Offline Reinforcement Learning with Behavioral Supervisor Tuning

Padmanaba Srinivasan , William Knottenbelt
Imperial College London
{ps3416, wjk}@imperial.ac.uk

Abstract

Offline reinforcement learning (RL) algorithms are applied to learn performant, well-generalizing policies when provided with a static dataset of interactions. Many recent approaches to offline RL have seen substantial success, but with one key caveat: they demand substantial per-dataset hyperparameter tuning to achieve reported perf ormance, which requires policy rollouts in the environment to evaluate; this can rapidly become cumbersome. Furthermore, substantial tuning requirements can hamper the adoption of these algorithms in practical domains. In this paper, we present TD3 with **B**ehavioral **S**upervisor **T**uning (TD3-BST), an algorithm that trains an uncertainty model and uses it to guide the policy to select actions within the dataset support. TD3-BST can learn more effective policies from offline datasets compared to previous methods and achieves the best performance across challenging benchmarks without requiring per-dataset tuning.

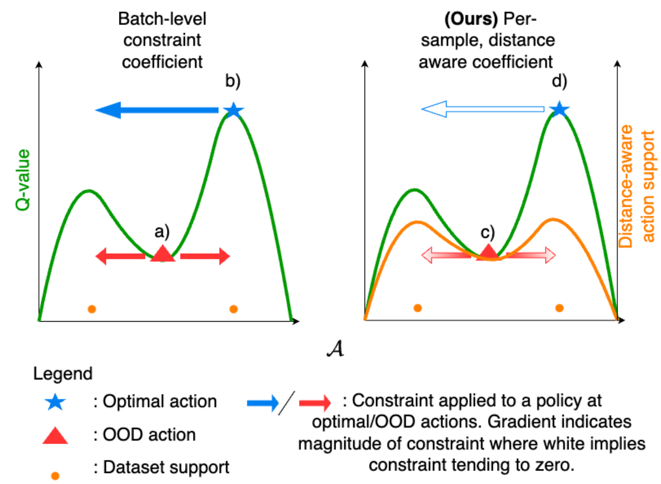


Figure 1: An illustration of our method versus typical, TD3-BC-like actor-constraint methods. **TD3-BC:** a) A policy selecting an OOD action is constrained to select in-dataset actions. b) A policy selecting the optimal action may be penalized for not selecting an in-dataset, but not in-batch, inferior action. **Our method:** c) A policy selecting OOD actions is drawn towards in-dataset actions with decreasing constraint coefficient as it moves closer to *any* supported action. d) An optimal policy is not penalized for selecting an in-dataset action when the action is not contained in the current batch.

1 Introduction

Reinforcement learning (RL) is a method of learning where an agent interacts with an environment to collect experiences and seeks to maximize the reward provided by the environment. This typically follows a repeating cycle of experience collecting and improvement [Sutton and Barto, 2018]. This is termed *online* RL due to the need for policy rollouts in the environment. Both on-policy and off-policy RL require some schedule of online interaction which, in some domains, can be infeasible due to experimental or environmental limitations [Mirowski *et al.*, 2018; Yu *et al.*, 2021]. With such constraints, a dataset may instead be collected that consists of demonstrations by arbitrary (potentially multiple, unknown) behavior policies [Lange *et al.*, 2012] that may be suboptimal. Offline reinforcement learning algorithms are designed to recover optimal policies from such static datasets.

The primary challenge in offline RL is the evaluation of out-of-distribution (OOD) actions; offline datasets rarely offer support over the entire state-action space and neural networks overestimate values when extrapolating to OOD actions [Fujimoto *et al.*, 2018; Gulcehre *et al.*, 2020; Kumar *et al.*, 2020; Kumar *et al.*, 2019]. If trained using standard off-policy methods, a policy will select any actions that maximize reward, which includes OOD actions. The difference between the rewards implied by the value function and the environment results in a distribution shift that can result in failure in real-world policy rollouts. Thus, offline RL algorithms must both maximize the reward and follow the behavioral policy, while having to potentially "stitch" together several suboptimal trajectories. The former requirement is usually satisfied by introducing a constraint on the actor to either penalize deviation from the behavior policy or epistemic uncertainty of the value function, or by regularizing the value function to directly minimize OOD action-values.

Many recent approaches to offline RL [Tarasov *et al.*, 2023; Zhu *et al.*, 2022; Li *et al.*, 2022; Nikulin *et al.*, 2023; Xu *et al.*, 2023] demonstrate success in D4RL benchmarks [Fu *et al.*, 2020], but demand the onerous task of per-dataset hyperparameter tuning [Zhang and Jiang, 2021]. Algorithms that require substantial offline fine-tuning can be infeasible in real-world applications [Tang and Wiens, 2021], hampering their adoption in favor of simpler, older algorithms [Emerson *et al.*, 2023; Zhu *et al.*, 2023]. These older methods [Fujimoto and Gu, 2021; Kumar *et al.*, 2020; Kostrikov *et al.*, 2021b] provide excellent "bang-for-buck" as their hyperparameters work well across a range of D4RL datasets.

Contributions In this paper, we show how a trained uncertainty model can be incorporated into the regularized policy objective as a *behavioral supervisor* to yield TD3 with behavioral supervisor tuning (TD3-BST). The key advantage of our method is the dynamic regularization weighting performed by the uncertainty network, which allows the learned policy to maximize Q-values around dataset modes. Evaluation on D4RL datasets demonstrates that TD3-BST achieves SOTA performance, and ablation experiments analyze the performance of the uncertainty model and the sensitivity of the parameters of the BST objective.

2 Related Work

Reinforcement learning is a framework for sequential decision making often formulated as a Markov decision process (MDP), $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, R, p, p_0, \gamma\}$ with state space $\mathcal{S}$, action space $\mathcal{A}$, a scalar reward dependent on state and action $R(s, a)$, transition dynamics p, initial state distribution p_0 and discount factor $\gamma \in [0, 1)$ [Sutton and Barto, 2018]. RL aims to learn a policy $\pi \in \Pi$ that executes action $a = \pi(s)$ that will maximize the expected discounted reward $J(\pi) = \mathbb{E}_{\tau \sim P_\pi(\tau)}\left[\sum_{t=0}^{T} \gamma_t R(s_t, a_t)\right]$ where $P_\pi(\tau) = p_0(s_0)\prod_{t=0}^{T} \pi(a_t \mid s_t)p(s_{t+1} \mid s_t, a_t)$ is the trajectory under π. Rather than rolling out an entire trajectory, a state-action value function (Q function) is often used: $Q_\pi(s, a) = \mathbb{E}_{\tau \sim P_\pi(\tau)}\left[\sum_{t=0}^{T} \gamma_t r(s_t, a_t) \mid s_0 = s, a_0 = a\right]$.

2.1 Offline Reinforcement Learning

Offline RL algorithms are presented with a static dataset $\mathcal{D}$ that consists of tuples $\{s, a, r, s'\}$ where $r \sim R(s, a)$ and $s' \sim p(\cdot \mid s, a)$.

$\mathcal{D}$ has limited coverage over $\mathcal{S} \times \mathcal{A}$; hence, offline RL algorithms must constrain the policy to select actions within the dataset support. To this end, algorithms employ one of three approaches: 1) policy constraints; 2) critic regularization; or 3) uncertainty penalization.

Policy constraint Policy constraints modify the actor's objective only to minimize divergence from the behavior policy. Most simply, this adds a constraint term [Fujimoto and Gu, 2021; Tarasov *et al.*, 2023] to the policy objective:

$$\arg\max_{\pi} \mathbb{E}_{\{s,a\} \sim \mathcal{D}}\left[Q(s, \pi(s)) - \alpha D(\pi, \pi_\beta)\right], \quad (1)$$

where α is a scalar controlling the strength of regularization, $D(\cdot, \cdot)$ is a divergence function between the policy π and the behavior policy π_β. In offline RL, we do not have access to π_β; some prior methods attempt to estimate it empirically [Kostrikov *et al.*, 2021a; Li *et al.*, 2023] which is challenging when the dataset is generated by a mixture of policies. Furthermore, selecting the constraint strength can be challenging and difficult to generalize across datasets with similar environments [Tarasov *et al.*, 2023; Kostrikov *et al.*, 2021a].

Other policy constraint approaches use weighted BC [Nair *et al.*, 2020; Kostrikov *et al.*, 2021b; Xu *et al.*, 2023] or (surrogate) BC constraints [Li *et al.*, 2022; Wu *et al.*, 2019; Li *et al.*, 2023]. The former methods may be too restrictive as they do not allow OOD action selection, which is crucial to improve performance [Fu *et al.*, 2022]. The latter methods may still require substantial tuning and in addition to training if using model-based score methods. Other methods impose architectural constraints [Kumar *et al.*, 2019; Fujimoto *et al.*, 2019] that parameterize separate BC and reward-maximizing policy models.

Critic Regularization Critic regularization methods directly address the OOD action-value overestimation problem by penalizing large values for adversarially sampled actions [Kostrikov *et al.*, 2021a].

Ensembles Employing an ensemble of neural network estimators is a commonly used technique for prediction with a measure of epistemic uncertainty [Kondratyuk *et al.*, 2020]. A family of offline RL methods employ large ensembles of value functions [An *et al.*, 2021] and make use of the diversity of randomly initialized ensembles to implicitly reduce the selection of OOD actions or directly penalize the variance of the reward in the ensemble [Ghasemipour *et al.*, 2022; Sutton and Barto, 2018].

Model-Based Uncertainty Estimation Learning an uncertainty model of the dataset is often devised analogously to exploration-encouraging methods used in online RL, but, employing these for anti-exploration instead [Rezaeifar *et al.*, 2022]. An example is SAC-RND which directly adopts such an approach [Nikulin *et al.*, 2023]. Other algorithms include DOGE [Li *et al.*, 2022] which trains a model to estimate uncertainty as a distance to dataset action and DARL [Zhang *et al.*, 2023] which uses distance to random projections of state-action pairs as an uncertainty measure. As a whole, these methods optimize a distance $d(\cdot, \cdot) \geq 0$ that represents the uncertainty of an action.

2.2 Uncertainty Estimation

Neural networks are known to predict confidently even when presented with OOD samples [Nguyen *et al.*, 2015; Goodfellow *et al.*, 2014; Lakshminarayanan *et al.*, 2017]. A classical approach to OOD detection is to fit a generative model to the dataset that produces a high probability for in-dataset samples and a low probability for OOD ones. These methods work well for simple, unimodal data but can become computationally demanding for more complex data with multiple modes. Another approach trains classifiers that are leveraged to become finer-grained OOD detectors [Lee *et al.*, 2018]. In this work, we focus on Morse neural networks [Dherin *et al.*, 2023], an approach that trains a generative model to produce an unnormalized density that takes on value 1 at the dataset modes.

3 Preliminaries

A Morse neural network produces an unnormalized density $M(x) \in [0,1]$ on an embedding space $\mathbb{R}^e$ [Dherin *et al.*, 2023]. A Morse network can produce a density in $\mathbb{R}^e$ that attains a value of 1 at mode submanifolds and decreases towards 0 when moving away from the mode. The rate at which the value decreases is controlled by a Morse Kernel.

Definition 1 (Morse Kernel). *A Morse Kernel is a positive definite kernel K. When applied in a space $Z = \mathbb{R}^k$, the kernel $K(z_1, z_2)$ takes values in the interval $[0,1]$ where $K(z_1, z_2) = 1$ iff $z_1 = z_2$.*

All kernels of the form $K(z_1, z_2) = e^{-D(z_1,z_2)}$ where $D(\cdot,\cdot)$ is a divergence [Amari, 2016] are Morse Kernels. Examples include common kernels such as the Radial Basis Function (RBF) Kernel,

$$K_{RBF}(z_1, z_2) = e^{-\frac{\lambda^2}{2}||z_1 - z_2||^2}. \tag{2}$$

The RBF kernel and its derivatives decay exponentially, leading learning signals to vanish rapidly. An alternative is the ubiquitous Rational Quadratic (RQ) kernel:

$$K_{RQ}(z_1, z_2) = \left(1 + \frac{\lambda^2}{2\kappa} \mid\mid z_1 - z_2 \mid\mid^2\right)^{-\kappa} \tag{3}$$

where λ is a scale parameter in each kernel. The RQ kernel is a scaled mixture of RBF kernels controlled by κ and, for small κ, decays much more slowly [Williams and Rasmussen, 2006].

Consider a neural network that maps from a feature space into a latent space $f_\phi : X \to Z$, with parameters ϕ, $X \in \mathbb{R}^d$ and $Z \in \mathbb{R}^k$. A Morse Kernel can impose structure on the latent space.

Definition 2 (Morse Neural Network). *A Morse neural network is a function $f_\phi : X \to Z$ in combination with a Morse Kernel on $K(z, t)$ where $t \subset Z$ is a target, chosen as a hyperparameter of the model. The Morse neural network is defined as $M_\phi(x) = K(f_\phi(x), t)$.*

Using Definition 1 we see that $M_\phi(x) \in [0,1]$, and when $M_\phi(x) = 1$, x corresponds to a mode that coincides with the level set of the submanifold of the Morse neural network. Furthermore, $M_\phi(x)$ corresponds to the *certainty* of the sample x being from the training dataset, so $1 - M_\phi(x)$ is a measure of the epistemic uncertainty of x.

The function $-\log M_\phi(x)$ measures a squared distance, $d(\cdot,\cdot)$, between $f_\phi(x)$ and the closest mode in the latent space at m:

$$d(z) = \min_{m \in M} d(z, m), \tag{4}$$

where M is the set of all modes. This encodes information about the topology of the submanifold and satisfies the Morse–Bott non-degeneracy condition [Basu and Prasad, 2020].

The Morse neural network offers the following properties:

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

Proof of each property is provided in the appendix.

4 Policy Constraint with a Behavioral Supervisor

We now describe the constituent components of our algorithm, building on the Morse network and showing how it can be incorporated into a policy-regularized objective.

4.1 Morse Networks for Offline RL

The target t is a hyperparameter that must be chosen. Experiments in [Dherin *et al.*, 2023] use simple, toy datasets with classification problems that perform well for categorical t. We find that using a static label for the Morse network yields poor performance; rather than a labeling model, we treat f_ϕ as a perturbation model that produces an action $f_\phi(s,a) = \hat{a}$ such that $\hat{a} = a$ if and only if $s, a \sim \mathcal{D}$.

An offline RL dataset $\mathcal{D}$ consists of tuples $\{s, a, r, s'\}$ where we assume $\{s, a\}$ pairs are i.i.d. sampled from an unknown distribution. The Morse network must be fitted on N state-action pairs $[\{s_1, a_1, \}, ..., \{s_N, a_N\}]$ such that $M_\phi(s_i, a_j) = 1, \forall i, j \in 1, ..., N$] only when $i = j$.

We fit a Morse neural network to minimize the KL divergence between unnormalized measures [Amari, 2016] following [Dherin *et al.*, 2023], $D_{\text{KL}}(\mathcal{D}(s,a) \mid\mid M_\phi(s,a))$:

$$\min_\phi \mathbb{E}_{s,a\sim\mathcal{D}} \left[\log \frac{\mathcal{D}(s,a)}{M_\phi(s,a)}\right] + \int M_\phi(s,a) - \mathcal{D}(s,a)\, da. \tag{5}$$

With respect to ϕ, this amounts to minimizing the empirical loss:

$$L(\phi) = -\frac{1}{N} \sum_{s,a\sim\mathcal{D}} \log K(f_\phi(s,a), a) + \frac{1}{N} \sum_{\substack{s\sim\mathcal{D} \\ a_{\mathcal{D}}\sim\mathcal{D}_{\text{uni}}}} K(f_\phi(s, a_{\text{u}}), a_{\text{u}}), \tag{6}$$

where a_{u} is an action sampled from a uniform distribution over the action space $\mathcal{D}_{\text{uni}}$.

A learned Morse density is well suited to modeling ensemble policies [Lei *et al.*, 2023], more flexibly [Dherin *et al.*, 2023; Kostrikov *et al.*, 2021a; Li *et al.*, 2023] and without down-weighting good, in-support actions that have low density under the behavior policy [Singh *et al.*, 2022] as *all* modes have unnormalized density value 1.

A Morse neural network can be expressed as an energy-based model (EBM) [Goodfellow *et al.*, 2016]:

Proposition 1. *A Morse neural network can be expressed as an energy-based model: $E_\phi(x) = e^{-\log M_\phi(x)}$ where $M_\phi : \mathbb{R}^d \to \mathbb{R}$.*

Note that the EBM E_ϕ is itself unnormalized. Representing the Morse network as an EBM allows analysis analogous to [Florence *et al.*, 2022].

Theorem 1. *For a set-valued function $F(x) : x \in \mathbb{R}^m \to \mathbb{R}^n \backslash \{\emptyset\}$, there exists a continuous function $g : \mathbb{R}^{m+n} \to \mathbb{R}$ that is approximated by a continuous function approximator g_ϕ with arbitrarily small bounded error ϵ. This ensures that any point on the graph $F_\phi(x) = \arg\min_y g_\phi(x, y)$ is within distance ϵ of F.*

We refer the reader to [Florence *et al.*, 2022] for a detailed proof.

The theorem assumes that $F(x)$ is an implicit function and states that the error at the level-set (i.e. the modes) of $F(x)$ is small.

4.2 TD3-BST

We can use the Morse network to design a regularized policy objective. Recall that policy regularization consists of Q-value maximization and minimization of a distance to the behavior policy (Equation 1). We reconsider the policy regularization term and train a policy that minimizes uncertainty while selecting actions close to the behavior policy. Let $C^\pi(s, a)$ denote a measure of uncertainty of the policy action. We solve the following optimization problem:

$$\pi^{i+1} = \arg\min_{\pi \in \Pi} \mathbb{E}_{a \sim \pi(\cdot|s)} \left[C^\pi(s, a)\right] \tag{7}$$

$$\text{s.t.} \quad D_{\text{KL}} \left(\pi(\cdot \mid s) \mid\mid \pi_\beta(\cdot \mid s)\right) \leq \epsilon. \tag{8}$$

This optimization problem requires an explicit behavior model, which is difficult to estimate and using an estimated model has historically returned mixed results [Kumar *et al.*, 2019; Fujimoto *et al.*, 2019]. Furthermore, this requires direct optimization through C^π which may be subject to exploitation. Instead, we enforce this *implicitly* by deriving the solution to the constrained optimization to obtain a closed-form solution for the actor [Peng *et al.*, 2019; Nair *et al.*, 2020]. Enforcing the KKT conditions we obtain the Lagrangian:

$$L(\pi, \mu) = \mathbb{E}_{a \sim \pi(\cdot|s)} \left[C^\pi(s, a)\right] + \mu(\epsilon - D_{\text{KL}}(\pi \mid\mid \pi_\beta)). \tag{9}$$

Computing $\frac{\partial L}{\partial \pi}$ and solving for π yields the uncertainty minimizing solution $\pi^{C^*}(a \mid s) \propto \pi_\beta(a \mid s) e^{\frac{1}{\mu} C^\pi(s,a)}$. When learning the parametric policy π_ψ, we project the non-parametric solution into the policy space as a (reverse) KL divergence minimization of π_ψ under the data distribution $\mathcal{D}$:

$$\arg\min_\psi \mathbb{E}_{s \sim \mathcal{D}} \left[D_{\text{KL}} \left(\pi^{C^*}(\cdot \mid s) \mid\mid \pi_\psi(\cdot \mid s)\right)\right] \tag{10}$$

$$= \arg\min_\psi \mathbb{E}_{s \sim \mathcal{D}} \left[D_{\text{KL}} \left(\pi_\beta(a \mid s) e^{\frac{1}{\mu} C^\pi(s,a)} \mid\mid \pi_\psi(\cdot \mid s)\right)\right] \tag{11}$$

$$= \arg\min_\psi \mathbb{E}_{s,a \sim \mathcal{D}} \left[-\log \pi_\psi(a \mid s) e^{\frac{1}{\mu} C^\pi(s,a)}\right], \tag{12}$$

which is a weighted maximum likelihood update where the supervised target is sampled from the dataset $\mathcal{D}$ and $C^\pi(s, a) = 1 - M_\phi(s, \pi_\psi(s))$. This avoids explicitly modeling the behavior policy and uses the Morse network uncertainty as a *behavior supervisor* to dynamically adjust the strength of behavioral cloning. We provide a more detailed derivation in the appendix.

Interpretation Our regularization method shares similarities with other weighted regression algorithms [Nair *et al.*, 2020; Peng *et al.*, 2019; Kostrikov *et al.*, 2021b] which weight the advantage of an action compared to the dataset/replay buffer action. Our weighting can be thought of as a measure of *disadvantage* of a policy action in the sense of how OOD it is.

We make modifications to the behavioral cloning objective. From Morse network property (1) we know $M_\phi \in [0, 1]$, hence $1 \leq e^{\frac{1}{\mu} C^\pi} \leq e^{\frac{1}{\mu}}$, i.e. the lowest possible disadvantage coefficient is 1. To minimize the coefficient in the mode, we require it to approach 0 when near a mode. We adjust the weighted behavioral cloning term and add Q-value maximization to yield the regularized policy update:

$$\begin{aligned} \pi^{i+1} \leftarrow \arg\max_\pi \mathbb{E}_{\substack{s,a \sim \mathcal{D}, \\ a_\pi \sim \pi^i(s)}} & \Big[\frac{1}{Z_Q} Q^{i+1}(s, a_\pi) \\ & - (e^{\frac{1}{\mu} C^\pi(s,a)} - 1)(a_\pi - a)^2\Big], \end{aligned} \tag{13}$$

where μ is the Lagrangian multiplier that controls the magnitude of the disadvantage weight and $Z_Q = \frac{1}{N} \sum_{n=1}^{N} |Q(s, a_\pi)|$ is a scaling term detached from the gradient update process [Fujimoto and Gu, 2021], necessary as $Q(s, a)$ can be arbitrarily large and the BC-coefficient is upper-bounded at $e^{\frac{1}{\mu}}$.

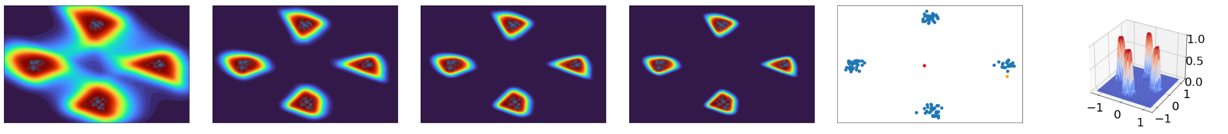
The value function update is given by:

$$Q^{i+1} \leftarrow \arg\min_Q \mathbb{E}_{s,a,s' \sim \mathcal{D}}[(y - Q^i(s, a))^2], \tag{14}$$

with $y = r(s, a) + \gamma \mathbb{E}_{s' \sim \bar{\pi}(s')} \bar{Q}(s', a')$ where $\bar{Q}$ and $\bar{\pi}$ are target value and policy functions, respectively.

4.3 Controlling the Tradeoff Constraint

Tuning TD3-BST is straightforward; the primary hyperparameters of the Morse network consist of the choice and scale of the kernel, and the temperature μ. Increasing λ for higher dimensional actions ensures that the high certainty region around modes remains tight. Prior empirical work has demonstrated the importance of allowing some degree of OOD actions [An *et al.*, 2021]; in the TD3-BST framework, this is dependent on λ. In Figure 2 we provide a didactic example of the effect of λ. We construct a dataset consisting of 2-dimensional actions in $[-1, 1]$ with means at the four locations $\{[0.0, 0.8], [0.0, -0.8], [0.8, 0.0], [-0.8, 0.0]\}$ and each with standard deviation 0.05. We sample $M = 128$ points, train a Morse network and plot the density produced by the Morse network for $\lambda = \{\frac{1}{10}, \frac{1}{2}, 1.0, 2.0\}$. A behavioral cloning policy learned using vanilla MLE where all targets are weighted equally results in an OOD action being selected. Training using Morse-weighted BC downweights the behavioral cloning loss for far away modes, enabling the policy to select and minimize error to a single mode.



(a) $\lambda = 0.1$ (b) $\lambda = 0.5$ (c) $\lambda = 1.0$ (d) $\lambda = 2.0$ (e) Ground Truth (f) Density $\lambda = 1.0$

Figure 2: a-d: Contour plots of unnormalized densities ($\in [0, 1]$) produced by a Morse network for increasing λ with ground truth actions included as $\times$ marks. e: Ground truth actions (blue) in the synthetic dataset, the MLE action (red). A Morse certainty weighted MLE model can select actions in a single mode, in this case, the (right-hand side) mode centred at [0.8, 0.0] (orange). Weighting a divergence constraint using a Morse (un)certainty will encourage the policy to select actions near the modes of M_ϕ that maximize reward.

Algorithm 1 TD3-BST Training Procedure Outline. The policy is updated once for every $m = 2$ critic updates, as is the default in TD3.

Input: Dataset $\mathcal{D} = \{s, a, r, s'\}$
Initialize: Initialize Morse network M_ϕ.
Output: Trained Morse network M_ϕ.

Let $t = 0$.
for t = 1 **to** T_M **do**
  Sample minibatch $(s, a) \sim \mathcal{D}$
  Sample random actions $a_{\bar{\mathcal{D}}} \sim \mathcal{D}_{\text{uni}}$ for each state s
  Update ϕ by minimizing Equation 6
end for

Initialize: Initialize policy network π_ψ, critic Q_θ, target policy $\bar{\psi} \leftarrow \psi$ and target critic $\bar{\theta} \leftarrow \theta$.
Output: Trained policy π.

Let $t = 0$.
for t = 1 **to** T_{AC} **do**
  Sample minibatch $(s, a, r, s') \sim \mathcal{D}$
  Update θ using Equation 14
  if t mod $m = 0$ **then**
    Obtain $a_\pi = \pi(s)$
    Update ψ using Equation 13
    Update target networks $\bar{\theta} \leftarrow \rho\theta + (1 - \rho)\bar{\theta}$, $\bar{\psi} \leftarrow \rho\psi + (1 - \rho)\bar{\psi}$
  end if
end for
return π

4.4 Algorithm Summary

Fitting the Morse Network The TD3-BST training procedure is described in Algorithm 1. The first phase fits the Morse network for T_M gradient steps.

Actor–Critic Training In the second phase of training, a modified TD3-BC procedure is used for T_{AC} iterations with alterations highlighted in red.

We provide full hyperparameter details in the appendix.

5 Experiments

In this section, we conduct experiments that aim to answer the following questions:

- How does TD3-BST compare to other baselines, with a focus on comparing to newer baselines that use per-dataset tuning?
- Can the BST objective improve performance when used with one-step methods (IQL) that perform in-sample policy evaluation?
- How well does the Morse network learn to discriminate between in-dataset and OOD actions?
- How does changing the kernel scale parameter λ affect performance?
- Does using independent ensembles, a second method of uncertainty estimation, improve performance?

We evaluate our algorithm on the D4RL benchmark [Fu *et al.*, 2020], including the Gym Locomotion and challenging Antmaze navigation tasks.

5.1 Comparison with SOTA Methods

We evaluate TD3-BST against the older, well known baselines of TD3-BC [Fujimoto and Gu, 2021], CQL [Kumar *et al.*, 2020], and IQL [Kostrikov *et al.*, 2021b]. There are more recent methods that consistently outperform these baselines; of these, we include SQL [Xu *et al.*, 2023], SAC-RND [Nikulin *et al.*, 2023], DOGE [Li *et al.*, 2022], VMG [Zhu *et al.*, 2022], ReBRAC [Tarasov *et al.*, 2023], CFPI [Li *et al.*, 2023] and MSG [Ghasemipour *et al.*, 2022] (to our knowledge, the best-performing ensemble-based method). It is interesting to note that most of these baselines implement policy constraints, except for VMG (graph-based planning) and MSG (policy constraint using a large, independent ensemble). We note that *all* the aforementioned SOTA methods (except SQL) report scores with per-dataset tuned parameters in stark contrast with the older TD3-BC, CQL, and IQL algorithms, which use the same set of hyperparameters in each D4RL domain. All scores are reported with 10 evaluations in Locomotion and 100 in Antmaze across five seeds.

We present scores for D4RL Gym Locomotion in Table 1. TD3-BST achieves best or near-best results compared to all previous methods and recovers expert performance on five of nine datasets. The best performing prior methods include SAC-RND and ReBRAC, both of which require per-dataset tuning of BRAC-variant algorithms [Wu *et al.*, 2019].

We evaluate TD3-BST on the more challenging Antmaze tasks which contain a high degree of suboptimal trajectories and follow a sparse reward scheme that requires algorithms to *stitch* together several trajectories to perform well. TD3-BST achieves the best scores overall in Table 2, especially as the maze becomes more complex. VMG and MSG are the best-performing prior baselines and TD3-BST is far simpler and more efficient in its design as a variant of TD3-BC. The authors of VMG report the best scores from checkpoints rather than from the final policy. MSG report scores from ensembles with both 4 and 64 critics of which the best scores included here are from the 64-critic variant.

We pay close attention to SAC-RND, which, among all baselines, is most similar in its inception to TD3-BST. SAC-RND uses a random and trained network pair to produce a dataset-constraining penalty. SAC-RND achieves consistent SOTA scores on locomotion datasets, but fails to deliver commensurate performance on Antmaze tasks. TD3-BST performs similarly to SAC-RND in locomotion and achieves SOTA scores in Antmaze.

5.2 Improving One-Step Methods

One-step algorithms learn a policy from an offline dataset, thus remaining on-policy [Rummery and Niranjan, 1994; Sutton and Barto, 2018], and using weighted behavioral cloning [Brandfonbrener *et al.*, 2021; Kostrikov *et al.*, 2021b]. Empirical evaluation by [Fu *et al.*, 2022] suggests that advantage-weighted BC is too restrictive and relaxing the policy objective to Equation 1 can lead to performance improvement. We use the BST objective as a drop-in replacement for the policy improvement step in IQL [Kostrikov *et al.*, 2021b] to learn an optimal policy while retaining in-sample policy evaluation.

We reproduce IQL results and report scores for IQL-BST, both times using a deterministic policy [Tarasov *et al.*, 2022] and identical hyperparameters to the original work in Table 3. Reproduced IQL closely matches the original results, with slight performance reductions on the `-large` datasets. Relaxing weighted-BC with a BST objective leads to improvements in performance, especially on the more difficult `-medium` and `-large` datasets. To isolate the effect of the BST objective, we do not perform any additional tuning.

5.3 Ablation Experiments

Morse Network Analysis We analyze how well the Morse network can distinguish between dataset tuples and samples from $\mathcal{D}_{\text{perm}}$, permutations of dataset actions, and $\mathcal{D}_{\text{uni}}$. We plot both certainty (M_ϕ) density and t-SNEs [Van der Maaten and Hinton, 2008] in Figure 3 which show that the unsupervised Morse network is effective in distinguishing between $\mathcal{D}_{\text{perm}}$ and $\mathcal{D}_{\text{uni}}$ and assigning high certainty to dataset tuples.

Ablating kernel scale We examine sensitivity to the kernel scale λ. Recall that $k = \dim(\mathcal{A})$. We see in Figure 4 that the scale $\lambda = \frac{k}{2}$ is a performance sweet-spot on the challenging Antmaze tasks. We further illustrate this by plotting policy deviations from dataset actions in Figure 5. The scale $\lambda = 1.0$ is potentially too lax a behavioral constraint, while $\lambda = k$ is too strong, resulting in performance reduction. However, performance on all scales remains strong and compares well with most prior algorithms. Performance may be further improved by tuning λ, possibly with separate scales for each input dimension.

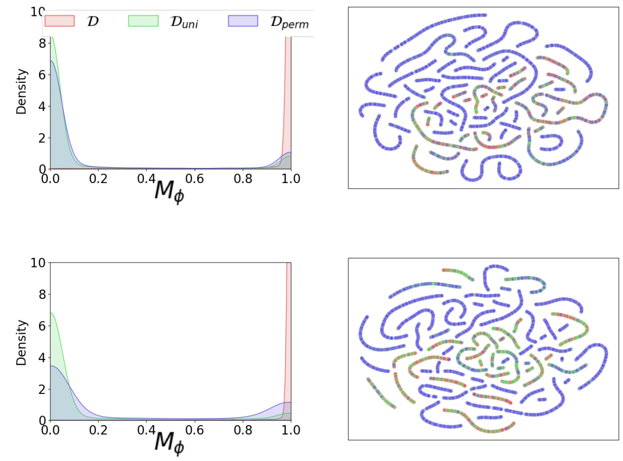


Figure 3: M_ϕ densities and t-SNE of deviations for `hopper-medium-expert` (top row) and `Antmaze-large-diverse` (bottom row). Density plots are clipped at 10.0 as density for $\mathcal{D}$ is large. 10 actions are sampled from $\mathcal{D}_{\text{uni}}$ and $\mathcal{D}_{\text{perm}}$ each, per state.

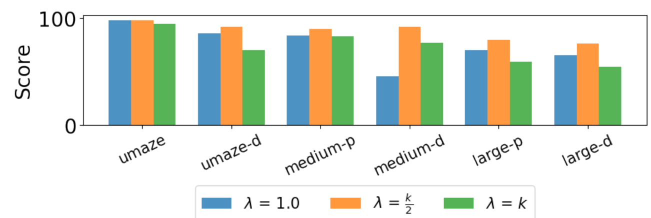


Figure 4: Ablations of λ on Antmaze datasets. Recall $k = \dim(\mathcal{A})$.

Independent or Shared Targets? Standard TD3 employs Clipped Double Q-learning (CDQ) [Hasselt, 2010; Fujimoto *et al.*, 2018] to prevent value overestimation. On tasks with sparse rewards, this may be too conservative [Moskovitz *et al.*, 2021]. MSG [Ghasemipour *et al.*, 2022] uses large ensembles of fully independent Q functions to learn offline. We examine how independent double Q functions perform compared to the standard CDQ setup in Antmaze with 2 and 10 critics. The results in Figure 6 show that disabling CDQ with 2 critics is consistently detrimental to performance. Using a larger 10-critic ensemble leads to moderate improvements. This suggests that combining policy regularization with an efficient, independent ensemble could bring further performance benefits with minimal changes to the algorithm.

6 Discussion

Morse Network In [Dherin *et al.*, 2023], deeper architectures are required even when training on simple datasets. This rings true for our application of Morse networks in this work, with low-capacity networks performing poorly. Training the Morse network for each locomotion and Antmaze dataset typically takes 10 minutes for 100 000 gradient steps using a batch size of 1 024. When training the policy, using the Morse network increases training time by approximately 15%.

Dataset	TD3-BC	CQL	IQL	SQL	SAC-RND[1]	DOGE	ReBRAC	CFPI	**TD3-BST (ours)**
halfcheetah-m	48.3	44.0	47.4	48.3	**66.6**	45.3	65.6	52.1	62.1 ± 0.8
hopper-m	59.3	58.5	66.3	75.5	97.8	98.6	102.0	86.8	**102.9 ± 1.3**
walker2d-m	83.7	72.5	78.3	84.2	**91.6**	86.8	82.5	88.3	90.7 ± 2.5
halfcheetah-m-r	44.6	45.5	44.2	44.8	42.8	**54.9**	51.0	44.5	53.0 ± 0.7
hopper-m-r	60.9	95.0	94.7	99.7	100.5	76.2	98.1	93.6	**101.2 ± 4.9**
walker2d-m-r	81.8	77.2	73.9	81.2	88.7	87.3	77.3	78.2	**90.4 ± 8.3**
halfcheetah-m-e	90.7	91.6	86.7	94.0	**107.6**	78.7	101.1	97.3	100.7 ± 1.1
hopper-m-e	98.0	105.4	91.5	**111.8**	109.8	102.7	107.0	104.2	110.3 ± 0.9
walker2d-m-e	110.1	108.8	109.6	110.0	105.0	110.4	111.6	**111.9**	109.4 ± 0.2

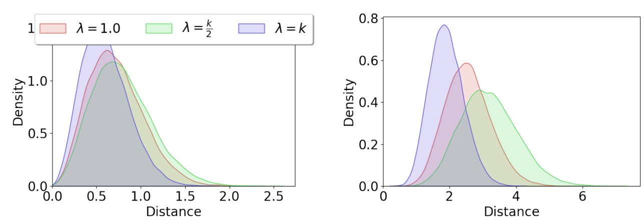
Table 1: Normalized scores on D4RL Gym Locomotion datasets. VMG scores are excluded because this method performs poorly and the authors of MSG do not report numerical results on locomotion tasks. Prior methods are grouped by those that do not perform per-dataset tuning and those that do. [1] SAC-RND in addition to per-dataset tuning, is trained for 3 million gradient steps. Though not included here, ensemble methods may perform better than the best non-ensemble methods on some datasets, albeit still requiring per-dataset tuning to achieve their reported performance. Top scores are in **bold** and second-best are underlined.

Dataset	TD3-BC	CQL	IQL	SQL	SAC-RND[1]	DOGE	VMG[2]	ReBRAC	CFPI	MSG[3]	**TD3-BST (ours)**
-umaze	78.6	74.0	87.5	92.2	97.0	97.0	93.7	97.8	90.2	**98.6**	97.8 ± 1.0
-umaze-d	71.4	84.0	62.2	74.0	66.0	63.5	**94.0**	88.3	58.6	81.8	91.7 ± 3.2
-medium-p	10.6	61.2	71.2	80.2	74.7	80.6	82.7	84.0	75.2	89.6	**90.2 ± 1.8**
-medium-d	3.0	53.7	70.0	79.1	74.7	77.6	84.3	76.3	72.2	88.6	**92.0 ± 3.8**
-large-p	0.2	15.8	39.6	53.2	43.9	48.2	67.3	60.4	51.4	72.6	**79.7 ± 7.6**
-large-d	0.0	14.9	47.5	52.3	45.7	36.4	74.3	54.4	52.4	71.4	**76.1 ± 4.7**

Table 2: Normalized scores on D4RL Antmaze datasets. [1] SAC-RND is trained for three million gradient steps. [2] VMG reports scores from the best-performing checkpoint rather than from the final policy; despite this, TD3-BST still outperforms VMG in all datasets except -umaze-diverse. [3] for MSG we report the best score among the reported scores of all configurations, also, MSG is trained for two million steps. Prior methods are grouped by those that do not perform per-dataset tuning and those that do. Other ensemble-based methods are not included, as MSG achieves higher performance. Top scores are in **bold** and second-best are underlined.

Dataset	IQL (reproduced)	IQL-BST
-umaze	87.6 ± 4.6	**90.8 ± 2.1**
-umaze-d	**64.0 ± 5.2**	63.1 ± 3.7
-medium-p	70.7 ± 4.3	**80.3 ± 1.3**
-medium-d	73.8 ± 5.9	**84.7 ± 2.0**
-large-p	35.2 ± 8.4	**55.4 ± 3.2**
-large-d	40.7 ± 9.2	**51.6 ± 2.6**

Table 3: Normalized scores on D4RL Antmaze datasets for IQL and IQL-BST. We use hyperparameters identical to the original IQL paper and use Equation 13 as the policy objective.



(a) hopper-medium (b) amaze-large-play

Figure 5: Histograms of deviation from dataset actions.

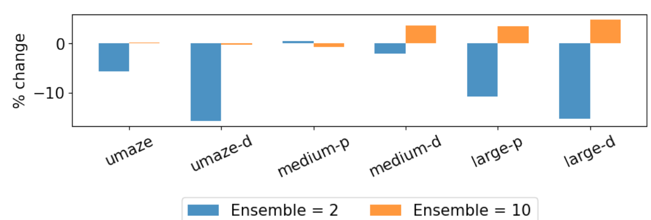


Figure 6: % change in Antmaze scores without CDQ for critic ensembles consisting of 2 and 10 Q functions.

Optimal Datasets On Gym Locomotion tasks TD3-BST performance is comparable to newer methods, all of which rarely outperform older baselines. This can be attributed to a significant proportion of high-return-yielding trajectories that are easier to improve.

7 Conclusion

In this paper, we introduce TD3-BST, an algorithm that uses an uncertainty model to dynamically adjust the strength of regularization. Dynamic weighting allows the policy to maximize reward around individual dataset modes. Our algorithm compares well against prior methods on Gym Locomotion tasks and achieves the best scores on the more challenging Antmaze tasks, demonstrating strong performance when learning from suboptimal data.

In addition, our experiments show that combining our policy regularization with an ensemble-based source of uncertainty can improve performance. Future work can explore how to apply alternative uncertainty measures and how best to combine multiple sources of uncertainty.

References

[Amari, 2016] Shun-ichi Amari. *Information geometry and its applications*, volume 194. Springer, 2016.

[An *et al.*, 2021] Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified Q-ensemble. *Advances in neural information processing systems*, 34:7436–7447, 2021.

[Basu and Prasad, 2020] Somnath Basu and Sachchidanand Prasad. A connection between cut locus, thom space and morse-bott functions. *arXiv preprint arXiv:2011.02972*, 2020.

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

[Dherin *et al.*, 2023] Benoit Dherin, Huiyi Hu, Jie Ren, Michael W Dusenberry, and Balaji Lakshminarayanan. Morse neural networks for uncertainty quantification. *arXiv preprint arXiv:2307.00667*, 2023.

[Emerson *et al.*, 2023] Harry Emerson, Matthew Guy, and Ryan McConville. Offline reinforcement learning for safer blood glucose control in people with type 1 diabetes. *Journal of Biomedical Informatics*, 142:104376, 2023.

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

[Fu *et al.*, 2020] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[Fu *et al.*, 2022] Yuwei Fu, Di Wu, and Benoit Boulet. A closer look at offline RL agents. *Advances in Neural Information Processing Systems*, 35:8591–8604, 2022.

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

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

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

[Ghasemipour *et al.*, 2022] Kamyar Ghasemipour, Shixiang Shane Gu, and Ofir 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.

[Goodfellow *et al.*, 2014] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[Goodfellow *et al.*, 2016] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[Gulcehre *et al.*, 2020] Caglar Gulcehre, Sergio Gómez Colmenarejo, Jakub Sygnowski, Thomas Paine, Konrad Zolna, Yutian Chen, Matthew Hoffman, Razvan Pascanu, and Nando de Freitas. Addressing extrapolation error in deep offline reinforcement learning. 2020.

[Hasselt, 2010] Hado Hasselt. Double Q-learning. *Advances in neural information processing systems*, 23, 2010.

[Kondratyuk *et al.*, 2020] Dan Kondratyuk, Mingxing Tan, Matthew Brown, and Boqing Gong. When ensembling smaller models is more efficient than single large models. *arXiv preprint arXiv:2005.00570*, 2020.

[Kostrikov *et al.*, 2021a] Ilya Kostrikov, Rob Fergus, Jonathan Tompson, and Ofir Nachum. Offline reinforcement learning with Fisher divergence critic regularization. In *International Conference on Machine Learning*, pages 5774–5783. PMLR, 2021.

[Kostrikov *et al.*, 2021b] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit Q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

[Kumar *et al.*, 2019] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy Q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 32, 2019.

[Kumar *et al.*, 2020] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

[Lakshminarayanan *et al.*, 2017] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[Lange *et al.*, 2012] Sascha Lange, Thomas Gabel, and Martin Riedmiller. *Batch reinforcement learning*, pages 45–73. Springer, 2012.

[Lee *et al.*, 2018] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in neural information processing systems*, 31, 2018.

[Lei *et al.*, 2023] Kun Lei, Zhengmao He, Chenhao Lu, Kaizhe Hu, Yang Gao, and Huazhe Xu. Uni-o4: Unifying online and offline deep reinforcement learning with multi-step on-policy optimization. *arXiv preprint arXiv:2311.03351*, 2023.

[Li *et al.*, 2022] Jianxiong Li, Xianyuan Zhan, Haoran Xu, Xiangyu Zhu, Jingjing Liu, and Ya-Qin Zhang. When data geometry meets deep function: Generalizing offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2022.

[Li *et al.*, 2023] Jiachen Li, Edwin Zhang, Ming Yin, Qinxun Bai, Yu-Xiang Wang, and William Yang Wang. Offline reinforcement learning with closed-form policy improvement operators. In *International Conference on Machine Learning*, pages 20485–20528. PMLR, 2023.

[Mirowski *et al.*, 2018] Piotr Mirowski, Matt Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Andrew Zisserman, and Raia Hadsell. Learning to navigate in cities without a map. *Advances in neural information processing systems*, 31, 2018.

[Moskovitz *et al.*, 2021] Ted Moskovitz, Jack Parker-Holder, Aldo Pacchiano, Michael Arbel, and Michael Jordan. Tactical optimism and pessimism for deep reinforcement learning. *Advances in Neural Information Processing Systems*, 34:12849–12863, 2021.

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

[Nguyen *et al.*, 2015] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

[Nikulin *et al.*, 2023] Alexander Nikulin, Vladislav Kurenkov, Denis Tarasov, and Sergey Kolesnikov. Anti-exploration by random network distillation. *arXiv preprint arXiv:2301.13616*, 2023.

[Peng *et al.*, 2019] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

[Rezaeifar *et al.*, 2022] Shideh Rezaeifar, Robert Dadashi, Nino Vieillard, Léonard Hussenot, Olivier Bachem, Olivier Pietquin, and Matthieu Geist. Offline reinforcement learning as anti-exploration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8106–8114, 2022.

[Rummery and Niranjan, 1994] Gavin A Rummery and Mahesan Niranjan. *On-line Q-learning using connectionist systems*, volume 37. University of Cambridge, Department of Engineering Cambridge, UK, 1994.

[Singh *et al.*, 2022] Anikait Singh, Aviral Kumar, Quan Vuong, Yevgen Chebotar, and Sergey Levine. Offline RL with realistic datasets: Heteroskedasticity and support constraints. *arXiv preprint arXiv:2211.01052*, 2022.

[Sutton and Barto, 2018] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[Tang and Wiens, 2021] Shengpu Tang and Jenna Wiens. Model selection for offline reinforcement learning: Practical considerations for healthcare settings. In *Machine Learning for Healthcare Conference*, pages 2–35. PMLR, 2021.

[Tarasov *et al.*, 2022] Denis Tarasov, Alexander Nikulin, Dmitry Akimov, Vladislav Kurenkov, and Sergey Kolesnikov. CORL: Research-oriented deep offline reinforcement learning library. *arXiv preprint arXiv:2210.07105*, 2022.

[Tarasov *et al.*, 2023] Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. *arXiv preprint arXiv:2305.09836*, 2023.

[Van der Maaten and Hinton, 2008] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11), 2008.

[Williams and Rasmussen, 2006] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

[Wu *et al.*, 2019] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[Xu *et al.*, 2023] Haoran Xu, Li Jiang, Jianxiong Li, Zhuoran Yang, Zhaoran Wang, Victor Wai Kin Chan, and Xianyuan Zhan. Offline RL with no OOD actions: In-sample learning via implicit value regularization. *arXiv preprint arXiv:2303.15810*, 2023.

[Yu *et al.*, 2021] Chao Yu, Jiming Liu, Shamim Nemati, and Guosheng Yin. Reinforcement learning in healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(1):1–36, 2021.

[Zhang and Jiang, 2021] Siyuan Zhang and Nan Jiang. Towards hyperparameter-free policy selection for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34:12864–12875, 2021.

[Zhang *et al.*, 2023] Hongchang Zhang, Jianzhun Shao, Shuncheng He, Yuhang Jiang, and Xiangyang Ji. DARL: distance-aware uncertainty estimation for offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 11210–11218, 2023.

[Zhu *et al.*, 2022] Deyao Zhu, Li Erran Li, and Mohamed Elhoseiny. Value memory graph: A graph-structured world model for offline reinforcement learning. *arXiv preprint arXiv:2206.04384*, 2022.

[Zhu *et al.*, 2023] Taiyu Zhu, Kezhi Li, and Pantelis Georgiou. Offline deep reinforcement learning and off-policy evaluation for personalized basal insulin control in type 1 diabetes. *IEEE Journal of Biomedical and Health Informatics*, 2023.