sim\nu_0, a\sim\pi}[f(s,a)] \\ &+ \gamma\mathbb{E}_{(s,a)\sim\rho^{\pi_D}_{T}, s'\sim\hat{T}, a'\sim\pi}[\tau(s,a)f(s',a')] \\ &- \mathbb{E}_{(s,a)\sim\rho^{\pi_D}_{T}}[\tau(s,a)f(s,a) + \frac{1}{2}f(s,a)^2] \\ &+ \lambda(\mathbb{E}_{(s,a)\sim\rho^{\pi_D}_{T}}[\beta\tau(s,a) - \beta] - \frac{1}{2}\beta^2) \ , \end{aligned} \tag{20}$$

where we have $\tau : \mathcal{S}\times\mathcal{A} \to \mathbb{R}$, $f : \mathcal{S}\times\mathcal{A} \to \mathbb{R}$ and $\beta \in \mathbb{R}$. In practical implementation, we use feed-forward neural networks to model the function f and τ. Since we use a model ensemble and the importance ratio is model-specific, we also construct separate DICE network for each dynamics model. Furthermore, because the simulated data is generated from some states in $\mathcal{D}_{\text{env}}$, we use the collected real states to approximate the initial state distribution $\nu_0(s)$. After the DICE network estimates the ratio for each real sample, we clip it to be in the interval $[\alpha_0, \alpha_1]$ with the intention of improving the stability.

We demonstrate the detailed process of IAMPO upon the MBPO (Janner et al., 2019) backbone in Algorithm 2. Specifically, in each iteration, IAMPO first trains the DICE network (line 3), which is then used to estimate the importance ratio (line 4). Like in Algorithm 1, the weighted sampling strategy (line 12 to 13) will be introduced in Section 5.

5. Weighted Sampling Strategy

Although the model adaptation methods proposed in previous sections help to alleviate the distribution shift problem in model learning, some inaccurate predictions can still lead to catastrophic performance degradation due to the imperfect approximation. A straightforward solution is to discard the data with large model error when use them for policy optimization. In practice, it is non-trivial to estimate the model error and instead we can use model uncertainty that is considered to be positively correlated to the model error to replace it. For

Algorithm 2 IAMPO

```
1: Initialize policy $\pi_\phi$, dynamics model $\hat{T}_\theta$, DICE network $\{\tau, f, \beta\}$, environment buffer
   $\mathcal{D}_{\text{env}}$, model buffer $\mathcal{D}_{\text{model}}$
2: repeat
3:   Perform $G_2$ gradient steps to train the DICE network $\{\tau, f, \beta\}$ with data from $\mathcal{D}_{\text{env}}$
     according to Eq. 20
4:   Use $\tau$ to estimate the ratio for each sample in $\mathcal{D}_{\text{env}}$
5:   Perform $G_1$ gradient steps to train the model $\hat{T}_\theta$ with the loss re-weighted by the ratio
     using data in $\mathcal{D}_{\text{env}}$
6:   for $F$ model rollouts do
7:     Sample a state $s$ uniformly from $\mathcal{D}_{\text{env}}$
8:     Use the policy $\pi_\phi$ to perform a $k$-step rollout on the model $\hat{T}_\theta$ starting from $s$; add
       to $\mathcal{D}_{\text{model}}$
9:   end for
10:  for $E$ timesteps in the real environment do
11:    Use the policy $\pi_\phi$ to take an action in the real environment; add the sample$(s, a, s', r)$
       to $\mathcal{D}_{\text{env}}$
12:    Use the model $\hat{T}_\theta$ to estimate the uncertainty for each sample in $\mathcal{D}_{\text{model}}$
13:    Perform $G_3$ gradient steps to train the policy $\pi_\phi$ with real data uniformly sampled
       from $\mathcal{D}_{\text{env}}$ and simulated data sampled from $\mathcal{D}_{\text{model}}$ according to the uncertainty
14:  end for
15: until certain number of real samples have been collected
```

example, Janner et al. (2019) used the model to generate short rollouts so that the longer rollouts with higher uncertainty are cut off. Pan et al. (2020) explicitly estimated the model uncertainty and masked the simulated data with high uncertainty. These methods directly enforce the occupancy measure of high-uncertainty simulated data to be zero. However, it may be too restrictive since there is no guarantee that high uncertainty means high model error, and the masked simulated data may be still useful if its ground-truth model error is small.

To tackle the above challenges, we propose to utilize a weighted sampling strategy during model usage phase for better generalization. Formally, as illustrated in Figure 1, when forming a mini-batch for policy training, the weighted sampling strategy chooses simulated data in the model buffer $\mathcal{D}_{\text{model}}$ according to a Boltzmann distribution based on the estimated uncertainty $d(s, a)$. For a simulated data $x_i = (s_i, a_i, s'_i, r_i)$, the probability $p(x_i)$ of being sampled into the mini-batch is

$$p(x_i) = \frac{\exp(-\sigma \cdot d(s_i, a_i))}{\sum_{x_j} \exp(-\sigma \cdot d(s_j, a_j))} \tag{21}$$

where σ is a temperature parameter. There are multiple ways to estimate the uncertainty, such as the maximum standard deviation of the learned models in the ensemble (Yu et al., 2020) or the prediction disagreement between one model versus the rest of the models (Pan et al., 2020). In this paper, we follow the uncertainty estimation method in Kidambi et al. (2020) and use the maximum prediction discrepancy $d(s, a) = \max_{i,j} \|\hat{T}_{\theta_i}(s, a) - \hat{T}_{\theta_j}(s, a)\|_2$

where $\hat{T}_{\theta_i}$ and $\hat{T}_{\theta_j}$ are members in the model ensemble $\{\hat{T}_{\theta_1}, \hat{T}_{\theta_2}, \dots\}$. Besides, the sampling temperature σ is set as $\min\{\sigma_0, \frac{\sigma_1}{d_{\max}-d_{\min}}\}$ in practice, where σ_0 and σ_1 are hyperparameters, and $d_{\max}$ and $d_{\min}$ are the maximum and minimum uncertainty estimated in the model buffer respectively.

Alternatively, one can also view the weighted sampling strategy as implicitly optimizing the second term of Eq. 14, since the second term, which is in the derivative form, suggests to minimize $\log \rho^{\pi}_{\hat{T}_\theta}$ weighted by $d_{\mathcal{F}}(T, \hat{T})$. And the weighted sampling strategy reduces the occupancy measure $\rho^{\pi}_{\hat{T}_\theta}$ of simulated data if its uncertainty is high, which is positively correlated to the model error $d_{\mathcal{F}}(T, \hat{T})$.

6. Experiments

The experiments aim to answer the following three questions: 1) How do FAMPO/IAMPO perform compared to prior model-free and model-based methods? 2) Do the feature-based and instance-based adaptation methods address the distribution shift problem in MBRL as we expected? 3) What are key ingredients in our algorithms?

6.1 Comparative Evaluation

Compared Methods. We compare the proposed FAMPO and IAMPO to other model-free and model-based algorithms: Soft Actor-Critic (SAC) (Haarnoja et al., 2018), the state-of-the-art model-free off-policy algorithm in terms of sample efficiency and asymptotic performance. For model-based methods, we compare to MBPO (Janner et al., 2019), PETS (Chua et al., 2018) and SLBO (Luo et al., 2018), where PETS directly uses the model for planning without explicit policy learning and SLBO trains the model with a multi-step L2-norm loss and updates the policy using TRPO (Schulman et al., 2015).

Environments. We evaluate our methods and other baselines on six MuJoCo continuous control tasks from OpenAI Gym (Brockman et al., 2016) with a maximum horizon of 1000, including InvertedPendulum, Swimmer, Hopper, Walker2d, Ant and HalfCheetah. For the Swimmer environment, we use the modified version introduced by Langlois et al. (2019) since the original version is quite difficult to solve. For the other five environments, we adopt the same settings as in Janner et al. (2019).

Implementation Details. We implement all our experiments using TensorFlow. For MBPO, FAMPO and IAMPO, we first apply a random policy to sample a certain number of real data to pre-train the dynamics model. Every time we train the dynamics model, we randomly sample several real data as a validation set and stop the model training if the model loss does not decrease for five gradient steps, which means we do not choose a specific value for the hyperparameter G_1. Other important hyperparameters used in our methods are chosen by grid search and detailed hyperparameter settings can be found in Appendix E.

Results. The learning curves of all compared methods are presented in Figure 2. From the comparison, we observe that FAMPO and IAMPO are more sample efficient as they learn faster than all other baselines in all six environments. Furthermore, our methods are capable of reaching comparable asymptotic performance of the state-of-the-art model-free baseline SAC. Compared with MBPO, our approaches achieve better performance in all the environments, which verifies the value of model adaptation. This also indicates that even

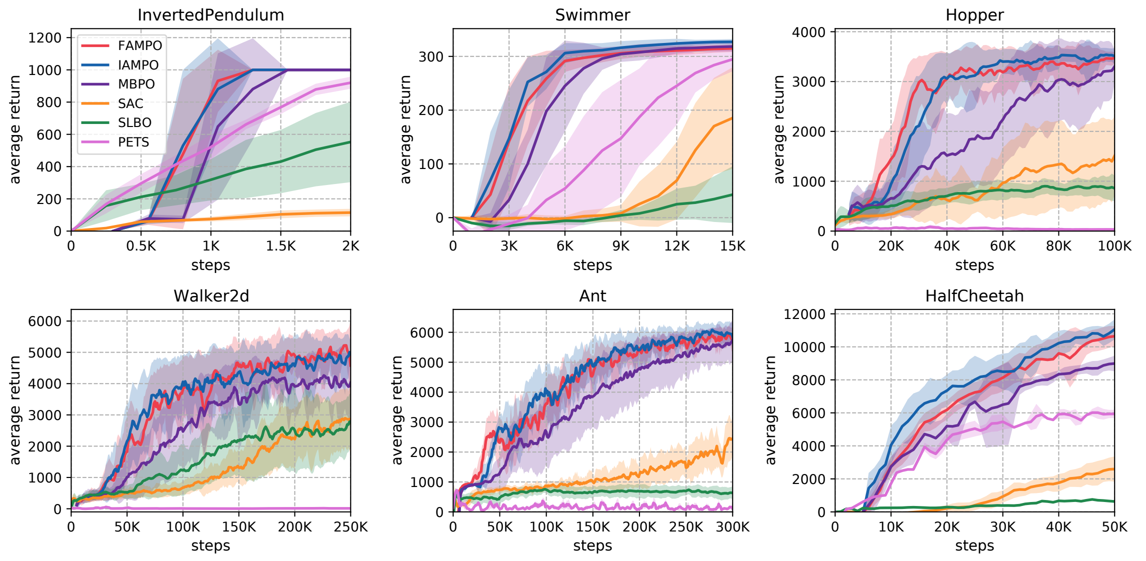
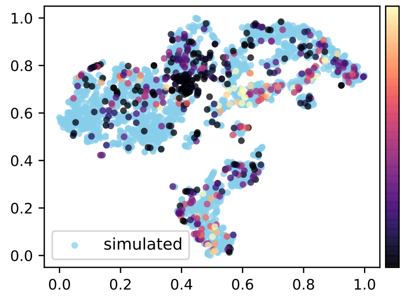


Figure 2: Performance curves of our methods and other baselines on six MuJoCo tasks. The results are averaged over eight random seeds, where solid curves depict the mean of eight trials and shaded areas indicate the standard deviation. For each trail, the average return over ten episodes in the real environment is evaluated every 1000 environment timesteps. For MBPO, FAMPO and IAMPO, in the dynamics model pre-training stage, the policy is not updated and thus the performance is not plotted at the beginning.

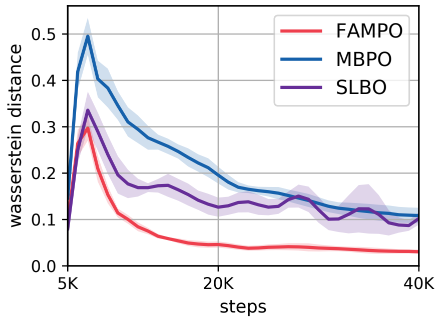
in the situation with reduced distribution shift by using short rollouts, model adaptation still helps. By comparing FAMPO and IAMPO, we can see that these two variants are comparably effective across different environments. Further analysis of the feature-based and instance-based adaptation is provided in Section 6.2.

6.2 Empirical Analysis

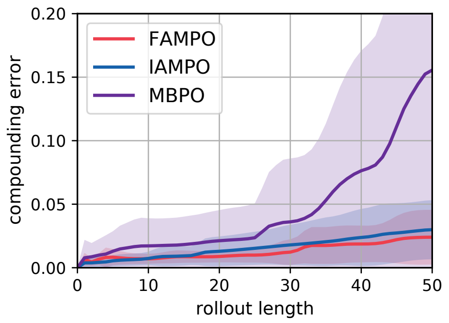
Distribution shift. To investigate how our proposed methods help mitigate the distribution shift problem in MBRL, we first visualize the real data and the simulated data in Figure 3(a). In particular, after training IAMPO for 50 epochs on Hopper, we randomly sample 500 real (s, a) data from $\mathcal{D}_{\text{env}}$ and 2000 simulated (s, a) data from $\mathcal{D}_{\text{model}}$. Then, we plot the normalized t-SNE visualization of the state-action pairs with the simulated data colored in blue and the real data colored differently according to the importance ratio estimated by IAMPO. It can be easily concluded that the distribution of simulated data deviates from that of real data, which may lead to poor model prediction in these areas. Moreover, if the real data is densely distributed and the simulated data is sparse, *i.e.*, the ground-truth density ratio $\rho^{\pi}_{\hat{T}_\theta}(s, a)/\rho^{\pi_D}_{T}(s, a)$ is small, the estimated ratio is also small (black points drawn in the figure), and vice versa. This visualization implies that the importance ratio of IAMPO is estimated as we expect.



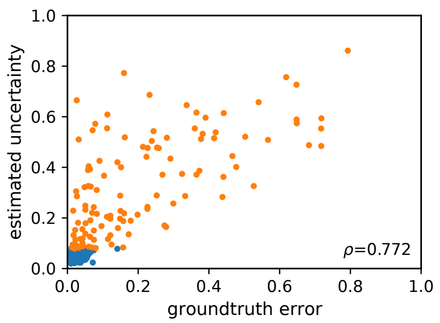
(a) distribution shift.



(b) Wasserstein-1 distance.



(c) Compounding errors.



(d) Uncertainty.

Figure 3: Empirical result on Hopper environment. (a) T-SNE visualization of state-action pairs randomly sampled from $\mathcal{D}_{\text{env}}$ and $\mathcal{D}_{\text{model}}$. Simulated data is drawn in blue while real data is drawn in different colors, and the brighter points represent the real samples with larger importance ratio. (b) The Wasserstein-1 distance between the feature distributions. (c) Average compounding errors with different rollout lengths. (d) Relationship between the estimated uncertainty and the ground-truth error of the real samples. The orange points are the half of the data with high uncertainty, and the blue ones are the half with low uncertainty.

Wasserstein-1 Distance. Instead of re-weighting the real samples as shown in Figure 3(a), FAMPO explicitly align the feature distribution of real samples and simulated ones. To further verify the effect of feature-based model adaptation, we visualize the estimated Wasserstein-1 distance between the real features and simulated ones. Besides MBPO and FAMPO, we additionally analyze the multi-step training loss of SLBO since it also utilizes the model output as the input of model training, which may help learn invariant features. According to the results shown in Figure 3(b), we find that: i) the vanilla model training in MBPO itself can slowly minimize the Wasserstein-1 distance between feature distributions; ii) the multi-step training loss in SLBO does help learn invariant features but the improvement is limited; iii) the model adaptation loss in FAMPO is effective in promoting feature distribution alignment, which is consistent with our initial motivation.

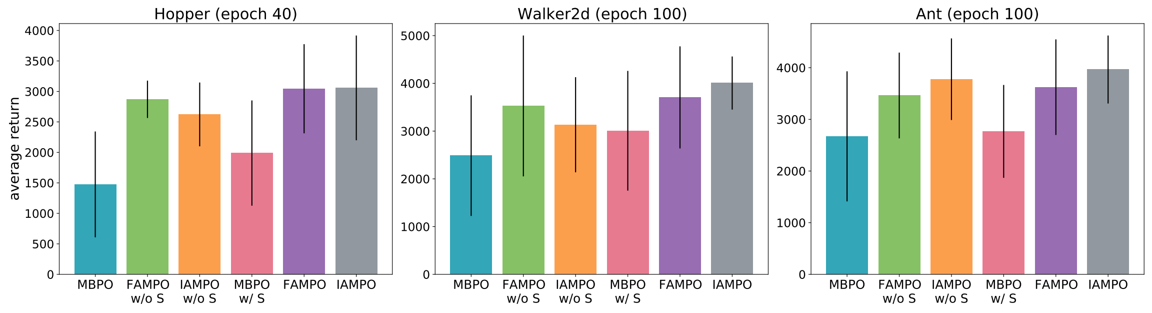


Figure 4: The results of the ablation studies conducted on three environments with few interactions (40K steps for Hopper, 100K steps for Walker2d and ant). The bars are average returns over five trials and black error lines indicate the standard deviation.

Compounding Errors. We also investigate if the model adaptation help alleviate the compounding model errors (Nagabandi et al., 2018) of multi-step forward predictions, which is largely caused by the distribution shift problem. Concretely, we use the current policy to sample a trajectory $(s_0, a_0, s_1, ..., a_{h-1}, s_h)$ of length h on real environment, and use the learned dynamics model to generate the corresponding simulated rollouts $(\hat{s}_0, a_0, \hat{s_1}, ..., a_{h-1}, \hat{s}_h)$ where $\hat{s}_0 = s_0$ and $\hat{s}_{i+1} = \hat{T}_\theta(\hat{s}_i, a_i)$. Then the empirical compounding error is calculated as $\epsilon_h = \frac{1}{h}\sum_{i=1}^{h} \|\hat{s}_i - s_i\|^2$. We conduct experiments with different trajectory length h and plot the results in Figure 3(c). We find that both FAMPO and IAMPO achieves smaller compounding errors than MBPO, which meets our motivation that model adaptation can successfully mitigate the distribution shift.

Uncertainty. For our weighted sampling strategy, one critical question is whether the estimated uncertainty matches the real state prediction error (*e.g.*the IPM in the second term of Eq. 14). However, since it is hard to obtain the ground-truth label of the simulated data due to the complexity of MuJoCo simulator, we instead evaluate the uncertainty quantification using sampled real data $(s, a, s') \in \mathcal{D}_{\text{env}}$. To be specific, we use the dynamics model to predict the next state $\hat{s}'$ of newly collected real samples, on which the model hasn't been trained. Then we calculate the ground-truth error $\|\hat{s}' - s'\|^2$ and estimate the uncertainty $d(s, a)$. Figure 3(d) shows the relationship between the estimated uncertainty and the ground-truth error, from which we get to know that in general there is a positive correlation between the estimated uncertainty and the ground-truth error justifying the incorporation of the uncertainty in algorithm design. On the other hand, we further find directly masking half of the data with highest uncertainty will also mask data with low ground-truth error, revealing the disadvantage of hard-masking mechanism.

6.3 Ablation Studies

In this section, we aim to investigate the key ingredients of our algorithms through ablation studies. Particularly, we compare the corresponding variants : (i) FAMPO w/o S, which only adopts feature-based model adaptation without incorporating weighted sampling strategy; (ii) IAMPO w/o S, which only adopts instance-based model adaptation without incorporating

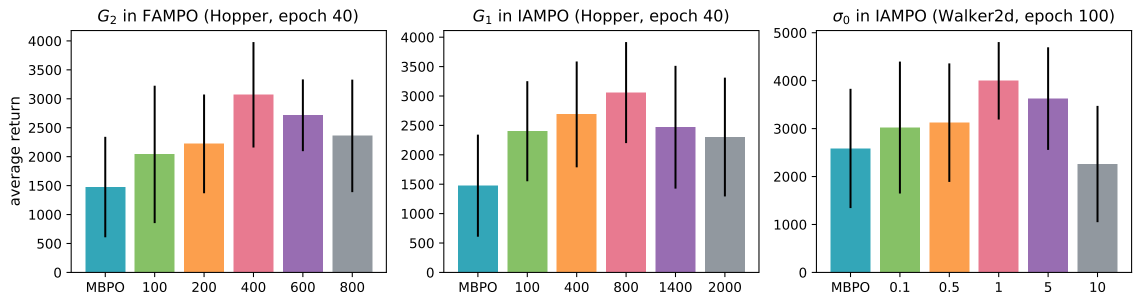


Figure 5: The results of the hyperparameter studies with respect to the adaptation iteration G_2 in FAMPO, the DICE training iteration G_1 and the temperature of the probabilistic sampling σ_0 in IAMPO. The experiments are conducted on Hopper with 40k steps and Walker2d with 100 steps.

weighted sampling strategy; (iii) MBPO w/ S, which only adopts weighted sampling strategy without incorporating model adaptation. We conduct the experiments of ablation studies on three environments and the results are shown in Figure 4.

From the comparison, we observe that: 1) Both the feature-based model adaptation and instance-based model adaptation are quite effective, since in all three environments FAMPO-S and IAMPO-S improve the performance in a considerable margin compared to MBPO. 2) The effectiveness of weighted sampling strategy varies across the three environments. To be more specific, weighted sampling strategy shows its effectiveness on Hopper and Walker2d with different degrees of performance improvement, but it does not improve much on Ant. The reason may be that on complex environments it is difficult to estimate the uncertainty well to reach the positive correlation to the ground-truth model error.

6.4 Hyperparameter Studies

In this section, we study the sensitivity of our methods to important hyperparameters. Specifically, For model adaptation learning, we investigate the sensitivity of FAMPO to the adaptation iteration G_2 and IAMPO to the DICE training iteration G_1. While for weighted sampling strategy, we would like to assess the importance of the probabilistic sampling temperature σ. We conduct experiments with different value of these hyperparameters and plot the result in Figure 5. According to the results, We observe that increasing G_2 yields better performance up to a certain level while too large G_2 degrades the performance, which means that we need to control the trade-off between model training and model adaptation to ensure the feature representations to be invariant and also discriminative.

Similar trend can be summarized in the result of G_1 and σ, but in most cases the performance is still better than MBPO. To explain, too small G_1 can't train the DICE networks sufficiently but too large G_1 may cause overfitting. And with relatively small σ, the algorithm just adopts uniform sampling in model usage, while using too large σ greatly degrades the performance since only focusing low-uncertainty simulated data may reduce the data diversity to obtain a good policy.

7. Related work

Recently, model-based reinforcement learning have attracted increasing attention, mainly due to its high sample efficiency by learning a model of the environment dynamics and using the model for policy optimization. In the literature, for MBRL, there are two stages, *i.e.*, model learning and model usage. Among them, model learning mainly involves two aspects: (1) function approximator choice like Gaussian process (Deisenroth and Rasmussen, 2011), time-varying linear models (Sutton et al., 2012; Levine et al., 2016) and neural networks (Nagabandi et al., 2018), and (2) objective design like multi-step L2-norm (Luo et al., 2018), log-likelihood loss (Chua et al., 2018) and adversarial loss (Wu et al., 2019). In this regard, Chua et al. (2018) have demonstrated the superior effectiveness of using an ensemble of probabilistic models with log-likelihood loss in reducing the potential overfitting. Inspired by their success, we also use this model architecture in the design of our methods.

For model usage, MBRL methods can be roughly categorized into four groups according to the specific model usage strategies (Lai et al., 2020; Zhu et al., 2020). The first group consists of analytic-gradient algorithms that use model derivatives to search the policy by back-propagation, including Deisenroth and Rasmussen (2011); Heess et al. (2015). The second group includes shooting algorithms that directly plan forward by model predictive control (MPC) without an explicit policy (Nagabandi et al., 2018; Chua et al., 2018). The third group mainly contains model-augmented value expansion algorithms that use model-based rollouts to improve targets for model-free temporal difference (TD) updates (Buckman et al., 2018; Feinberg et al., 2018). The last group of algorithms are Dyna-style methods that use the learned model to generate simulated data to augment the real data for model-free policy training (Sutton, 1990; Luo et al., 2018). Under this taxonomy, our approaches are Dyna-style methods that are inspired by the recent MBPO (Janner et al., 2019) algorithm.

One major challenge of Dyna-style MBRL in model usage is the distribution shift problem between the simulated and the real data, and this error will only exacerbate when the learned model is used to make multi-step predictions, because of the error compounding problem. To this end, various solutions have been proposed to mitigate the distribution shift. Among them, there are mainly two types: the first one aims at improving model learning while the second one proposes to use the model more conservatively. Our algorithms take idea from both lines of works and also makes novel contributions to both.

To facilitate model learning, Asadi et al. (2019) proposed to build multi-step models to directly predict the outcome of an action sequence. Furthermore, the model predicted outputs are used to construct the model training samples in addition to the real samples (Talvitie, 2017), so that the model can generalize to its output distribution. Asadi et al. (2018) also proposed to add Lipschitz continuity constraint on the model and provided a bound on multi-step prediction error accordingly. As a comparison, FAMPO leveraged feature-based domain adaptation and proposed a model adaptation strategy to learn a feature space where real data and simulated data are close in distribution. The design principle of IAMPO is heavily inspired by domain adaptation as well. However, instead of feature-based approaches, IAMPO is more related to instance-based methods for adaptation, which often involves an estimation of importance ratio to correct for the potential domain shift.

Other works include constraining the model usage so as not to explore the regions with disastrous errors. For example, MBPO (Janner et al., 2019) used the model to generate short rollouts to avoid large departure from the real distribution in long rollouts. In a similar vein, STEVE (Buckman et al., 2018) interpolated between rollouts of various lengths according to the estimated model uncertainty. Recently, Mu et al. (2021) trained an evaluator to assess the reliability of the imagined trajectories, and M2AC (Pan et al., 2020) proposed a masking mechanism that masks the simulated data whose estimated uncertainty is high in usage phase. Our model usage component also exploits model uncertainty to reduce the possibility of selecting unreliable simulated data for policy optimization.

On the theoretical side, previous works on MBRL mostly focused on either the tabular MDPs or linear dynamics (Szita and Szepesvári, 2010; Jaksch et al., 2010; Dean et al., 2019; Simchowitz et al., 2018), but not much in the context of continuous state space and non-linear systems. Recently, Luo et al. (2018) gave a theoretical guarantee of monotonic improvement by introducing a reference policy and imposing constraints on policy optimization and model learning related to the reference policy. Janner et al. (2019) then derived a lower bound focusing on branched short rollouts, but their algorithm was a heuristic instead of being designed to maximize the lower bound. In contrast, our algorithms are naturally inspired by our theoretical results and directly minimize a derived return discrepancy upper bound without enforcing any constraint on the policy.

8. Conclusion

In this paper, we investigate how to explicitly tackle the distribution shift problem in MBRL. We first provide an upper bound of the return discrepancy to justify the necessity of model adaptation to correct the potential distribution bias in MBRL. With this insight, We then propose to incorporate model adaptation into model learning from both feature and instance perspective. In this way, the model gives more accurate predictions when generating simulated data, and therefore the follow-up policy optimization performance can be improved. Besides model adaptation learning, we additionally propose the weighted sampling strategy to further avoid the impact of inaccurate model predictions. Extensive experiments on continuous control tasks have shown the effectiveness of our work. We believe our work takes an important step towards more sample-efficient MBRL.

Acknowledgments

The Shanghai Jiao Tong University team is supported by Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102) and National Natural Science Foundation of China (62076161). The author Hang Lai is supported by Wu Wen Jun Honorary Doctoral Scholarship, AI Institute, Shanghai Jiao Tong University. The work of Han Zhao was supported in part by the Defense Advanced Research Projects Agency (DARPA) under Cooperative Agreement Number: HR00112320012, a Facebook Research Award, and Amazon AWS Cloud Credit.

Appendix A. A Different View of Analysis

In this appendix, we provide an alternative perspective on the expected return upper bound derivation.

Lemma 3 *Define the normalized state visit distribution as $\nu_T^\pi(s) := (1-\gamma)\sum_{t=0}^{\infty}\gamma^t P_{T,t}^\pi(s)$, where $P_{T,t}^\pi(s) = P(s_t = s \mid \pi, T)$. Assume the initial state distributions of the real dynamics T and the dynamics model $\hat{T}$ are the same. For any state s', assume there exists a witness function class $\mathcal{F}_{s'} = \{f : \mathcal{S}\times\mathcal{A}\to\mathbb{R}\}$ such that $\hat{T}(s' \mid \cdot,\cdot) : \mathcal{S}\times\mathcal{A}\to\mathbb{R}$ is in $\mathcal{F}_{s'}$. Then the following holds:*

$$|\nu_T^{\pi_D}(s') - \nu_{\hat{T}}^\pi(s')| \le \gamma d_{\mathcal{F}_{s'}}(\rho_T^{\pi_D}, \rho_{\hat{T}}^\pi) + \gamma\mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}\left|T(s' \mid s,a) - \hat{T}(s' \mid s,a)\right|. \tag{22}$$

Proof For the state visit distribution $\nu_{\hat{T}}^\pi(s')$, we have

$$\nu_{\hat{T}}^\pi(s') = (1-\gamma)\nu_0(s') + \gamma\int \rho_{\hat{T}}^\pi(s,a)\hat{T}(s'|s,a)\,\mathrm{d}s\,\mathrm{d}a \tag{23}$$

where ν_0 denotes the probability of the initial state being the state s'. Then we have

$$\begin{aligned}
&|\nu_T^{\pi_D}(s') - \nu_{\hat{T}}^\pi(s')| \\
&= \gamma\left|\int_{s,a} T(s'|s,a)\rho_T^{\pi_D}(s,a) - \hat{T}(s'|s,a)\rho_{\hat{T}}^\pi(s,a)\,\mathrm{d}s\,\mathrm{d}a\right| \\
&= \gamma\left|\mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}[T(s'|s,a)] - \mathbb{E}_{(s,a)\sim\rho_{\hat{T}}^\pi}[\hat{T}(s'|s,a)]\right| \\
&\le \gamma\left|\mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}[T(s'|s,a) - \hat{T}(s'|s,a)]\right| + \gamma\left|\mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}[\hat{T}(s'|s,a)] - \mathbb{E}_{(s,a)\sim\rho_{\hat{T}}^\pi}[\hat{T}(s'|s,a)]\right| \\
&\le \gamma\mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}\left|T(s'|s,a) - \hat{T}(s'|s,a)\right| + \gamma d_{\mathcal{F}_{s'}}(\rho_T^{\pi_D}, \rho_{\hat{T}}^\pi),
\end{aligned} \tag{24}$$

which completes the proof. ■

Lemma 3 states that the discrepancy between two state visit distributions for each state is bounded by the dynamics model error for predicting this state and the discrepancy between the two state-action occupancy measures. Intuitively, it means that when both the input state-action distributions and the conditional dynamics distributions are close then the output state distributions will be close as well.

Theorem 4 *Let $\mathcal{F} := \cup_{s'\in\mathcal{S}}\mathcal{F}_{s'}$ and define $\epsilon_\pi := 2d_{\mathrm{TV}}(\nu_T^\pi, \nu_T^{\pi_D})$. Under the assumption of Lemma 3, the expected return $\eta_T(\pi)$ admits the following bound:*

$$\eta_{\hat{T}}(\pi) - \eta_T(\pi) \le r_{\max}\cdot\epsilon_\pi + \gamma r_{\max}\cdot d_{\mathcal{F}}(\rho_T^{\pi_D}, \rho_{\hat{T}}^\pi)\cdot \mathit{Vol}(\mathcal{S}) \tag{25}$$

$$+ \gamma r_{\max}\cdot\mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}}\sqrt{2D_{\mathrm{KL}}(T(\cdot|s,a)\parallel\hat{T}(\cdot|s,a))}, \tag{26}$$

where $\mathit{Vol}(\mathcal{S})$ is the volume of state space $\mathcal{S}$.

Proof The return discrepancy is bounded as follows

$$
\begin{aligned}
|\eta_{\hat{T}}(\pi) - \eta_T(\pi)| &= \left| \int_{s,a} \left(\rho_T^\pi(s,a) - \rho_{\hat{T}}^\pi(s,a) \right) r(s,a) \, \mathrm{d}s \, \mathrm{d}a \right| \\
&= \left| \int_{s,a} \left(\nu_T^\pi(s)\pi(a|s) - \nu_{\hat{T}}^\pi(s)\pi(a|s) \right) r(s,a) \, \mathrm{d}s \, \mathrm{d}a \right| \\
&\leq r_{\max} \cdot \int_{s,a} \left| \nu_T^\pi(s)\pi(a|s) - \nu_{\hat{T}}^\pi(s)\pi(a|s) \right| \mathrm{d}s \, \mathrm{d}a \\
&= r_{\max} \cdot \int_s \left| \nu_T^\pi(s) - \nu_{\hat{T}}^\pi(s) \right| \mathrm{d}s \\
&= r_{\max} \cdot \int_s \left| \nu_T^{\pi_D}(s) - \nu_{\hat{T}}^\pi(s) + \nu_T^\pi(s) - \nu_T^{\pi_D}(s) \right| \mathrm{d}s \\
&\leq r_{\max} \cdot \int_s \left| \nu_T^{\pi_D}(s) - \nu_{\hat{T}}^\pi(s) \right| \mathrm{d}s + r_{\max} \cdot \epsilon_\pi
\end{aligned}
\tag{27}
$$

Replacing the above state s with the notation s', then according to Lemma 3, we have

$$
\begin{aligned}
&|\eta(\pi) - \hat{\eta}(\pi)| \\
&\leq r_{\max} \cdot \epsilon_\pi + \gamma r_{\max} \cdot \int_{s'} d_{\mathcal{F}_{s'}}(\nu_T^{\pi_D}, \nu_{\hat{T}}^\pi) \, \mathrm{d}s' + \gamma r_{\max} \cdot \mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}} \int_{s'} \left| T(s'|s,a) - \hat{T}(s'|s,a) \right| \mathrm{d}s' \\
&\leq r_{\max} \cdot \epsilon_\pi + \gamma r_{\max} \cdot d_{\mathcal{F}}(\rho_T^{\pi_D}, \rho_{\hat{T}}^\pi) \cdot \mathrm{Vol}(\mathcal{S}) + \gamma r_{\max} \cdot \mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}} \int_{s'} \left| T(s'|s,a) - \hat{T}(s'|s,a) \right| \mathrm{d}s' \\
&= r_{\max} \cdot \epsilon_\pi + \gamma r_{\max} \cdot d_{\mathcal{F}}(\rho_T^{\pi_D}, \rho_{\hat{T}}^\pi) \cdot \mathrm{Vol}(\mathcal{S}) + 2\gamma r_{\max} \cdot \mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}} d_{\mathrm{TV}}(T(\cdot|s,a), \hat{T}(\cdot|s,a)) \\
&\leq r_{\max} \cdot \epsilon_\pi + \gamma r_{\max} \cdot d_{\mathcal{F}}(\rho_T^{\pi_D}, \rho_{\hat{T}}^\pi) \cdot \mathrm{Vol}(\mathcal{S}) + \gamma r_{\max} \cdot \mathbb{E}_{(s,a)\sim\rho_T^{\pi_D}} \sqrt{2 D_{\mathrm{KL}}(T(\cdot|s,a), \hat{T}(\cdot|s,a))} \,,
\end{aligned}
\tag{28}
$$

where the last inequality holds due to Pinsker's inequality, which completes the proof. ■

Theorem 4 gives another upper bound on the return discrepancy between the return in the model and the real environment. In this bound, the first term denotes the divergence between state visit distributions induced by the policy π and the behavioral policy π_D in the environment, which is an important objective in batch reinforcement learning (Fujimoto et al., 2019) for reliable exploitation of off-policy samples. The second term corresponds to the distribution shift problem and the last term corresponds to the model estimation error on real data.

By comparing this bound to the one in Theorem 2, it seems the bound in Theorem 2 might be tighter since there is no extra ϵ_π term. However, we should notice that the assumptions made in Theorem 2 are stronger. To be more specific, we assume $G_{\hat{T}}^\pi$ satisfies the constraint in Theorem 2, while here we only assume the model $\hat{T}$ to satisfy the constraint, which is easier to hold.

Appendix B. Interchange Justification

In this appendix, we provide a proof of the interchangeability of the gradient and integration in Equation 14, which is guaranteed by the Leibniz Integral Rule (Protter et al., 1985; Talvila, 2001). Formally,

Lemma 5 *(Leibniz Integral Rule) Let $f(x,t)$ be a function such that $f(x,t)$ and its partial derivative $\partial f(x,t)/\partial x$ are continuous in t and x in some region of the $xt-$plane, including $a \le t \le b$, $x_0 \le x \le x_1$, then for $x_0 \le x \le x_1$,*

$$\frac{d}{dx}\left(\int_a^b f(x,t)\,\mathrm{d}t\right) = \int_a^b \frac{\partial}{\partial x} f(x,t)\,\mathrm{d}t. \tag{29}$$

Let $f(\theta,(s,a)) = \rho^{\pi}_{\hat{T}_\theta}(s,a)\cdot d_{\mathcal{F}}(T(\cdot|s,a),\hat{T}_\theta(\cdot|s,a))$. According to Lemma 5, now we just need to prove that i). f is continuous, and ii). the partial derivative $\partial f/\partial\theta$ is continuous in the state-action space. Note that π, $\hat{T}_\theta$ and $\partial\hat{T}_\theta/\partial\theta$ are continuous since they are constructed using neural networks. We assume that the ground-truth dynamic T and $V^{\pi}_{T} \in \mathcal{F}$ are continuous and the initial state s_0 is sampled from a continuous distribution.

Proof To prove i), f is continuous if $\rho^{\pi}_{\hat{T}_\theta}$ and $d_{\mathcal{F}}(T,\hat{T}_\theta)$ are both continuous. We first consider $\rho^{\pi}_{\hat{T}_\theta}(s,a) = (1-\gamma)\cdot\pi(a|s)\sum_{t=0}^{\infty}\gamma^t P^{\pi}_{\hat{T},t}(s)$. For $t=0$, $P^{\pi}_{\hat{T},0}(s) = P(s_0 = s)$ is continuous since s_0 is sampled from a continuous distribution. For $t \ge 1$, $P^{\pi}_{\hat{T},t}(s) = \int_{s_{t-1},a} P^{\pi}_{\hat{T},t-1}(s_{t-1})\cdot\pi(a|s_{t-1})\cdot\hat{T}_\theta(s|s_{t-1},a)\,\mathrm{d}s_{t-1}\,\mathrm{d}a$ is continuous if $P^{\pi}_{\hat{T},t-1}(s)$ is continuous, since π and $\hat{T}_\theta$ are continuous. Then by induction, $P^{\pi}_{\hat{T},t}(s)$ is continuous for $\forall t \ge 0$. Therefore, $\rho^{\pi}_{\hat{T}_\theta}(s,a)$ is continuous as a discounted sum over $P^{\pi}_{\hat{T},t}(s)\cdot\pi(a|s)$. We then consider $d_{\mathcal{F}}(T(\cdot|s,a),\hat{T}_\theta(\cdot|s,a))$,

$$\begin{aligned} d_{\mathcal{F}}(T(\cdot|s,a),\hat{T}_\theta(\cdot|s,a)) &= \sup_{f_1\in\mathcal{F}} \mathbb{E}_{s'\sim T}\left[f(s') \mid s,a\right] - \mathbb{E}_{s'\sim\hat{T}}\left[f(s') \mid s,a\right] \\ &= \sup_{f_1\in\mathcal{F}} \int_{s'} (T(s'|s,a) - \hat{T}_\theta(s'|s,a)) f_1(s')\,\mathrm{d}s'. \end{aligned} \tag{30}$$

Note that the supreme of continuous functions is still continuous, thus $d_{\mathcal{F}}(T(\cdot|s,a),\hat{T}_\theta(\cdot|s,a))$ is continuous under the assumption that T and $\hat{T}_\theta$ are continuous. So far, we have proven (i).

To prove ii), $\partial f/\partial\theta = \partial\rho^{\pi}_{\hat{T}_\theta}/\partial\theta\cdot d_{\mathcal{F}}(T,\hat{T}_\theta) + \rho^{\pi}_{\hat{T}_\theta}\cdot\partial d_{\mathcal{F}}(T,\hat{T}_\theta)/\partial\theta$. We have proven that $d_{\mathcal{F}}(T,\hat{T}_\theta)$ and $\rho^{\pi}_{\hat{T}_\theta}$ are continuous, so $\partial f/\partial\theta$ is continuous if $\partial\rho^{\pi}_{\hat{T}_\theta}/\partial\theta$ and $\partial d_{\mathcal{F}}(T,\hat{T}_\theta)/\partial\theta$ are continuous. First,

$$\frac{\partial \rho^\pi_{\hat{T}_\theta}}{\partial \theta} =(1-\gamma)\cdot \pi(a|s)\sum_{t=0}^{\infty}\gamma^t \frac{\partial P^\pi_{\hat{T},t}(s)}{\partial \theta} \tag{31}$$

$$=(1-\gamma)\cdot \pi(a|s)\sum_{t=1}^{\infty}\gamma^t \frac{\partial \int_{s_{t-1},a} P^\pi_{\hat{T},t-1}(s_{t-1})\cdot \pi(a|s_{t-1})\cdot \hat{T}_\theta(s|s_{t-1},a)\,\mathrm{d}s_{t-1}\,\mathrm{d}a}{\partial \theta} \tag{32}$$

$$=(1-\gamma)\cdot \pi(a|s)\sum_{t=1}^{\infty}\gamma^t \int_{s_{t-1},a} \pi(a|s_{t-1})\cdot \Big(\frac{\partial P^\pi_{\hat{T},t-1}(s_{t-1})}{\partial \theta}\cdot \hat{T}_\theta(s|s_{t-1},a) \tag{33}$$

$$+ P^\pi_{\hat{T},t-1}(s_{t-1})\cdot \frac{\partial \hat{T}_\theta(s|s_{t-1},a)}{\partial \theta}\Big)\,\mathrm{d}s_{t-1}\,\mathrm{d}a, \tag{34}$$

where the interchange of the integration and the gradient step in the last equation can be proven trivially via Lemma 5. Similarly by induction, we can prove that $\partial P^\pi_{\hat{T},t}(s)/\partial\theta$ is continuous for $\forall t \geq 0$. Note that the integration of a continuous function is continuous. Therefore, $\partial \rho^\pi_{\hat{T}_\theta}/\partial\theta$ is continuous since $\partial P^\pi_{\hat{T},t-1}(s_{t-1})/\partial\theta$ and $\partial\hat{T}_\theta/\partial\theta$ are continuous. Secondly,

$$\frac{\partial d_{\mathcal{F}}(T,\hat{T}_\theta)}{\partial\theta} = \frac{\partial \sup_{f_1\in\mathcal{F}}\int_{s'}(T(s'|s,a)-\hat{T}_\theta(s'|s,a))f_1(s')\,\mathrm{d}s'}{\partial\theta} \tag{35}$$

$$= \frac{\partial \int_{s'}(T(s'|s,a)-\hat{T}_\theta(s'|s,a))f_1^*(s')\,\mathrm{d}s'}{\partial\theta} \tag{36}$$

$$= \int_{s'} -f_1^*(s')\frac{\partial \hat{T}_\theta(s'|s,a)}{\partial\theta}\,\mathrm{d}s', \tag{37}$$

where $f_1^* = \arg\max_{f_1\in\mathcal{F}}\int_{s'}(T(s'|s,a)-\hat{T}_\theta(s'|s,a))f_1(s')\,\mathrm{d}s'$. Here we can directly take the supremum since $\mathcal{F} := \left\{f : \|f\|_\infty \leq \frac{r_{\max}}{1-\gamma}\right\}$ is closed. And the supremum is almost surely unique, since the supremum simply lets $f_1^* = r_{\max}/(1-\gamma)$ when $T(s'|s,a)-\hat{T}_\theta(s'|s,a)$ is positive, and $f_1^* = -r_{\max}/(1-\gamma)$ otherwise. The interchange of the integration and the gradient step from (8) to (9) can be proven trivially via Lemma 5 again. Similarly, as $\partial\hat{T}_\theta/\partial\theta$ is continuous, $\partial d_{\mathcal{F}}(T,\hat{T}_\theta)/\partial\theta$ is continuous too, which completes the proof of ii). Combining i) and ii) completes the proof. ■

Appendix C. Design Evaluation

In this appendix, we evaluate the design choices for each part of our algorithm. More specifically, we compare different choices of feature alignment metrics, importance ratio estimation methods, and sampling strategy schemes.

C.1 MMD Variant of FAMPO

Besides Wasserstein-1 distance, we can use other distribution divergence metrics to align the features. MMD is another instance of IPM when the witness function class is the unit

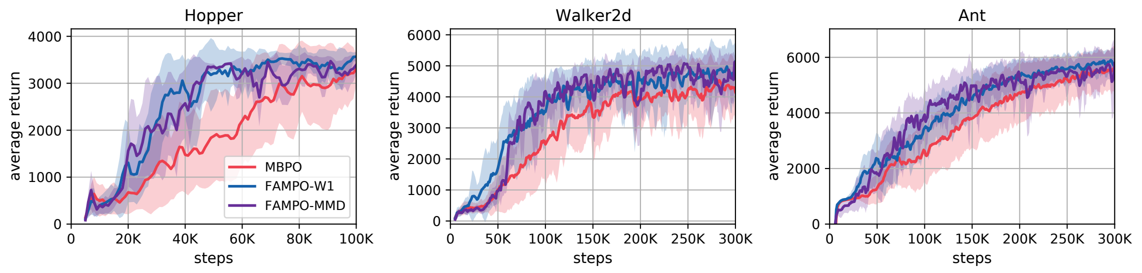


Figure 6: Learning curves of FAMPO using different metrics (Wasserstein-1 distance and MMD).

ball in a reproducing kernel Hilbert space (RKHS). Let k be the kernel of the RKHS $\mathcal{H}_k$ of functions on $\mathcal{X}$. Then the squared MMD in $\mathcal{H}_k$ between two feature distributions $\mathbb{P}_{h_e}$ and $\mathbb{P}_{h_m}$ is Gretton et al. (2012):

$$\mathrm{MMD}_k^2(\mathbb{P}_{h_e}, \mathbb{P}_{h_m}) := \mathbb{E}_{h_e, h'_e}[k(h_e, h'_e)] + \mathbb{E}_{h_m, h'_m}[k(h_m, h'_m)] - 2\mathbb{E}_{h_e, h_m}[k(h_e, h_m)], \tag{38}$$

which is a non-parametric measurement based on kernel mappings. In practice, given finite feature samples from distributions $\{h_e^1, \cdots, h_e^{N_e}\} \sim \mathbb{P}_{h_e}$ and $\{h_m^1, \cdots, h_m^{N_m}\} \sim \mathbb{P}_{h_m}$, where N_e and N_m are the number of real samples and simulated ones, one unbiased estimator of $\mathrm{MMD}_k^2(\mathbb{P}_{h_e}, \mathbb{P}_{h_m})$ can be written as follows:

$$\mathcal{L}_{\mathrm{MMD}}(\theta_g) = \frac{1}{N_e(N_e-1)} \sum_{i \neq i'} k(h_e^i, h_e^{i'}) + \frac{1}{N_m(N_m-1)} \sum_{j \neq j'} k(h_m^j, h_m^{j'}) - \frac{2}{N_e N_m} \sum_{i=1}^{N_e} \sum_{j=1}^{N_m} k(h_e^i, h_m^j). \tag{39}$$

To achieve model adaptation through MMD, we optimize the feature extractor to minimize the above adaptation loss $\mathcal{L}_{\mathrm{MMD}}$ with real (s, a) data and simulated one as input. When implementing the MMD variant, choosing optimal kernels remains an open problem and we use a linear combination of eight RBF kernels with bandwidths $\{0.001, 0.005, 0.01, 0.05, 0.1, 1, 5, 10\}$. The results are shown in Figure 6. Note that we only compare different metrics for feature alignment here and do not incorporate the weighted sampling strategy. We can observe that using MMD as the distribution divergence measure is also effective in the FAMPO framework, only slightly worse than using Wasserstein-1 distance on Hopper.

C.2 Classification Variant of IAMPO

We compare different methods to estimate the importance ratio for IAMPO: DICE network and binary classification, as introduced in Section 4.2. The results are shown in Figure 7. Similarly, here we do not use weighted sampling strategy. While IAMPO with binary classification achieves better performance than MBPO, it is not as effective as DICE, mainly due to the imbalance between the amount of real data and simulated data.

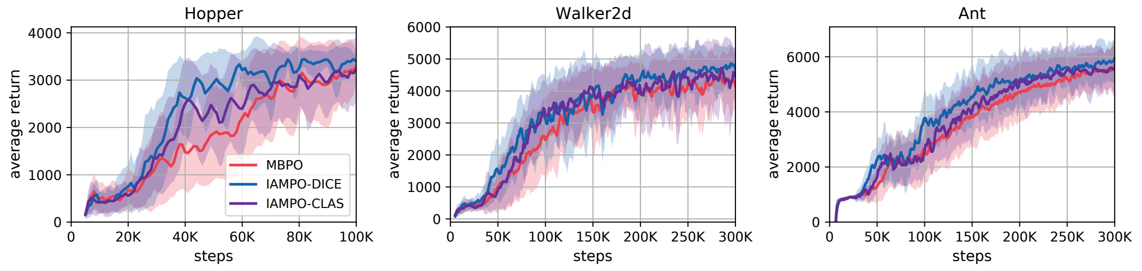


Figure 7: Learning curves of IAMPO with DICE network (IAMPO-DICE) and IAMPO with classification (IAMPO-CLAS).

C.3 Masking Scheme of Sampling Strategy

We also use the same uncertainty estimation method as in weighted sampling strategy to implement the masking scheme(Pan et al., 2020). To be more specific, after estimating the uncertainty, we directly mask half the simulated data with high uncertainty in $\mathcal{D}_{\text{model}}$ and use the remaining data to train the policy. In practice, we assure that after the masking scheme, the size of $\mathcal{D}_{\text{model}}$ is still the same as MBPO by increasing the rollout batch size. The preliminary results on Hopper and Walker2d are shown in Figure 8. We find that the masking scheme cannot achieve better performance than MBPO. The reason may be that after masking half the simulated data, only those very close to the real data are reserved to train the policy, as analyzed in Section 5. This observation further verifies the effectiveness weighted sampling strategy.

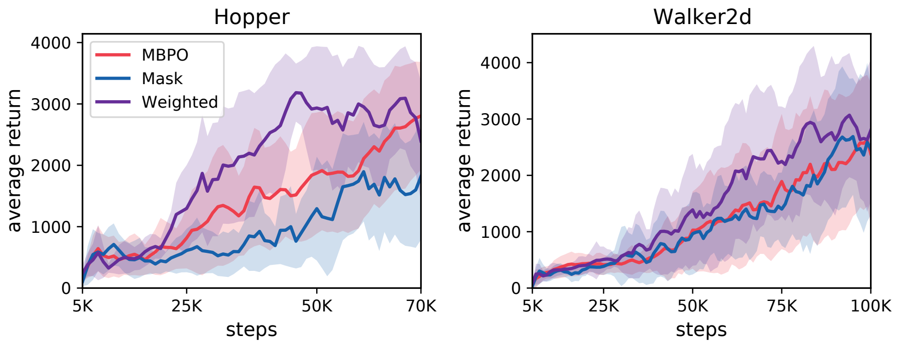
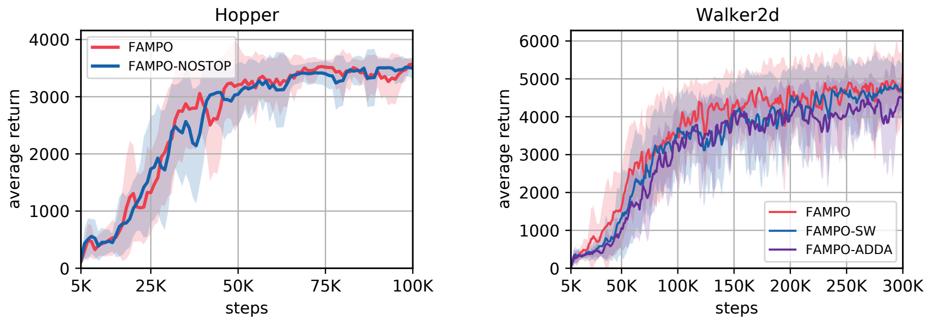


Figure 8: Comparison results of original MBPO to MBPO with masking scheme and weighted sampling strategy.



(a) Early stopping. (b) Adaptation strategy.

Figure 9: More empirical analysis for FAMPO. (a) FAMPO-NOSTOP denotes the FAMPO variant without early stopping the model adaptation procedure. (b) FAMPO-SW denotes the FAMPO variant of sharing the feature extractor weights for two data distributions. FAMPO-ADDA denotes the FAMPO variant of fixing the feature extractor of real data.

Appendix D. More Experimental Results for FAMPO

D.1 Model Adaptation Early Stopping

In practice, we find that after a certain number of environment steps, the model loss difference between FAMPO and MBPO becomes small. So in FAMPO, we early stop the model adaptation procedure after collecting a certain number of real data, such as 40K in the Hopper environment. We then conduct experiments without early stopping model adaptation, and the results are demonstrated in Figure 9(a). We find that keeping adapting the dynamics model throughout the whole learning process does not bring performance improvement. This indicates that model adaptation makes a difference only when the model training data is insufficient. So we set a model adaptation early stopping epoch for each environment (see Table 2 for detail) to improve the computation efficiency.

D.2 Adaptation Strategy

In FAMPO, we untie the feature extractor weights for two data distributions and learn the two feature extractors simultaneously, which is a variant of the adaptation strategy in Adversarial Discriminative Domain Adaptation (ADDA) (Tzeng et al., 2017). Differently, in ADDA the feature mapping for source domain (*i.e.* real data) is fixed. Another alternative is to share the feature extractor weights between the two data distributions. From the comparison in Figure 9(b), we observe that the performance of these three adaptation strategies differs not much, but FAMPO performs slightly better. The reason may be that, different from general domain adaptation, in MBRL scenarios, the real and simulated data will be collected and generated continuously. Therefore learning a feature extractor specified for the simulated data at each iteration is unnecessary.

Appendix E. Hyperparameter Settings

Table 1 lists the common hyperparameters used in FAMPO and IAMPO. Table 2 and Table 3 list the distinct hyperparameters of FAMPO and IAMPO, respectively.

Table 1: Common hyperparameters for FAMPO and IAMPO.

		Inverted Pendulum	Swimmer	Hopper	Walker2d	Ant	Half Cheetah
	network architecture	MLP with four hidden layers of size 200 feature extractor: four hidden layers; decoder: one output layer					
	real samples for model pretraining	300	2000	5000			
	real steps per epoch	250	1000				
E	real steps between model training	125	250				
F	model rollout batch size	100000					
B	ensemble size	7					
G_3	policy updates per real step	30	20				40

Table 2: Distinct hyperparameters for FAMPO. $[a, b, x, y]$ denotes a linear function, *i.e.* at epoch e, $f(e) = \min(\max(x + \frac{e-a}{b-a} \cdot (x - y), x), y)$.

		Inverted Pendulum	Swimmer	Hopper	Walker2d	Ant	Half Cheetah
	model adaptation batch size	64	256				
k	rollout length	1	1	[5,45,1,15]	1	[10,60,1,25]	[1,30,1,5]
G_2	model adaptation updates	6	40	400	1000	3000	[1,30,100,1000]
	model adaptation early stop epoch	6	6	40	80	60	30
σ_0	temperature in weighted sampling	1	0	1	1	0.2	0
σ_1	temperature in weighted sampling	20	0	1	50	30	0

Table 3: Distinct hyperparameters for IAMPO. $[a, b, x, y]$ denotes a linear function, *i.e.* at epoch e, $f(e) = \min(\max(x + \frac{e-a}{b-a} \cdot (x - y), x), y)$.

		Inverted Pendulum	**Swimmer**	**Hopper**	**Walker2d**	**Ant**	**Half Cheetah**
	DICE training batch size	256					
k	rollout length	1	3	[10,60,1,15]	1	[10,60,1,25]	5
G_2	DICE training updates	100	100	800	600	300	450
α_0	ratio clipping minimum value	0.1					
α_1	ratio clipping maximum value	5			20		5
λ	coefficient in DICE loss	0.1			3	5	0.1
σ_0	temperature in weighted sampling	2	5	1	2	0.1	0
σ_1	temperature in weighted sampling	20	20	40	50	7.5	0

References

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017.

Kavosh Asadi, Dipendra Misra, and Michael Littman. Lipschitz continuity in model-based reinforcement learning. In *International Conference on Machine Learning*, pages 264–273, 2018.

Kavosh Asadi, Dipendra Misra, Seungchan Kim, and Michel L Littman. Combating the compounding-error problem with a multi-step model. *arXiv preprint arXiv:1905.13320*, 2019.

Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *Advances in neural information processing systems*, pages 137–144, 2007.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010.

Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. *arXiv preprint arXiv:1801.01401*, 2018.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Jacob Buckman, Danijar Hafner, George Tucker, Eugene Brevdo, and Honglak Lee. Sample-efficient reinforcement learning with stochastic ensemble value expansion. In *Advances in Neural Information Processing Systems*, pages 8224–8234, 2018.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems*, pages 4754–4765, 2018.

Sarah Dean, Horia Mania, Nikolai Matni, Benjamin Recht, and Stephen Tu. On the sample complexity of the linear quadratic regulator. *Foundations of Computational Mathematics*, pages 1–47, 2019.

Marc Deisenroth and Carl E Rasmussen. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472, 2011.

Amir-massoud Farahmand. Iterative value-aware model learning. In *Advances in Neural Information Processing Systems*, pages 9072–9083, 2018.

Vladimir Feinberg, Alvin Wan, Ion Stoica, Michael I Jordan, Joseph E Gonzalez, and Sergey Levine. Model-based value estimation for efficient model-free reinforcement learning. *arXiv preprint arXiv:1803.00101*, 2018.

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

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3389–3396. IEEE, 2017.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of Wasserstein GANs. In *Advances in neural information processing systems*, pages 5767–5777, 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

Nicolas Heess, Gregory Wayne, David Silver, Timothy Lillicrap, Tom Erez, and Yuval Tassa. Learning continuous control policies by stochastic value gradients. *Advances in Neural Information Processing Systems*, 28:2944–2952, 2015.

Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in neural information processing systems*, pages 4565–4573, 2016.

Thomas Jaksch, Ronald Ortner, and Peter Auer. Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11(Apr):1563–1600, 2010.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, pages 12498–12509, 2019.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Bartosz Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, 2016.

Hang Lai, Jian Shen, Weinan Zhang, and Yong Yu. Bidirectional model-based policy optimization. In *International Conference on Machine Learning*, pages 5618–5627. PMLR, 2020.

Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. *arXiv preprint arXiv:1907.02057*, 2019.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

Yuping Luo, Huazhe Xu, Yuanzhi Li, Yuandong Tian, Trevor Darrell, and Tengyu Ma. Algorithmic framework for model-based deep reinforcement learning with theoretical guarantees. *arXiv preprint arXiv:1807.03858*, 2018.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

Yao Mu, Yuzheng Zhuang, Bin Wang, Guangxiang Zhu, Wulong Liu, Jianyu Chen, Ping Luo, Shengbo Li, Chongjie Zhang, and Jianye Hao. Model-based reinforcement learning via imagination with derived memory. *Advances in Neural Information Processing Systems*, 34, 2021.

Alfred Müller. Integral probability metrics and their generating classes of functions. *Advances in Applied Probability*, 29(2):429–443, 1997.

Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. DualDICE: Behavior-agnostic estimation of discounted stationary distribution corrections. *arXiv preprint arXiv:1906.04733*, 2019.

Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566. IEEE, 2018.

Feiyang Pan, Jia He, Dandan Tu, and Qing He. Trust the model when it is confident: Masked model-based actor-critic. *Advances in Neural Information Processing Systems*, 33, 2020.

Murray H Protter, Charles B Morrey, Murray H Protter, and Charles B Morrey. Differentiation under the integral sign. improper integrals. the gamma function. *Intermediate Calculus*, pages 421–453, 1985.

R Tyrrell Rockafellar. *Convex analysis*, volume 36. Princeton university press, 1970.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897, 2015.

Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu. Wasserstein distance guided representation learning for domain adaptation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Jian Shen, Han Zhao, Weinan Zhang, and Yong Yu. Model-based policy optimization with unsupervised model adaptation. *Advances in Neural Information Processing Systems*, 33, 2020.

Max Simchowitz, Horia Mania, Stephen Tu, Michael I Jordan, and Benjamin Recht. Learning without mixing: Towards a sharp analysis of linear system identification. *arXiv preprint arXiv:1802.08334*, 2018.

Bharath K Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Bernhard Schölkopf, and Gert RG Lanckriet. On integral probability metrics, ϕ-divergences and binary classification. *arXiv preprint arXiv:0901.2698*, 2009.

Wen Sun, Nan Jiang, Akshay Krishnamurthy, Alekh Agarwal, and John Langford. Model-based reinforcement learning in contextual decision processes. *arXiv preprint arXiv:1811.08540*, 2018.

Richard S Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine Learning Proceedings 1990*, pages 216–224. Elsevier, 1990.

Richard S Sutton, Csaba Szepesvári, Alborz Geramifard, and Michael P Bowling. Dyna-style planning with linear function approximation and prioritized sweeping. *arXiv preprint arXiv:1206.3285*, 2012.

István Szita and Csaba Szepesvári. Model-based reinforcement learning with nearly tight exploration complexity bounds. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 1031–1038, 2010.

Erik Talvila. Necessary and sufficient conditions for differentiating under the integral sign. *The American Mathematical Monthly*, 108(6):544–548, 2001.

Erik Talvitie. Model regularization for stable sample rollouts. In *UAI*, pages 780–789, 2014.

Erik Talvitie. Self-correcting models for model-based reinforcement learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017.

Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

Yueh-Hua Wu, Ting-Han Fan, Peter J Ramadge, and Hao Su. Model imitation for model-based reinforcement learning. *arXiv preprint arXiv:1909.11821*, 2019.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33, 2020.

Bianca Zadrozny. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the twenty-first international conference on Machine learning*, page 114, 2004.

Ruiyi Zhang, Bo Dai, Lihong Li, and Dale Schuurmans. GenDICE: Generalized offline estimation of stationary values. *arXiv preprint arXiv:2002.09072*, 2020a.

Shangtong Zhang, Bo Liu, and Shimon Whiteson. GradientDICE: Rethinking generalized offline estimation of stationary values. In *International Conference on Machine Learning*, pages 11194–11203. PMLR, 2020b.

Han Zhao, Remi Tachet des Combes, Kun Zhang, and Geoffrey J Gordon. On learning invariant representation for domain adaptation. *arXiv preprint arXiv:1901.09453*, 2019.

Guangxiang Zhu, Minghao Zhang, Honglak Lee, and Chongjie Zhang. Bridging imagination and reality for model-based deep reinforcement learning. *Advances in Neural Information Processing Systems*, 33, 2020.