ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240808

Offline Model-Based Reinforcement Learning with Anti-Exploration

Padmanaba Srinivasan [1,*] **and William Knottenbelt** [1]

[1]Department of Computing, Imperial College London

Abstract. Model-based reinforcement learning (MBRL) algorithms learn a dynamics model from collected data and apply it to generate synthetic trajectories to enable faster learning. This is an especially promising paradigm in offline reinforcement learning (RL) where data may be limited in quantity, in addition to being deficient in coverage and quality. Practical approaches to *offline* MBRL usually rely on ensembles of dynamics models to prevent exploitation of any individual model and to extract uncertainty estimates that penalize values in states far from the dataset support. Uncertainty estimates from ensembles can vary greatly in scale, making it challenging to generalize hyperparameters well across even similar tasks. In this paper, we present Morse Model-based offline RL (MoMo), which extends the *anti-exploration* paradigm found in offline model-free RL to the model-based space. We develop model-free and model-based variants of MoMo and show how the model-free version can be extended to detect and deal with out-of-distribution (OOD) states using explicit uncertainty estimation without the need for large ensembles. MoMo performs offline MBRL using an anti-exploration bonus to counteract value overestimation in combination with a policy constraint, as well as a truncation function to terminate synthetic rollouts that are excessively OOD. Experimentally, we find that both model-free and model-based MoMo perform well, and the latter outperforms prior model-based and model-free baselines on the majority of D4RL datasets tested.

1 Introduction

Reinforcement learning (RL) aims to learn policies for sequential decision-making that maximize an expected reward [43]. In online RL, this means alternating between interacting with the environment to collect new data and then improving the policy using previously collected data. Model-based reinforcement learning (MBRL) denotes techniques in which an established environment model or an approximation of the environment is used to simulate policy rollouts without having to directly query the environment [41, 42, 43, 30].

Offline RL tackles problems where the policy cannot interact with and explore the real environment but instead can only access a static dataset of trajectories. The offline dataset can be composed of mixed-quality trajectories with poor coverage of the state-action space. With no ability to extrapolate beyond the offline dataset, standard model-free offline RL algorithms often apply systematic pessimism by either injecting penalties into the estimation of values or by constraining the policy [32, 27] to remain in action support. Offline MBRL can offer more flexibility by augmenting the offline dataset with additional synthetic rollouts that expand coverage, while also remaining within the dataset support [22, 49, 50].

MBRL is fraught with danger: approximation errors in the dynamics model can cause divergence in temporal difference updates [45], errors can accumulate over multistep rollouts, resulting in performance deterioration if unaddressed [18], and the dynamics model can be exploited by the agent [33, 26, 6]. In offline MBRL, these issues are amplified due to the double dose of extrapolation error present in both dynamics and value functions. Consequently, standard online MBRL methods perform poorly when directly applied offline [49].

Offline RL approaches typically fall into one of two categories: 1) *critic regularization* methods which penalize OOD action-values or 2) *policy constraint* methods that minimize a divergence between the learned policy and (a potentially estimated) behavior policy. Studying model-free offline RL literature exposes a variety of approaches in both paradigms [32, 27], but offline MBRL methods exhibit far less diversity [17].

Many previous offline MBRL algorithms are based on MBPO [18] and perform conservative value updates that identify the increased epistemic uncertainty in out-of-distribution (OOD) regions using dynamics ensembles [6] or actively penalize all action-values from synthetic samples [50]. Ultimately, these fall firmly under the critic regularization paradigm, and their empirical evaluation reveals that they perform little better than their model-free counterparts [2, 14]. A small number of methods implement policy-constrained offline MBRL [51, 20], though these do not realize the benefits of augmenting training with samples from the dynamics model.

Our approach to offline MBRL is based on the *anti-exploration* [35] paradigm originating in model-free offline RL which uses a policy constraint in combination with an anti-exploration bonus to curb value overestimation. We train a Morse neural network [7] and exploit its properties as an implicit behavior model, and an OOD detector to extend model-free anti-exploration for MBRL, which realizes the benefits of both policy-constrained offline RL **and** training with synthetic data.

Contributions In this work, we present Morse Model-based offline RL (MoMo) which performs offline model-free RL using anti-exploration. MoMo trains a dynamics model and a Morse neural network model, the latter of which is an energy-based model that learns an *implicit* behavior policy and an OOD detector. We demonstrate how this can be incorporated into a model-free anti-exploration-based offline RL method and then adapt it to detect and truncate rollouts in the dynamics model that are becoming excessively out-of-distribution. The Morse neural network's uncertainty estimation

* Corresponding Author. Email: ps3416@imperial.ac.uk.
Extended version available at https://arxiv.org/abs/2408.10713

helps us avoid the need for large dynamics ensembles – we can instead use a single dynamics model. Furthermore, the Morse network's calibrated uncertainty estimates allow for better hyperparameter generalization across similar tasks.

Our experiments evaluate both model-free and model-based versions of MoMo on the D4RL Gym Locomotion and Adroit tasks [9] and establish MoMo's performance to be equivalent to or better than recent baselines in 17 of 24 datasets. In-depth ablation experiments analyze MoMo's hyperparameter sensitivity and the effectiveness of anti-exploration value penalties in offline MBRL.

2 Related Work

Reinforcement Learning We consider RL in the form of a Markov Decision Process (MDP), $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, R, T, P_0, \gamma\}$, which consists of a tuple with state space $\mathcal{S}$, action space $\mathcal{A}$, a scalar reward $R(s,a)$, transition dynamics T, initial state distribution P_0 and discount factor $\gamma \in [0,1)$ [43]. RL aims to learn a policy π that will execute an action $a = \pi(s)$ that maximizes the expected discounted reward $J(\pi) = \mathbb{E}_{\tau\sim T_\pi(\tau)}\left[\sum_{t=0}^{T}\gamma_t R(s_t,a_t)\right]$ where $T_\pi(\tau) = P_0(s_0)\prod_{t=0}^{T}\pi(a_t \mid s_t)T(s_{t+1} \mid s_t, a_t)$ is the trajectory under π.

Model-Free Offline RL Offline RL methods are designed to learn from a static dataset $\mathcal{D}$, which consists of interactions collected by applying a policy β, without access to the environment [27, 12, 39]. The offline dataset may provide poor coverage of the state-action space, and so when using function approximators, the policy must be constrained to select the actions within the support of the dataset. Offline RL methods (generally) belong to one of two families of approaches: 1) critic regularization or 2) policy constraint [32, 27]. Uncertainty-based model-free methods use large ensembles [2, 14] to perform a variation of critic regularization.

Anti-Exploration Rezaeifar et al. [35] pose model-free offline RL as an exercise in *anti-exploration*, an approach in which the actor is constrained via an explicit divergence minimization term in the policy improvement step and policy evaluation is augmented with an anti-exploration bonus (value penalty) to counteract overestimation in off-policy evaluation. This bears some similarity to the BRAC framework [47] which regularizes (either policy or value) using an explicit divergence between the current and estimated behavior policies.

Several works propose different heuristics for estimating divergence in anti-exploration algorithms [31, 44], though thus far anti-exploration has remained exclusive to model-free methods.

Offline Model-Based RL Offline MBRL injects samples from a learned dynamics model into the RL learning process and leverages the diversity of simulated rollouts to augment the offline dataset with additional data. In online MBRL, Janner et al. [18] introduce MBPO which uses an ensemble of dynamics models and performs Dyna-style policy learning [41, 42]. MBPO's ensemble does not discourage OOD exploration; Yu et al. [49] address this by penalizing reward with ensemble uncertainty and Kidambi et al. [22] use ensemble uncertainty to detect, penalize and terminate synthetic rollouts when they are excessively OOD. Several works develop methods to better estimate and penalize values using different uncertainty estimation techniques [40, 19].

COMBO [50] performs CQL-like critic regularization [25] on samples from model rollouts. This adversarial approach to offline MBRL is further developed by Rigter et al. [36], Liu et al. [28], Bhardwaj et al. [5] and Luo et al. [29] who present alternative approaches to learning pessimistic value functions.

Policy constraint-based offline MBRL embraces an overarching goal to direct the policy towards states with known dynamics when starting in unknown ones. Zhang et al. [51] train ensemble dynamics and a behavioral transition model that is used to ensure that the next state the policy will transition to is within the dataset support. Jiang et al. [20] train an inverse dynamics model that constrains the policy to stay close to the behavior policy. Neither method performs policy rollouts using dynamics models, as estimates of the behavior policy do not extrapolate well. Their lack of training on synthetic data puts these methods at odds with critic-regularized offline MBRL.

3 Preliminaries

Dherin et al. [7] develop the Morse neural network, an uncertainty estimator that produces an unnormalized density $M(x) \in [0,1]$ in an embedding space $\mathbb{R}^e$. At its modes, the density achieves a value of 1 and approaches 0 with increasing distance from the mode. A Morse neural network is defined by the combination of a neural network function approximator and a Morse kernel:

Definition 1 (Morse Kernel)**.** *A Morse Kernel is a positive definite kernel K that is applied on a space $Z = \mathbb{R}^e$, $K(z_1, z_2)$ and takes its values in the interval $[0,1]$ with $K(z_1,z_2) = 1$ only when $z_1 = z_2$.*

All kernels that use a measure of divergence are Morse kernels [1]. A commonly used kernel is the radial basis function (RBF) kernel with scale parameter $\lambda > 0$

$$K(z,t) = e^{-\lambda||z-t||^2}. \tag{1}$$

The Morse kernel describes how close an embedding z is to the target t. The relationship between the input x and its corresponding embedding z is determined by a neural network $f_\psi : X \to Z$, $X \in \mathbb{R}^d$ and $Z \in \mathbb{R}^e$ with parameters ψ. Combining the Morse kernel with a neural network yields the Morse neural network:

Definition 2 (Morse Neural Network)**.** *A Morse neural network is comprised of a function $f_\psi : X \to Z$ and a Morse Kernel $K(z,t)$ where $t \subset Z$ is a target embedding where its size and value are chosen as a model hyperparameter. The Morse neural network is given by $M_\psi(x) = K(f_\psi(x), t)$.*

Using Definitions 1 and 2, if x maps to a mode in the level set of the submanifold Z, then $M_\psi(x) = 1$. We interpret $M_\psi(x)$ as the *certainty* that the sample x is from the training dataset and $1 - M_\psi(x)$ is an estimate of the epistemic uncertainty of x. The function $-\log M_\psi(x)$ measures measures the distance between z and the nearest mode at t of the set of modes M:

$$d(z) = \min_{t\in M} \quad d(z,t), \tag{2}$$

The trained Morse neural network offers the following properties [7]:

1. $M_\psi(x) \in [0,1]$.
2. $M_\psi(x) = 1$ at all mode submanifolds.
3. $-\log M_\psi(x) \geq 0$ is a squared distance that satisfies the Morse–Bott non-degeneracy condition at the mode submanifolds [4].
4. $M_\psi(x)$ is an exponentiated squared distance, the function is distance aware in the sense that as $f_\psi(x) \to t$, $M_\psi(x) \to 1$.

See Dherin et al. [7] for the proof of each property.

Algorithm 1 Morse Neural Network Training

Input: Offline dataset $\mathcal{D} = \{s, a, r, s'\}$
Initialize: Initialize Morse network M_ψ.
for t = 1 **to** T **do**
  $(s, a) \sim \mathcal{D}$ ▷ Sample real state-action tuples
  $a_u \sim \mathcal{D}_{\text{uni}}$ ▷ Sample random actions
  Update ψ using Equation 4
end for
return M_ψ

4 Morse Model-Based Offline RL

In this section, we describe our approach to policy-constrained offline MBRL that can use samples from synthetic rollouts to augment the offline dataset. We begin by adapting the Morse neural network for offline RL as a method for anti-exploration. Then in Section 4.3 we show how this can be extended to MBRL.

4.1 Morse Neural Networks Learn an Implicit Behavior Policy

Morse Networks for Offline RL The offline RL dataset consists of K tuples $\mathcal{D} = \{s, a, r, s'\}_{k=1}^{K}$ that provide partial coverage of the state-action space. We train a Morse neural network to learn a conditional uncertainty model over actions in the dataset. Adapting the objective from Dherin et al. [7], we minimize the KL divergence $D_{\text{KL}}(\mathcal{D}(s, a) \,||\, M_\psi(s, a))$ using:

$$\min_\psi \mathbb{E}_{s,a\sim\mathcal{D}} \left[\log \frac{\mathcal{D}(s, a)}{M_\psi(s, a)}\right] + \int M_\psi(s, a) - \mathcal{D}(s, a)\, da, \quad (3)$$

optimizing with respect to ψ, we minimize the empirical loss:

$$\min_\psi \quad \mathbb{E}_{s,a\sim\mathcal{D}} \left[-\log K(f_\psi(s, a), t)\right] + \mathbb{E}_{\substack{s\sim\mathcal{D} \\ a_{\text{u}}\sim\mathcal{D}_{\text{uni}}}} \left[K(f_\psi(s, a_{\text{u}}), t)\right], \quad (4)$$

where $\mathcal{D}_{\text{uni}}$ denotes a uniform distribution over the action space. We outline the Morse neural network training procedure in Algorithm 1.

The Morse neural network is an energy-based model (EBM) [15, 7]:

Proposition 1. *The Morse neural network is an EBM:* $E_\psi(x) = -\log M_{\psi(x)}$.

See Dherin et al. [7] for the proof.

By interpreting the Morse neural network as an EBM we can recover a behavior policy density using the Boltzmann distribution:

$$\beta(a|s) = \frac{M_\psi(s, a)}{Z_\psi(s)} \quad (5)$$

$$Z_\psi(s) = \int_{\mathcal{A}} M_\psi(s, a)\, da, \quad (6)$$

where $Z_\psi(s)$ is a per-state normalization constant that, in practice, is intractable for continuous actions. This yields an implicit behavior policy:

$$\log \beta(a|s) = \log M_\psi(s, a) - \log Z_\psi(s) \quad (7)$$

$$\leq \log M_\psi(s, a) \leq 0, \quad (8)$$

where the logarithm of the Morse neural network is an upper bound on the behavior density. Hence, maximizing certainty (or conversely, minimizing the distance $-\log M_\psi$ in the submanifold to a mode) will maximize log-likelihood under the implicit behavior policy. This also has the distinct advantage that all modes of the dataset occur with the same unnormalized density of one with no limiting inductive bias about the number of modes/policies that produced the training dataset.

Many prior offline model-free methods estimate and constrain using explicit behavior policies [12, 47]. We perform similar constrained policy optimization with changes highlighted in blue:

$$\max_\pi \quad \mathbb{E}_{\substack{s\sim\mathcal{D} \\ a\sim\pi}} \left[Q_\theta(s, a) - D_{KL}(\pi \,||\, \beta)\right] \quad (9)$$

$$\approx \max_\pi \quad \mathbb{E}_{\substack{s\sim\mathcal{D} \\ a\sim\pi}} \left[Q_\theta(s, a) + \log M_\psi(s, a)\right]. \quad (10)$$

This policy optimization is identical to that of prior anti-exploration methods [34, 44, 31], where optimizing $\log M_\psi$ allows minimization of the reverse KL divergence.

Following prior anti-exploration approaches, we use the following value function update with the anti-exploration bonus in red:

$$\min_\theta \quad \mathbb{E}_{\substack{s,a,s'\sim\mathcal{D} \\ a'\sim\pi}} \left[(Q_\theta(s, a) - y)^2\right]$$

$$y = r + \gamma(Q_{\bar\theta}(s', a') + \log M_\psi(s', a')), \quad (11)$$

where θ denotes the parameters of the Q function and $\bar\theta$ the parameters of the target Q function.

Note that in Equation 10 and 11 we disregard the scalar multiplier typically included with the $\log M_\psi$ term. We implicitly set this to one and instead use the kernel scale λ in Equation 1 to control the behavioral cloning and anti-exploration tradeoff.

Control with Kernel Scale We illustrate how λ can control the strength of the constraint with a didactic example in Figure 1. We select eight, two-dimensional action modes spaced uniformly on a unit circle that are assigned to one of two states in an alternating fashion. 64 action samples are drawn for actions in each state with a standard deviation of 0.01. With a total of 128 samples from two states with four action modes each, we produce a synthetic dataset that exhibits a strong multimodal behavior policy (see Figure 1(a)). We fit a Morse neural network to the dataset using the objective in Equation 4 and evaluate 16 000 uniformly spaced samples over the action space to produce unnormalized densities for each state for different λ in Figure 1. As λ increases, the degree of "closeness" in the embedding space decreases resulting in a more sharply constraining density.

Analysis We aim to constrain the policy to match the occupancy measure of the behavior policy. Assuming deterministic transition dynamics, we consider the normalized occupancy measure of the behavior policy using an EBM (see Proposition 1):

$$(1 - \gamma)\rho_\beta(s, a) = \frac{\exp(-E_\psi(s, a))}{Z_\psi(s)}. \quad (12)$$

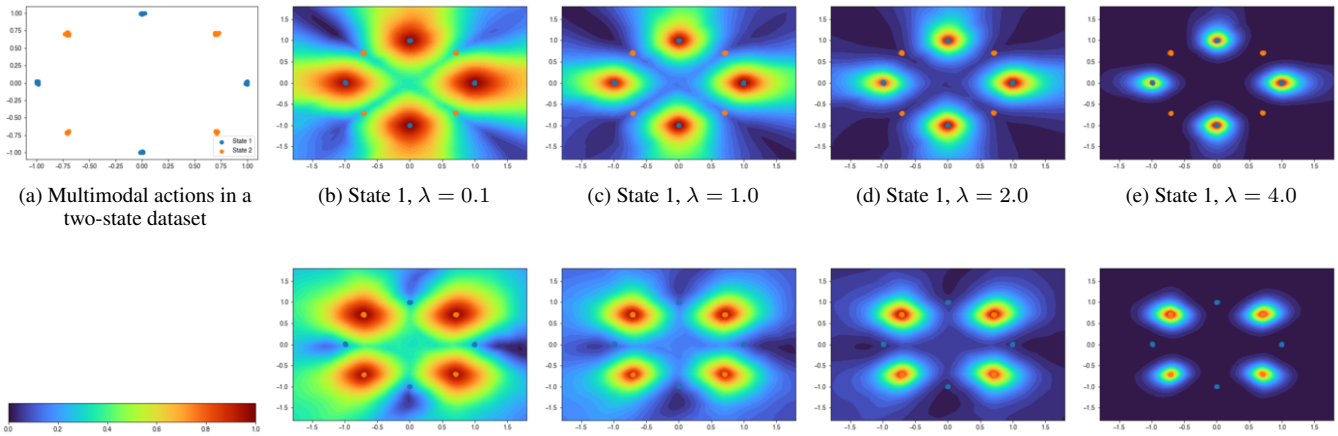
We minimize the KL divergence between occupancies and show that the policy constraint term in Equation 10 upper-bounds the divergence:

$$D_{\text{KL}}(\rho_\pi \,||\, \rho_\beta) = \sum_{s,a} \rho_\pi(s, a) \frac{\log \rho_\pi(s, a)}{\log \rho_\beta(s, a)} \quad (13)$$

$$\leq \mathbb{E}_\pi \left[-\log M_\psi(s, a)\right] - \mathcal{H}(\pi) + \text{const}, \quad (14)$$

therefore, to minimize the KL divergence we choose to minimize the upper bound:

$$\arg\min_\pi D_{\text{KL}}(\rho_\pi \,||\, \rho_\beta) \rightarrow \arg\min_\pi \mathbb{E}_\pi \left[-\log M_\psi(s, a)\right]. \quad (15)$$



(a) Multimodal actions in a two-state dataset (b) State 1, $\lambda = 0.1$ (c) State 1, $\lambda = 1.0$ (d) State 1, $\lambda = 2.0$ (e) State 1, $\lambda = 4.0$

(f) Density colormap (g) State 2, $\lambda = 0.1$ (h) State 2, $\lambda = 1.0$ (i) State 2, $\lambda = 2.0$ (j) State 2, $\lambda = 4.0$

Figure 1: Unnormalized density plots from a Morse neural network with RBF kernel (Equation 1) for $\lambda = \{0.1, 1.0, 2.0, 4.0\}$. Eight action modes are uniformly spaced points on a unit circle with alternating modes assigned to each state. 64 action samples for each state are drawn with a standard deviation 0.01 and clipped to lie in the range $[-1.0, 1.0]$. Subfig (a) displays the training data and actions sampled for each state. Color-coded training points are overlaid on density plots for convenience. The Morse network (2 hidden layers of size 64, ReLU nonlinearity, gradient penalty, and layer normalization) captures all modes in each state. Note that the density plots cover a larger action range of $[-1.8, 1.8]$.

We provide the proof in the Supplementary Material [38]. This shows that the behavioral constraint in Equation 10 is sufficient to ensure that the policy is restricted to following supported trajectories.

Furthermore, using Proposition 1, theorems on performance from Florence et al. [8] apply. Namely, Theorem 2 suggests that any behavior policy with an arbitrary Lipschitz constant can approximated by an implicit model with a bounded Lipschitz constant with low model error.

4.2 *Challenges when Constraining Offline MBRL*

Naïvely applying our policy constraint to states from model rollouts will result in poor performance. The Morse neural network as an implicit behavior policy poorly extrapolates to unseen states; prior work by Xu et al. [48] and Florence et al. [8] shows that implicit models formed by fully connected neural networks can only perform linear function extrapolation, yet assuming that the behavior policy is linear can be overly restrictive.

This is a critical shortcoming of EBMs when used for the policy constraint and anti-exploration bonus in unknown states, which can lead to incorrect modes being learned/penalized. This drawback extends to prior offline MBRL methods using a dynamics constraint [51, 20] who subsequently restrict themselves to training on small perturbations of dataset states rather than using dynamics rollouts. To use dynamics models for data augmentations, we must ensure that rollouts remain within the data support, both so that the Morse-based constraint is valid and to prevent the exploitation of learned dynamics. The ability to combine policy constraint with rollout OOD detection is (to the best of our knowledge) unique to MoMo and in the next section, we describe how the uncertainty detection properties of the Morse neural network enable detection of when states in synthetic trajectories have moved too far from the dataset support.

4.3 *Extending to Offline MBRL*

MBPO-based offline MBRL algorithms [18] typically penalize OOD regions by directly using ensemble uncertainty estimates [22, 49] or assuming that all states produced by the dynamics models are OOD [50, 28, 36, 5, 29]. MOReL [22] terminates the rollouts when the estimated uncertainty increases beyond a threshold. We perform a similar rollout truncation when the policy is excessively OOD using the Morse neural network.

Our rollout truncation refers to stopping the rollout without performing any subsequent bootstrapping or episode termination (i.e., we do not update truncated action-values during policy evaluation). This differs slightly from the treatment in online RL as in the offline domain, truncation is not an environment property, but a property that we impose on the empirical MDP.

Let $\beta(a|s)$ denote the behavior policy, s_t denote the state at time t, a_t denote the action selected by an agent, P_0 denote the initial state distribution and $\mathcal{T}(s_{t+1} \mid s_t, a_t)$ denote the state transition. We assume that the environment transition is deterministic and aim to train an agent to minimize the (cross) entropy under the behavior policy over a trajectory $\tau_{0:T}$:

$$\mathcal{H}(\tau_{0:T}) = -\sum_{t=0}^{T} \mathbb{E}_{a_t \sim \pi(s_t)} \left[\log \beta(a_t|s_t)\right] \mid {}^{s_0 \sim P_0}_{s_{t+1} \sim \mathcal{T}(\cdot|s_t, a_t)} \tag{16}$$

$$\approx -\sum_{t=0}^{T} \mathbb{E}_{a_t \sim \pi(s_t)} \left[\log M_\psi(s_t, a_t) - \log Z_\psi(s_t)\right] \tag{17}$$

$$= -\sum_{t=0}^{T} \mathbb{E}_{a_t \sim \pi(s_t)} \left[\log M_\psi(s_t, a_t)\right] + \text{const} \tag{18}$$

$$\geq -\sum_{t=0}^{T} \mathbb{E}_{a_t \sim \pi(s_t)} \left[\log M_\psi(s_t, a_t)\right]. \tag{19}$$

Using an accurate learned dynamics model $\hat{\mathcal{T}}$, we can estimate a lower bound of trajectory entropy using M_ψ. By performing additional manipulation, we can instead extract a *certainty* estimate over a trajectory:

$$\exp(-\mathcal{H}(\tau_{0:T})) \leq \exp\left(\sum_{t=0}^{T} \mathbb{E}_\pi \left[\log M_\psi(s_t, a_t)\right]\right) \tag{20}$$

$$= \prod_{t=0}^{T} \mathbb{E}_\pi \left[M_\psi(s_t, a_t)\right] \tag{21}$$

$$:= P_{M_\psi}(\tau_{0:T}) \in [0, 1] \tag{22}$$

Algorithm 2 Truncated Synthetic Rollouts

Input: Initial state $s_0 \sim P_0$, Morse neural network M_ψ and MLE trained dynamics $\hat{T}$
Initalize: $P_{M_\psi}(\tau_{0:0}) = 1$, $\mathcal{D}_{\text{model}} = \{\}$
for t = 1 **to** T **do**
  $a_t \sim \pi(a_t \mid s_t)$ ▷ Get action
  Get $M_\psi(s_t, a_t)$ and compute $P_{M_\psi}(\tau_{0:t})$ ▷ Trajectory prob.
  $\texttt{trunc}(P_{M_\psi}(\tau_{0:t}))$ ▷ Truncate rollout
  $s_{t+1}, r_t \sim \hat{\mathcal{T}}(\cdot \mid s_t, a_t)$ ▷ Step dynamics
  $\mathcal{D}_{\text{model}} \leftarrow \mathcal{D}_{\text{model}} \cup \{s_t, a_t, r_t, s_{t+1}\}$ ▷ Append to buffer
end for
return $\mathcal{D}_{\text{model}}$

where the final line is the Morse-probability that the trajectory is behavior-consistent. This is an upper-bound estimate of the negative-exponentiated entropy. Therefore, a low entropy trajectory has a high trajectory-probability $P_{M_\psi}(\tau_{0:T})$. With the ability to estimate the probability of a rollout, we design a function to truncate a rollout when its probability falls below a threshold.

Rollout Truncation We design a detection mechanism to identify when a trajectory is excessively OOD and truncate the synthetic rollout.

Definition 3 (OOD Truncation Function)**.** *Given a rollout, at each timestep compute the Morse-probability $M_\psi(s_t, a_t)$ and the trajectory-probability $P_{M_\psi}(\tau_{0:t}) = P_{M_\psi}(\tau_{0:t-1}) \times M_\psi(s_t, a_t)$ and truncate the episode if the probability is below a threshold:*

$$\texttt{trunc}(P_{M_\psi}(\tau_{0:t})) = \begin{cases} \textit{True} & \textit{if } P_{M_\psi}(\tau_{0:t}) < \epsilon_{\textit{trunc}}, \\ \textit{False} & \textit{otherwise.} \end{cases} \tag{23}$$

Here, $\epsilon_{\textit{trunc}} \in [0, 1]$ is a hyperparameter to be chosen that determines how much a trajectory can deviate from a behaviorally consistent one before the rollout is truncated.

Selecting $\epsilon_{\text{trunc}} = 1$ will demand perfect reproduction of behavior policy trajectories and (in the absence of dynamics model errors) will be equivalent to model-free RL using only the offline dataset. Selecting a lower threshold allows the policy to explore the dynamics-estimated environment with increasingly low probability trajectories permitted as $\epsilon_{\text{trunc}} \to 0$. Algorithm 2 describes how we employ the truncation function when generating rollouts.

4.4 Practical Implementation

We discuss aspects of MoMo's implementation here and provide additional details in the Supplementary Material [38].

Morse Net Training an implicit model with bounded Lipschitz constant also bounds the model error [8]. When using neural network function approximators, this can be realized in several ways. We use a maximum-margin gradient penalty [21] with Lipschitz constant set to 1 and add the gradient penalty loss as an auxiliary objective to Equation 4 with equal weighting.

Deep Architectures Rank collapse limits the expressivity of deep neural networks in offline RL [16, 24]. Morse neural networks benefit from deep architectures [7, 8] which can suffer from implicit underparametrization. This is a concern in MoMo as we directly use the Morse network as a loss function. Sinha et al. [37] present D2RL, a deep fully-connected architecture that uses skip connections which alleviate rank collapse at minimal extra computational cost. We adopt this architecture for the Morse neural network. In addition, we use layer normalization [3] for the Morse and critic networks which improves stability.

Actor–Critic MoMo can be used with any model-free RL algorithm. We use TD3 [11] as the base actor–critic algorithm and retain standard TD3 hyperparameters. We use the policy objective specified in Equation 10 and also normalize the Q values for easier optimization.

Dynamics Model A hallmark of most previous offline MBRL is the training and use of an ensemble (typically 4-7) of dynamics models to both prevent exploitation of individual models and for improved uncertainty estimation. The Morse network is an uncertainty estimator that we use directly to identify when a rollout goes OOD and for the anti-exploration bonus. This negates the need for ensemble-based uncertainty. Consequently, we train and use a **single** Gaussian dynamics model for all tasks.

Hyperparameters Model-based MoMo is an offline MBRL algorithm that largely inherits the hyperparameters of MBPO. Furthermore, model-based MoMo adds two new parameters: 1) kernel scale λ and 2) truncation threshold ϵ_{trunc}. Prior critic-regularizing offline MBRL methods demand per-dataset tuning of hyperparameters to achieve reported performance. Such fine-grained tuning requirements can hamper the adoption of algorithms in real-world applications. In our experiments, we use a single set of hyperparameters in each domain to evaluate MoMo (i.e. one constant set of hyperparameters for **all** Locomotion tasks and a separate constant set for **all** Adroit tasks). Our ablation experiments demonstrate robust performance over a range of sensible hyperparameters. For primary results, we use $\lambda = 1.0$ and $\epsilon_{\text{trunc}} = 0.95$ for Locomotion tasks and $\lambda = 2.0$, $\epsilon_{\text{trunc}} = 0.98$ for Adroit.

5 Experiments

We evaluate both model-based and model-free (i.e. generating no synthetic rollouts using learned dynamics) versions of MoMO and compare them with a range of baselines. We include the model-free baselines: CQL [25], IQL [23], TD3-BC [10] and TD3-CVAE (model-free with anti-exploration) [34], the last of which is the progenitor of anti-exploration methods.

We also include the offline MBRL baselines: MoREL [22], MOPO [49], RAMBO [36], COMBO [50], ARMOR [5], MOBILE [40] and OSR [20]. The final three are recent methods that achieve state-of-the-art performance on benchmarks and are representative of the major techniques in modern offline MBRL.

ARMOR adopts an adversarial approach and performs value penalization on actions produced by the policy π and increases values for actions produced by a reference policy π_{ref}, which must either be provided or be learned via behavioral cloning. MOBILE estimates uncertainty using estimated Q-values over the next states predicted by the dynamics models and penalizes the bootstrapped update with a penalty derived from the uncertainty. The authors of OSR propose two different methods of regularization: one that is policy constraining and the other a CQL-like critic regularization. OSR's policy is directed to recover towards in-dataset states when starting from unseen ones. We select results for the former version of OSR as a direct comparison to our policy constraint-based MoMo.

In our experiments, we aim to answer the following questions:

- How does MoMo perform in both model-based and model-free regimes?
- How important is the threshold ϵ_{trunc} for model-based performance?
- How sensitive is performance to the kernel scale λ?

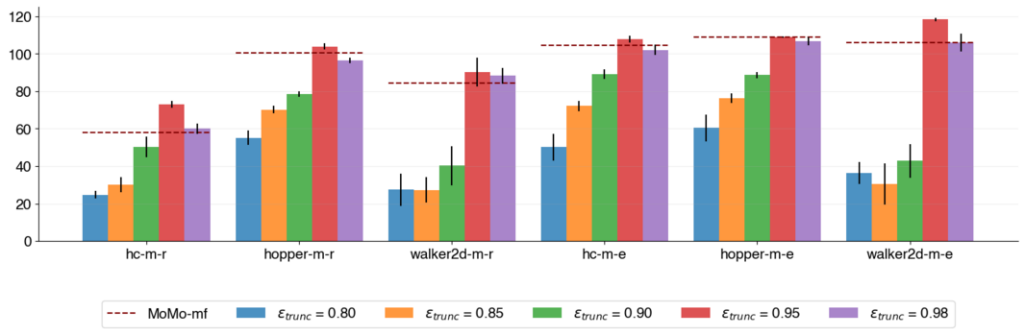


Figure 2: Ablations of ϵ_{trunc} on `-m-r` and `-m-e` datasets. The standard deviation over seeds is also included for each ablation. Note that $\epsilon_{\text{trunc}} = 0.95$ is the configuration used for MoMo-mb for primary results and MoMo-mf scores are included here for reference.

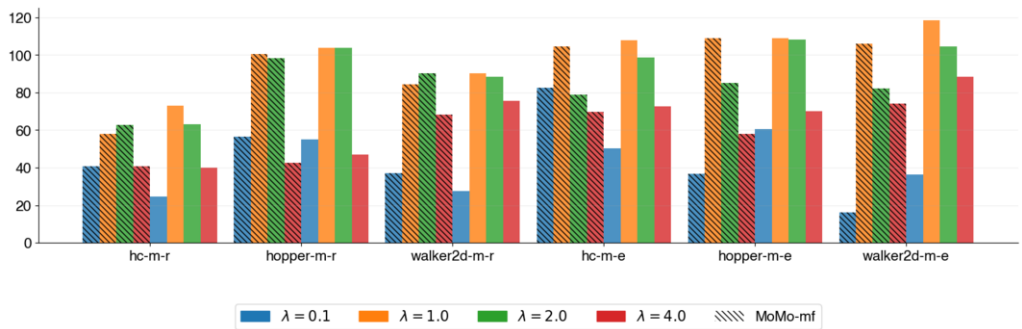


Figure 3: Ablations of λ on `-m-r` and `-m-e` datasets. We use $\lambda = 1.0$ for primary results and perform additional sweeps over $\lambda = [0.1, 2.0, 4.0]$. The standard deviation over seeds is omitted for the sake of clarity. For MoMo-mb we use clear bars and maintain $\epsilon_{\text{trunc}} = 0.95$ in experiments.

- Is anti-exploration necessary for model-based MoMo?

We answer the first question in the context of performance on D4RL Locomotion and Adroit datasets [9] and compare scores with the aforementioned baselines. The remainder of the questions are answered via ablation experiments that analyze the impact of hyperparameters and design choices. All experiments were conducted over five seeds with evaluation over ten episodes.

5.1 D4RL Performance

We compare model-free and model-based MoMo with the aforementioned baselines for Gym Locomotion tasks in Table 1 and Adroit tasks in Table 2.

Gym Locomotion Gym Locomotion results show that model-based MoMo achieves the highest scores (either best or within 1 standard deviation of best) in 9 of the 12 datasets. Furthermore, MoMo-mb recovers expert performance (≥ 95.0) in 6 of 12 datasets with 100% expert performance recovery on the `-m-e` datasets which contain a mixture of optimal and suboptimal trajectories. The `-r` datasets pose a significant challenge to all methods they contain highly suboptimal examples generated by a random uniform policy and all methods tend to perform poorly on these tasks. Comparing MoMo-mf to MoMo-mb, changing to the model-based domain improves performance by an average of 21.5%, with marked improvements in the `-r`, `-m` and `-m-r` datasets. Finally, MoMo-mf outperforms both policy-constrained TD3-CVAE and dynamics-constrained OSR.

Adroit The Adroit datasets pose a very different set of challenges to those in Locomotion tasks: Adroit datasets are marked by having few, human-generated examples (`-h`) or a mix between human and BC-trained behavior policies (`-c`). Consequently, learning algorithms must be able to deal with highly reward-disparate trajectories effectively and must generalize well. MoMo-mb achieves the best performance among all methods in 8 of 12 Adroit datasets and MoMo-mf consistently outperforms model-free baselines.

5.2 Ablating Threshold ϵ_{trunc}

The `-m-r` and `-m-e` Gym Locomotion tasks contain trajectories from two or more policies, which makes them candidate datasets for ablating MoMo hyperparameters. We evaluate performance when changing the truncation threshold parameter for values $\epsilon_{\text{trunc}} = \{0.80, 0.85, 0.90, 0.95, 0.98\}$ in Figure 2.

Across the board, reducing ϵ_{trunc} degrades performance. This is plausible, as reducing ϵ_{trunc} allows a trajectory to deviate more from one following the behavior policy before it is truncated. Overestimated action values inevitably propagate, resulting in an uncorrected distribution shift that results in low performance. Increasing ϵ_{trunc} to 0.98 has a relatively small effect on performance.

A higher ϵ_{trunc} increases the aggressiveness of truncation to ensure that trajectories are *closer* to those following the behavior policy. In effect, this tries to ensure that the trajectories generated vary less from those in the dataset and consequently, we expect performance to be similar to model-free MoMo. In practice, dynamics model error may cause MoMo-mb to underperform compared to MoMo-mf.

5.3 Sensitivity to λ

We ablate the kernel scale parameter λ on the `-m-r` and `-m-e` datasets for both MoMo-mf and MoMo-mb in Figure 3. Increasing λ *sharpens* the density around the dataset modes (see Figure 1) and is equivalent to increasing the relative weighting of the policy constraint (recall that the log RBF kernel is $-\lambda \mid\mid z - t \mid\mid^2$).

Figure 3 shows that using $\lambda = 0.1$ leads to a substantial performance drop in both model-free and model-based conditions as the

Table 1: Normalized scores on D4RL Gym Locomotion datasets. Scores are taken from the reported score in the original work with top scores in **bold** and second-best underlined. Model-free and model-based versions of MoMo are indicated by -mf and -mb, respectively, and scores include one standard deviation across seeds.

	Model-free				Model-based						Inv-dyn	**Ours**	
Dataset	TD3-BC	CQL	IQL	TD3-CVAE	MoREL	MOPO	RAMBO	COMBO	ARMOR	MOBILE	OSR	MoMo-mf	MoMo-mb
hc-r	10.2	35.4	-	28.6	25.6	35.4	**40.0**	38.8	-	39.3	35.2	30.2 ± 2.3	39.6 ± 3.7
hopper-r	11.0	10.7	-	11.7	**53.6**	11.7	21.6	17.9	-	31.9	10.3	9.4 ± 5.2	18.3 ± 2.8
walker2d-r	1.4	2.7	-	5.5	**37.3**	13.6	11.5	7.0	-	17.9	13.5	20.7 ± 1.4	26.8 ± 3.3
hc-m	42.8	37.2	47.4	43.2	42.1	42.3	**77.6**	54.2	54.2	74.6	48.8	62.1 ± 1.4	77.1 ± 0.9
hopper-m	99.5	44.2	66.2	55.9	95.4	28.0	92.8	97.2	101.2	106.6	95.2	95.7 ± 2.4	**110.8 ± 2.3**
walker2d-m	79.7	57.5	78.3	68.2	77.8	17.8	86.9	81.9	90.7	87.7	85.1	84.6 ± 2.8	**95.0 ± 1.4**
hc-m-r	43.3	41.9	44.2	45.3	40.2	53.1	68.9	55.1	50.5	71.7	46.8	58.1 ± 0.6	**72.9 ± 1.8**
hopper-m-r	31.4	28.6	94.7	46.7	93.6	67.5	96.6	89.5	97.1	103.9	96.7	**106.1 ± 1.0**	104.0 ± 1.8
walker2d-m-r	25.2	15.8	73.8	15.4	49.8	39.0	85.0	56.0	85.6	89.9	87.9	84.4 ± 2.9	**90.4 ± 7.7**
hc-m-e	97.9	27.1	86.7	86.1	53.3	63.3	93.7	90.0	93.5	108.2	94.7	**110.7 ± 1.5**	107.9 ± 1.9
hopper-m-e	112.2	111.4	91.5	111.6	108.7	111.9	83.3	111.1	112.6	103.4	**113.2**	108.8 ± 0.6	109.1 ± 0.4
walker2d-m-e	101.1	68.1	109.6	84.9	95.6	44.6	68.3	103.3	115.2	112.2	112.9	106.0 ± 8.3	**118.4 ± 0.9**

Table 2: Normalized scores on D4RL Adroit datasets. Baseline scores are taken from the original work with top scores in **bold** and second-best underlined. Prior methods that do not evaluate on Adroit are excluded. Model-free and model-based versions of MoMo are indicated by -mf and -mb, respectively. We include the behavioral cloning baseline as it tends to perform well in Adroit tasks.

Dataset	BC	CQL	IQL	TD3-CVAE	ARMOR	MOBILE	MoMo-mf	MoMo-mb
pen-h	34.4	37.5	71.5	59.2	72.8	30.1	67.2	**74.9**
hammer-h	1.5	**4.4**	1.4	0.2	1.9	0.4	0.5	1.7
door-h	0.5	9.9	4.3	0.0	6.3	-0.2	6.8	**11.3**
relocate-h	0.0	0.2	0.1	0.0	**0.4**	-	0.2	**0.4**
pen-c	56.9	39.2	37.5	45.4	51.4	69.0	**77.2**	74.1
hammer-c	0.8	**2.1**	**2.1**	0.3	0.7	1.5	0.5	0.7
door-c	-0.1	0.4	1.6	0.0	-0.1	**24.0**	1.1	5.8
relocate-c	-0.1	-0.1	-0.2	-0.2	-0.1	-	-0.2	**0.0**
pen-e	85.1	107.0	-	112.3	112.2	-	97.4	**132.6**
hammer-e	125.6	86.7	-	128.9	118.8	-	126.9	**131.8**
door-e	34.9	101.5	-	59.4	98.7	-	76.7	**103.4**
relocate-e	101.3	95.0	-	**106.4**	96.0	-	95.0	104.7

constraint is too lax. Increasing $\lambda = 2.0$ has mixed results, and further increasing to $\lambda = 4.0$ likely overconstrains the policy. Interestingly, the change from $\lambda = 1.0 \rightarrow 2.0$ has a larger impact on MoMo-mf, on the m-e datasets where this change results in a -22.9% performance change compared to a -7.1% change for MoMo-mb.

5.4 *Importance of Anti-Exploration*

The primary mechanism by which MoMo encourages the policy to stay in-support is via a policy constraint. Previous work suggests that removing the policy constraint results in poor performance [44]. The anti-exploration bonus (see Equation 11) augments the bootstrapped return estimate, which in MoMo upper bounds the logarithm of the behavior policy density. This performs a regularized update that bounds KL regularized value iteration [13, 46] and curbs value overestimation caused by off-policy evaluation. We evaluate the impact of the anti-exploration bonus on MoMo-mb in Table 3. Removing the bonus causes performance to drop in almost all datasets, with an average performance decrease of 12.5%, suggesting MoMo's distance-aware conservatism is critical to performance. The mixed impact on -random datasets may be due to the challenges of training the Morse network on actions sampled from a random policy.

6 Discussion

Computational Cost Model-based MoMo trains a dynamics model as well as the Morse neural network. This translates into an increase in total training time (pretraining dynamics and Morse network for 100 000 steps followed by policy/value learning for 1 million steps) of approximately 65% over model-free TD3-BC and 25% over MBPO.

Uncertainty Estimation Morse neural networks are a specific method of uncertainty estimation that also enjoy equivalency to EBMs, resulting in specific properties used in this work. Alternative methods of uncertainty estimation can be used, the Morse network itself being identical to other estimators depending on instantiation (see Dherin et al. [7] for a more detailed discussion).

Limitations The primary limitation of MoMo is the inability of the Morse network is extrapolate beyond the offline dataset state support, thus requiring the truncation function when using model-based MoMo, which introduces the ϵ_{trunc} hyperparameter. An alternative is to update the Morse network periodically using synthetically generated trajectories, though this will add the optimization challenges of adversarial training dynamics and increase training cost.

Table 3: Normalized scores on D4RL datasets when removing the anti-exploration bonus. We also include the percentage change in performance.

Dataset	MoMo-mb	-anti-exploration bonus
hc-r	39.6 ± 3.7	42.7 ± 3.2 (+7.8%)
hopper-r	18.3 ± 2.8	20.9 ± 4.5 (+14.2%)
walker2d-r	26.8 ± 3.3	24.5 ± 1.7 (-8.6%)
hc-m	77.1 ± 0.9	70.5 ± 1.1 (-8.6%)
hopper-m	110.8 ± 2.3	92.4 ± 2.6 (-16.6%)
walker2d-m	95.0 ± 1.4	74.9 ± 3.9 (-21.6%)
hc-m-r	72.9 ± 1.8	66.1 ± 1.4 (-9.3%)
hopper-m-r	104.0 ± 1.8	68.1 ± 1.9 (-34.5%)
walker2d-m-r	90.4 ± 7.7	87.2 ± 5.8 (-3.5%)
hc-m-e	107.9 ± 1.9	100.7 ± 3.1 (-6.7%)
hopper-m-e	109.1 ± 0.4	102.6 ± 2.5 (-6.3%)
walker2d-m-e	118.4 ± 0.9	98.2 ± 0.7 (-17.1%)
Total	970.3	848.8 (-12.5%)

7 Conclusion

In this paper, we studied how a model-free offline RL paradigm could be extended to the model-based domain. We incorporated Morse neural networks into the anti-exploration framework to yield MoMo, which constrains both the policy and penalizes off-policy evaluation with an anti-exploration bonus. To overcome the limitations of model-free MoMo, we deploy Morse networks as both an EBM and an uncertainty estimator to truncate OOD trajectories. Experiments comparing both model-free and model-based versions of MoMo with recent baselines demonstrate MoMo's superior results as well as the benefits of incorporating dynamics models.

Ensemble-based uncertainty estimates can vary greatly in scale which necessitates per-dataset tuning to achieve reported performance. In contrast, MoMo uses identical hyperparameter values across tasks in the same domain, and subsequent ablation experiments demonstrate that performance is stable across a range of hyperparameters. Future work should analyze the approaches employed by modern model-free offline RL and explore how they may be adapted for offline MBRL.

References

[1] S.-i. Amari. *Information geometry and its applications*, volume 194. Springer, 2016. ISBN 4431559787.

[2] G. An, S. Moon, J.-H. Kim, and H. O. Song. Uncertainty-based offline reinforcement learning with diversified Q-ensemble. *Advances in neural information processing systems*, 34:7436–7447, 2021.

[3] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[4] S. Basu and S. Prasad. A connection between cut locus, Thom space and Morse-Bott functions. *arXiv preprint arXiv:2011.02972*, 2020.

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

[6] K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

[7] B. Dherin, H. Hu, J. Ren, M. W. Dusenberry, and B. Lakshminarayanan. Morse neural networks for uncertainty quantification. *arXiv preprint arXiv:2307.00667*, 2023.

[8] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson. Implicit behavioral cloning. In *Conference on Robot Learning*, pages 158–168. PMLR, 2022. ISBN 2640-3498.

[9] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine. D4RL: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[10] S. Fujimoto and S. S. Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34: 20132–20145, 2021.

[11] S. Fujimoto, H. Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018. ISBN 2640-3498.

[12] S. Fujimoto, D. Meger, and D. Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pages 2052–2062. PMLR, 2019. ISBN 2640-3498.

[13] M. Geist, B. Scherrer, and O. Pietquin. A theory of regularized markov decision processes. In *International Conference on Machine Learning*, pages 2160–2169. PMLR, 2019. ISBN 2640-3498.

[14] K. Ghasemipour, S. S. Gu, and O. Nachum. Why so pessimistic? estimating uncertainties for offline RL through ensembles, and why their independence matters. *Advances in Neural Information Processing Systems*, 35:18267–18281, 2022.

[15] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016. ISBN 0262337371.

[16] C. Gulcehre, S. Srinivasan, J. Sygnowski, G. Ostrovski, M. Farajtabar, M. Hoffman, R. Pascanu, and A. Doucet. An empirical study of implicit regularization in deep offline RL. *arXiv preprint arXiv:2207.02099*, 2022.

[17] H. He. A survey on offline model-based reinforcement learning. *arXiv preprint arXiv:2305.03360*, 2023.

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

[19] J. Jeong, X. Wang, M. Gimelfarb, H. Kim, B. Abdulhai, and S. Sanner. Conservative bayesian model-based value expansion for offline policy optimization. *arXiv preprint arXiv:2210.03802*, 2022.

[20] K. Jiang, J.-y. Yao, and X. Tan. Recovering from out-of-sample states via inverse dynamics in offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[21] A. Jolicoeur-Martineau and I. Mitliagkas. Gradient penalty from a maximum margin perspective. *arXiv preprint arXiv:1910.06922*, 2019.

[22] R. Kidambi, A. Rajeswaran, P. Netrapalli, and T. Joachims. MoREL: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33:21810–21823, 2020.

[23] I. Kostrikov, A. Nair, and S. Levine. Offline reinforcement learning with implicit Q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

[24] A. Kumar, R. Agarwal, D. Ghosh, and S. Levine. Implicit under-parameterization inhibits data-efficient deep reinforcement learning. *arXiv preprint arXiv:2010.14498*, 2020.

[25] A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative Q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

[26] T. Kurutach, I. Clavera, Y. Duan, A. Tamar, and P. Abbeel. Model-ensemble trust-region policy optimization. *arXiv preprint arXiv:1802.10592*, 2018.

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

[28] X.-Y. Liu, X.-H. Zhou, X.-L. Xie, S.-Q. Liu, Z.-Q. Feng, H. Li, M.-J. Gui, T.-Y. Xiang, D.-X. Huang, and Z.-G. Hou. Domain: Mildly conservative model-based offline reinforcement learning. *arXiv preprint arXiv:2309.08925*, 2023.

[29] F.-M. Luo, T. Xu, X. Cao, and Y. Yu. Reward-consistent dynamics models are strongly generalizable for offline reinforcement learning. *arXiv preprint arXiv:2310.05422*, 2023.

[30] T. M. Moerland, J. Broekens, A. Plaat, and C. M. Jonker. Model-based reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 16(1):1–118, 2023. ISSN 1935-8237.

[31] A. Nikulin, V. Kurenkov, D. Tarasov, and S. Kolesnikov. Anti-exploration by random network distillation. *arXiv preprint arXiv:2301.13616*, 2023.

[32] R. F. Prudencio, M. R. Maximo, and E. L. Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *arXiv preprint arXiv:2203.01387*, 2022.

[33] A. Rajeswaran, S. Ghotra, B. Ravindran, and S. Levine. Epopt: Learning robust neural network policies using model ensembles. *arXiv preprint arXiv:1610.01283*, 2016.

[34] S. Rezaeifar, R. Dadashi, N. Vieillard, L. Hussenot, O. Bachem, O. Pietquin, and M. Geist. Offline reinforcement learning as anti-exploration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8106–8114, 2022. ISBN 2374-3468.

[35] S. Rezaeifar, R. Dadashi, N. Vieillard, L. Hussenot, O. Bachem, O. Pietquin, and M. Geist. Offline reinforcement learning as anti-exploration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8106–8114, 2022. ISBN 2374-3468.

[36] M. Rigter, B. Lacerda, and N. Hawes. RAMBO-RL: Robust adversarial model-based offline reinforcement learning. *Advances in neural information processing systems*, 35:16082–16097, 2022.

[37] S. Sinha, H. Bharadhwaj, A. Srinivas, and A. Garg. D2RL: Deep dense architectures in reinforcement learning. *arXiv preprint arXiv:2010.09163*, 2020.

[38] P. Srinivasan and W. Knottenbelt. Offline model-based reinforcement learning with anti-exploration. *arXiv preprint arXiv:2408.10713*, 2024.

[39] P. Srinivasan and W. Knottenbelt. Offline reinforcement learning with behavioral supervisor tuning. In K. Larson, editor, *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pages 4929–4937. International Joint Conferences on Artificial Intelligence Organization, 8 2024. doi: 10.24963/ijcai.2024/545. URL https://doi.org/10.24963/ijcai.2024/545. Main Track.

[40] Y. Sun, J. Zhang, C. Jia, H. Lin, J. Ye, and Y. Yu. Model-bellman inconsistency for model-based offline reinforcement learning. 2023.

[41] R. S. Sutton. *Integrated architectures for learning, planning, and reacting based on approximating dynamic programming*, pages 216–224. Elsevier, 1990.

[42] R. S. Sutton. *Planning by incremental dynamic programming*, pages 353–357. Elsevier, 1991.

[43] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018. ISBN 0262352702.

[44] D. Tarasov, V. Kurenkov, A. Nikulin, and S. Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. *arXiv preprint arXiv:2305.09836*, 2023.

[45] H. P. Van Hasselt, M. Hessel, and J. Aslanides. When to use parametric models in reinforcement learning? *Advances in Neural Information Processing Systems*, 32, 2019.

[46] N. Vieillard, T. Kozuno, B. Scherrer, O. Pietquin, R. Munos, and M. Geist. Leverage the average: an analysis of KL regularization in reinforcement learning. *Advances in Neural Information Processing Systems*, 33:12163–12174, 2020.

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

[48] K. Xu, M. Zhang, J. Li, S. S. Du, K.-i. Kawarabayashi, and S. Jegelka. How neural networks extrapolate: From feedforward to graph neural networks. *arXiv preprint arXiv:2009.11848*, 2020.

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

[50] T. Yu, A. Kumar, R. Rafailov, A. Rajeswaran, S. Levine, and C. Finn. COMBO: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967, 2021.

[51] H. Zhang, J. Shao, Y. Jiang, S. He, G. Zhang, and X. Ji. State deviation correction for offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9022–9030, 2022. ISBN 2374-3468.