

Diffusion Policies for Out-of-Distribution Generalization in Offline Reinforcement Learning

Suzan Ece Ada¹, Erhan Oztop², *Member, IEEE*, and Emre Ugur¹

Abstract—Offline Reinforcement Learning (RL) methods leverage previous experiences to learn better policies than the behavior policy used for data collection. However, they face challenges handling distribution shifts due to the lack of online interaction during training. To this end, we propose a novel method named State Reconstruction for Diffusion Policies (SRDP) that incorporates state reconstruction feature learning in the recent class of diffusion policies to address the problem of out-of-distribution (OOD) generalization. Our method promotes learning of generalizable state representation to alleviate the distribution shift caused by OOD states. To illustrate the OOD generalization and faster convergence of SRDP, we design a novel 2D Multimodal Contextual Bandit environment and realize it on a 6-DoF real-world UR10 robot, as well as in simulation, and compare its performance with prior algorithms. In particular, we show the importance of the proposed state reconstruction via ablation studies. In addition, we assess the performance of our model on standard continuous control benchmarks (D4RL), namely the navigation of an 8-DoF ant and forward locomotion of half-cheetah, hopper, and walker2d, achieving state-of-the-art results. Finally, we demonstrate that our method can achieve 167% improvement over the competing baseline on a sparse continuous control navigation task where various regions of the state space are removed from the offline RL dataset, including the region encapsulating the goal.

Index Terms—Reinforcement Learning, Deep Learning Methods, Learning from Demonstration

I. INTRODUCTION

LEVERAGING large datasets and generalizing to unforeseen situations are critical components of intelligent systems. Offline Reinforcement Learning (RL) has garnered significant attention for learning from previously collected datasets without interacting with the real world [1]. Similarly, Out-of-Distribution (OOD) generalization is crucial for developing reliable systems adapting to unexpected conditions. While offline RL promises to find better policies than the behavior policy that generated the trajectories in the dataset

without making assumptions about the agents' expertise, they struggle when faced with states not present in the training set. Hence, we aim to develop a generalizable generative offline RL model, State Reconstruction for Diffusion Policies (SRDP), to learn robust skills from offline datasets by extrapolating sequential decision-making to OOD states.

The main challenges of offline RL are the distribution shift and uncertainty estimation [2]. Since state and action distributions in the offline dataset can differ from those encountered in the evaluation environment, dealing with OOD state and action samples is a prominent topic in offline RL [3]. Thus, we are interested in policy-regularized offline RL algorithms where the divergence from the behavior policy that collected the dataset is discouraged.

There is a growing interest in applying diffusion models in robotics. Notably, Chi et al. [4] have demonstrated the effectiveness of convolutional and transformer-based diffusion networks in representing multimodal action distributions with visuomotor policy learning for behavior cloning. Likewise, other behavior cloning [5], [6] methods have also learned multimodal expert behavior, though not specifically targeting offline RL and diffusion models. Despite these achievements, our current focus is on OOD generalization in offline RL without a visual component while believing in the potential applicability of our findings in visuomotor policy learning. Diffusion-QL [7] and Diffusers [8] are recent RL algorithms that utilize diffusion models [9]–[11] by guiding the diffusion process toward regions that can yield a high reward. Diffusers [8] use diffusion models in the planning procedure to generate trajectories from the diffusion model, while Diffusion-QL [7] generates actions based on a state-conditional diffusion model and uses Q-function maximization with behavior cloning loss. Even though Diffusion-QL [7] can represent multimodal actions, it is often unstable in OOD state regions.

Autoencoders can reconstruct a subset of OOD samples with low error via learning representations in the bottleneck layer [12]. Denouden et al. [12] highlight that OOD samples close to a linear or a nonlinear latent dimension manifold of the training data can have a low reconstruction loss. Hence, guidance from a reconstruction loss and similar architectural designs can benefit OOD generalization. Mutlu et al. [13] use reconstruction loss to provide hints to the generator. Although we employ separate output heads to integrate the state reconstruction loss, we use a shared representation layer to generalize to OOD states close to the latent dimension manifold.

Our key contributions include SRDP, a new offline RL method that alleviates the distribution shifts incurred by OOD

Manuscript received: September 3, 2023; Revised: December 16, 2023; Accepted: January 16, 2024.

This paper was recommended for publication by Editor Aleksandra Faust upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by Grant-in-Aid for Scientific Research – the project with number 22H03670, the project JPNP16007 commissioned by the New Energy and Industrial Technology Development Organization (NEDO), the Scientific and Technological Research Council of Turkey (TUBITAK, 118E923), and INVERSE project (101136067) funded by the European Union.

¹Suzan Ece Ada and Emre Ugur are with Department of Computer Engineering, Bogazici University, Istanbul, Turkey {ece.ada, emre.ugur}@bogazici.edu.tr

²Erhan Oztop is with SISReC, OTRI, Osaka University, Japan, and Department of Computer Science, Ozyegin University, Istanbul, Turkey erhan.oztop@otri.osaka-u.ac.jp

Digital Object Identifier (DOI): see top of this page.

Copyright ©2024 IEEE

states using representation learning. SRDP is tailored to guide diffusion policies using a state reconstruction signal. Through extensive experiments, ablation studies, as well as real robot experiments, we demonstrate that incorporating state reconstruction signal at each diffusion timestep not only enhances the performance of foundational diffusion models in severe OOD settings where a large portion of data is missing during training but also in widely used offline RL tasks.

II. RELATED WORK

A. Offline Reinforcement Learning

In **dynamic programming-based offline RL** methods, the Q-function is approximated using the offline RL dataset that comprises samples collected by the behavior policy. As the learned policy diverges from the behavior policy, Q-function estimates tend to exhibit overestimation in regions where uncertainty is high [14]. Prior works impose a constraint on the statistical distance between the learned policy and the behavior policy in the policy optimization objective [2] or the reward function [15]. However, both methods require the computation of the behavior policy through behavior cloning and enforce a constraint on the learned policy. Nair et al. [16] alleviated the need for computing the behavior policy by approximating the policy objective using an advantage-weighted maximum likelihood. Still, this method is prone to distribution shift as the policy can query the OOD actions during training. Recent work addressed this distribution shift by avoiding OOD actions in the Q-function estimation using an expected loss instead of the mean squared error loss [17]. As an alternative, in Conservative Q-Learning (CQL) [18], the Q-function is approximated using a minimax objective where overestimated action values are minimized, and actions from the dataset are maximized. However, conservative estimation of the value function is prone to overfitting when data is scarce [2].

Importance sampling-based offline RL methods attempt to approximate the learned policy or expected return via off-policy RL techniques. However, off-policy RL allows environment interaction in the training loop, whereas offline RL learns from a static replay buffer constructed before training. The accuracy of the importance sampling estimator depends on the proximity of the learned policy to the policy used for data collection, the dimensionality of the state-action space, and the horizon of the task. Hence, these algorithms have limited applicability to real tasks and impose constraints on our objective of obtaining the best policy supported by the data. Prior work on importance sampling-based offline RL addresses the bias-variance trade-off. The marginal importance ratio [19] can be used via dynamic programming to reduce bias. Doubly robust estimator [20] uses recursive regression-based value evaluation to reduce variance. However, these methods are still prone to distribution shifts, as they exploit OOD regions in the policy improvement step. Hence, this work uses an auxiliary state reconstruction signal in the policy improvement step that encourages learning more generalizable state features.

Model-based offline RL methods focus on deriving the environment model by estimating the transition function, unlike

model-free methods. Although, theoretically, their advantage over model-free methods has not been proven [2], they offer sample efficiency and quick adaptation. In the model-based offline RL setting, the model cannot correct OOD states in addition to the OOD actions. To avoid OOD states and actions, a scaled uncertainty function penalizes the reward function over state action pairs [21]. However, estimating an uncertainty function that accurately quantifies uncertainty regions over the state action space is a challenging open problem [2]. Conservative Model-Based-RL [22], on the other hand, follows a similar approach used in CQL [18] and penalizes overestimated Q-values of state action tuples sampled from the model distribution. Most model-based RL algorithms are myopic and fail to make accurate predictions in tasks where the dimensions of the state action space are high [2]. Recent works have viewed the problem through the lens of sequence modeling and used high-capacity transformers [23], [24]. However, these architectures are computationally expensive to train as they need careful tuning of hyperparameters. Validation and hyperparameter optimization in offline RL remains an open area of research.

B. Diffusion Models

Diffusion models are probabilistic generative models used in computer vision [9], [25]–[27], natural language processing [28], [29], and more recently, RL [7], [8]. Diffusion probabilistic models (DPM) [9] formulate a forward diffusion process by adding a small amount of Gaussian noise to the data samples. By learning the reverse diffusion process, diffusion models learn to generate samples from Gaussian noise. Denoising diffusion probabilistic models (DDPMs) [11] explore DPM's relation to denoising score matching with annealed Langevin dynamics in image synthesis tasks. On the other hand, score-based generative models (SGMs) use a Noise Conditional-Score Network to learn scores at different levels of noise after perturbing the data for training stabilization [30]. Although our approach is developed for the offline RL framework, it can be applied to conditional diffusion models that use classifier guidance. In our case, however, the guidance comes from the states.

III. BACKGROUND

A. Offline Reinforcement Learning

A Markov Decision Process (MDP) tuple is defined by $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \rho_0, \gamma)$ where \mathcal{S} is the state space of state $\mathbf{s} \in \mathcal{S}$, \mathcal{A} is the action space of action $\mathbf{a} \in \mathcal{A}$, $\mathcal{P}(\mathbf{s}'|\mathbf{s}, \mathbf{a}) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ is the conditional probability distribution expressing dynamics, \mathbf{s}' is the next state, r is the reward function, ρ_0 is the initial state distribution and $\gamma \in (0, 1]$ is the discount factor [31]. The objective of an RL agent is to learn a policy $\pi_\theta(\mathbf{a}|\mathbf{s})$, parameterized by θ , that maximizes the expected cumulative discounted reward denoted by $\mathbb{E}_\pi [\sum_t \gamma^t r(\mathbf{s}_t, \mathbf{a}_t)]$. The state-action value function, Q-function, $Q^\pi(\mathbf{s}, \mathbf{a})$ is defined as the expected cumulative discounted reward gained for taking an action \mathbf{a} at a state \mathbf{s} then following π . In offline RL, the agent is tasked with learning the best policy supported by the dataset of MDP tuples denoted by $\mathcal{D} = \{(\mathbf{s}_i, \mathbf{a}_i, \mathbf{s}'_i, r_i)\}$. The dataset

is constructed from the rollouts obtained by a behavior policy $\pi_\beta(\mathbf{a}|\mathbf{s})$ [2].

B. Diffusion Models

Diffusion probabilistic models [9]–[11], commonly called diffusion models for brevity, are a class of probabilistic generative models that seek to generate new samples by learning the underlying probability distribution of the data. The forward diffusion process follows a Markov chain to slowly destroy the structure of an original data sample, x_0 , by adding noise to obtain a sequence of noisy samples $x_1 \dots x_T$. Here, the Gaussian noise added to the data depends on a variance schedule $\{\beta_t \in (0, 1)\}_{t=1}^T$, where β_t is the diffusion rate at timestep t . Since the sequences of noisy samples are available during training, we can train a neural network to predict the noise ϵ_t added to the data at a given timestep. At a high level, we can generate samples from Gaussian noise through an iterative denoising procedure in the reverse diffusion process by using the predicted noise added at each timestep.

In diffusion models, we can directly sample the noisy image at any time t . By replacing the diffusion rate β_t with $\alpha_t = 1 - \beta_t$, we obtain the distribution $q(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_0, (1 - \alpha_t)\mathbf{I})$ using recursion and reparametrization technique where $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$. To estimate the reverse process, we learn the parameters of $p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$. The joint distribution of the reverse diffusion is denoted by $p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^T p_\theta(\mathbf{x}_{t-1} | \mathbf{x}_t)$ where \mathbf{x}_T is the isotropic Gaussian distribution. Ho et al. [11], formulated the simplified loss function in DDPMs for diffusion timestep t

$$L_t = \mathbb{E}_{t, \mathbf{x}_0, \epsilon_t} \left[\left\| \epsilon_t - \epsilon_\theta(\sqrt{\alpha_t}\mathbf{x}_0 + \sqrt{1 - \alpha_t}\epsilon_t, t) \right\|^2 \right]$$

where $\epsilon_\theta(\sqrt{\alpha_t}\mathbf{x}_0 + \sqrt{1 - \alpha_t}\epsilon_t, t)$ is the noise predicted by the neural network and ϵ_t is the true noise used in the forward process.

IV. METHOD

To address the OOD states in Offline RL, we propose incorporating an auxiliary state reconstruction loss in diffusion policies. We first describe the details of Diffusion Q-Learning (Diffusion-QL) [7] and its implementation. Subsequently, we derive a robust diffusion policy that leverages representation learning. Finally, we discuss our approach using a 2D multi-modal contextual bandit environment.

A. Diffusion Q-Learning

Diffusion-QL uses a conditional diffusion model to generate actions conditioned on states [7]. The conditional reverse diffusion chain conditioned on state s is defined by $\pi_\theta(\mathbf{a}_{0:T} | s)$ where T denotes the diffusion timestep. The action obtained by the reverse diffusion process is then used in Q-learning and policy learning in an iterative procedure. Initially, a minibatch of MDP tuples $\{(s, a, r, s')\}$ are sampled from the offline RL dataset. The next action a' is generated by the target diffusion

policy $\pi_{\theta'}$ conditioned on the next state s' . Using double Q-learning trick by [32] and Bellman operator minimization by [33], [34], Diffusion-QL, minimizes

$$\mathbb{E}_{(s, \mathbf{a}, s') \sim \mathcal{D}} \left[\left\| \left(r(s, \mathbf{a}) + \gamma \min_{i=1, 2} Q_{\phi'_i}(s', \mathbf{a}'_0) \right) - Q_{\phi_i}(s, \mathbf{a}) \right\|^2 \right] \quad (1)$$

$$\mathbf{a}'_0 \sim \pi_{\theta'}$$

where \mathcal{D} is the offline RL dataset, $Q_{\phi'_i}$ are the target critic networks and Q_{ϕ_i} are the critic networks. Diffusion policies are optimized by policy improvement using Q-function and behavior cloning loss minimization. We update the critic networks similarly to evaluate the improvement from the state reconstruction loss.

B. State Reconstruction for Diffusion Policies

Diffusion policies learn to generate actions with the guidance of states. Since state information is available in the training and evaluation phase, the diffusion policy can be conditioned on the state to generate actions. The diffusion model learns to predict the noise ϵ_t added to the input at each diffusion iteration t following the simplified loss in DDPM. Diffusion policies extend this idea to the RL framework by concatenating the noisy input (noisy action) vector and the embedded diffusion timestep with the state vector during training. In contrast, using an auxiliary head, SRDP learns to extract the state representation in a shared representation layer. In brief, SRDP operates as follows. The shared SRDP feature extraction module f_ϕ maps the time embedding, state, and noisy action to a latent representation z . This representation is then directed into distinct task-specific heads: the diffusion head f_θ predicts the added noise at that diffusion timestep for the given noisy action and state, and the auxiliary head f_ψ reconstructs the input state. Importantly, the state representation signal is deeply embedded for all diffusion timesteps sampled during training through the state reconstruction loss.

We use a shared fully connected module f_ϕ to map the noisy action $\sqrt{\alpha_t}\mathbf{a} + \sqrt{1 - \alpha_t}\epsilon$, time-embedding and the state s into a shared representation

$$z = f_\phi(\sqrt{\alpha_t}\mathbf{a} + \sqrt{1 - \alpha_t}\epsilon, s, t). \quad (2)$$

The diffusion head f_θ uses this representation to predict the noise added at timestep t , while state head f_ψ learns to reconstruct the state. Thus, diffusion policy loss \mathcal{L}_{DP} is defined by

$$\mathbb{E}_{t \sim U(\{1, \dots, T\})} \left[\left\| \epsilon - f_\theta(f_\phi(\sqrt{\alpha_t}\mathbf{a} + \sqrt{1 - \alpha_t}\epsilon, s, t)) \right\|^2 \right] \quad (3)$$

$$\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

where diffusion timesteps are denoted by t . The diffusion timesteps are sampled from the uniform distribution U over the set $\{1, \dots, T\}$. State reconstruction guidance is also propagated through the network during policy learning. The state reconstruction loss \mathcal{L}_R minimizes

$$\mathbb{E}_{t \sim U(\{1, \dots, T\})} \left[\left\| s - f_\psi(f_\phi(\sqrt{\alpha_t}\mathbf{a} + \sqrt{1 - \alpha_t}\epsilon, s, t)) \right\|^2 \right] \quad (4)$$

$$\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

where f_ψ takes the shared representation as input to predict the state. Behavior cloning loss \mathcal{L}_{BC} in SRDP, jointly learn

the contextual representation of the state and the noise added to the action by minimizing

$$\mathcal{L}_{BC} = \mathcal{L}_{DP} + \lambda \mathcal{L}_R \quad (5)$$

where λ is a hyperparameter that controls the weight of the state reconstruction loss. A key point to note is that the state reconstruction loss in this step is propagated through the network for each randomly selected diffusion timestep.

At each training iteration, we first sample a mini-batch of transitions $\{(s, a, r, s')\}$ from the offline RL dataset. Then, we sample a mini-batch of next actions $\{(a'_0)\}$ from the target diffusion policy using $\{(s')\}$ from the offline RL dataset [7]. The mini-batch of transitions and the sampled $\{(a'_0)\}$ are used to update the critic network using Eq. 1 following the implementation in [7], [35]. After the critic update, we sample $\{(a_0)\}$ from our policy π_θ through an iterative denoising procedure. To guide the action generation procedure to high reward regions, we subtract the scaled [36] expectation of Q-function, $\frac{\eta \mathbb{E}_{s \sim \mathcal{D}, a_0 \sim \pi_\theta} [Q_\phi(s, a_0)]}{\mathbb{E}_{(s, a) \sim \mathcal{D}} \|Q_\phi(s, a)\|}$, from SRDP \mathcal{L}_{BC} in Eq. 5 to obtain \mathcal{L}_{SRDP} . Crucially at this minimization step, the state reconstruction loss is propagated through the network for each diffusion timestep, promoting learning more descriptive features from the OOD-state samples. Then, we update the target policy network and the target critic networks.

In the evaluation phase, we sample actions from our policy through an iterative denoising procedure. First, we sample $a_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ at diffusion timestep T. Then, we concatenate this sampled action a_T with our current state from the environment and the embedding of the diffusion timestep to obtain the input vector for our diffusion policy. Subsequently, our diffusion model predicts the noise $\epsilon_\theta = f_\theta(f_\phi(a_t, s, t))$ added to the action at that diffusion timestep given the state while the reconstructed states from f_ψ are not used. To obtain the action a_0 generated by our model, we run the reverse diffusion chain by computing $a_{t-1} | a_t = \frac{1}{\sqrt{\alpha_t}} \left(a_t - \frac{1-\alpha_t}{\sqrt{1-\alpha_t}} f_\theta(f_\phi(a_t, s, t)) \right) + \sqrt{\beta_t} \epsilon$. As suggested in [11] to enhance the sampling quality, we sample $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ for $t = T, \dots, 2$ and assign $\epsilon = 0$ for $t = 1$.

C. Out-of-Distribution States in Offline RL

We aim to learn a robust policy from offline RL datasets consisting of expert and novice demonstrations with different modalities. Therefore, we designed an environment named 2D multimodal contextual bandit to evaluate the agent's performance in OOD states. Previous work [7] used a 2D-bandit environment to illustrate the advantage of using a highly expressive model to represent policies. Here, we extend this environment to a contextual bandit setting to assess the trained policy in OOD states. The primary objective in this task is to learn multimodal expert behavior, particularly in challenging OOD states. Thus, Double Q-learning [32] is not used in this context to isolate the regularization induced by SRDP behavior cloning loss.

We consider a two-dimensional continuous state and action space where the states and actions are characterized as real-valued x- and y-coordinates. To illustrate multimodal expert behaviors, the states in each pair of quadrants map to actions generated from a mixture of two Gaussian distributions.

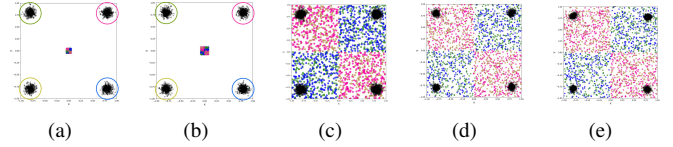


Fig. 1. The states mapped to actions (represented by black dots) in the first, second, third, and fourth quadrants are colored magenta, green, light brown, and blue, respectively. The training dataset (a) is constructed with $s_{train} \sim \mathcal{U}([-0.05, 0.05] \times [-0.05, 0.05])$ and used for experiments illustrated in Fig. 2. The training dataset in (b) is constructed from $s_{train} \sim \mathcal{U}([-0.08, 0.08] \times [-0.08, 0.08])$ and used for experiments illustrated in Fig. 3 and Fig. 7. State samples for ground truth (c) are generated from the uniform distribution $\mathcal{U}([-1.0, 1.0] \times [-1.0, 1.0])$. Actions are generated from the GMM detailed in Eq. 6. Results for training with non-OOD data generated from the uniform distribution $\mathcal{U}([-1.0, 1.0] \times [-1.0, 1.0])$ are illustrated for BC-Diffusion (d) and SRDP(ours) (e). Chamfer distance between the ground truth contextual action distribution and the actions generated using the non-OOD dataset for BC-Diffusion is 0.047, and SRDP(ours) is **0.037**.

Given a state (x, y) , the agent needs to generate an action $[a_1(x, y), a_2(x, y)]$ that has a high probability in the matching mixture of two Gaussians. Subsequently, we construct a training dataset by sampling states from two uniform distributions and actions from two Gaussian mixture models (GMM), where each Gaussian mixture comprises two Gaussian distributions. For a given state s , an action a is sampled from a GMM with two Gaussians as $a \sim \pi \mathcal{N}(\mu_1(s), \Sigma) + \pi \mathcal{N}(\mu_2(s), \Sigma)$ where Σ is the diagonal covariance matrix with $\sigma = [0.05, 0.05]$, π is the mixture weight 0.5, μ_1 , and μ_2 are the mean vectors of Gaussian distributions. We use the following procedure to determine the values of the mean vectors $\mu_1(s)$, and $\mu_2(s)$

$$\mu_1(s), \mu_2(s) = \begin{cases} (-0.8, -0.8), (0.8, 0.8) & \text{if } s \in [-1.0, 0.0] \times [0.0, 1.0] \\ (-0.8, -0.8), (0.8, 0.8) & \text{if } s \in [0.0, 1.0] \times [-1.0, 0.0] \\ (-0.8, 0.8), (0.8, -0.8) & \text{if } s \in [-1.0, 0.0] \times [-1.0, 0.0] \\ (-0.8, 0.8), (0.8, -0.8) & \text{if } s \in [0.0, 1.0] \times [0.0, 1.0] \end{cases} \quad (6)$$

where a state (s) is sampled from the uniform distributions ($s_{train} \sim \mathcal{U}([-0.05, 0.05] \times [-0.05, 0.05])$) and ($s_{train} \sim \mathcal{U}([-0.08, 0.08] \times [-0.08, 0.08])$) when constructing the training data. During testing, we sample states from subregions within the Euclidean plane larger than those in the training dataset. In particular, we sample states from the uniform distribution $\mathcal{U}([-1.0, 1.0] \times [-1.0, 1.0])$ to evaluate the policies in OOD states.

V. EXPERIMENTS

This section empirically evaluates our proposed approach, i.e., state reconstruction guidance on the 2D-Multimodal Contextual Bandit environment, multimodal real-robot setup, sparse reward continuous control maze navigation dataset with missing data generated in [37] for offline RL pretraining, and D4RL benchmarks from [1].

A. 2D-Multimodal Contextual Bandit Experiments

The training dataset in 2D-Multimodal Contextual Bandit $D = \{(s_i, a_i)\}_{i=1}^n$, consists of $n = 10000$ state-action tuples with horizon $H = 1$. To examine the policy in OOD states, the states generated for training are sampled from the uniform

distribution $s_{train} \sim \mathcal{U}([-0.05, 0.05] \times [-0.05, 0.05])$ in Fig. 1(a), and $s_{train} \sim \mathcal{U}([-0.08, 0.08] \times [-0.08, 0.08])$ in Fig. 1(b). The states used for the evaluation are sampled from the uniform distribution of $s_{test} \sim \mathcal{U}([-1, 1] \times [-1, 1])$. Ground truth for OOD state generalization corresponding to the training datasets in Fig. 1(a) and (b) is illustrated in Fig. 1(c) where the black points show the actions in the dataset following Eq. 6. Similarly, the black points in Fig. 2(a) and (b) show the actions generated by the SRDP and BC-Diffusion policies, respectively. If a state generates an action in the first, second, third, and fourth quadrants, the state is colored magenta, green, light brown, and blue, respectively.

Visually distinguishable quadrants with respective coloring in Fig. 2(a) (SRDP) indicate that the reverse diffusion process can generate actions accurately for OOD states. In contrast, Fig. 2(b) shows that BC-Diffusion memorizes actions without using state information. Subsequently, results in Fig. 2 show that the representation learning with state reconstruction loss promotes finding expert skills in multimodal data, achieving faster convergence. Although a larger portion of the state space is not included in the training set distribution in Fig. 2, SRDP can generalize well to OOD states and learn at least one of the two expert behaviors. Conversely, BC-Diffusion finds the action distribution of the dataset, yet it cannot assign the correct action distribution for the given state. Accurate partitioning of the state space, in terms of the effect of the actions taken, is particularly important in real-world robotics tasks where naive memorization of the action distribution in the data without state dependence can have severe consequences. Although SRDP can represent a single mode in this challenging OOD task where only 0.25% of the state space is present in the data, it can correctly learn an expert policy. All hidden layers have the same size of 16 for SRDP and BC-Diffusion.

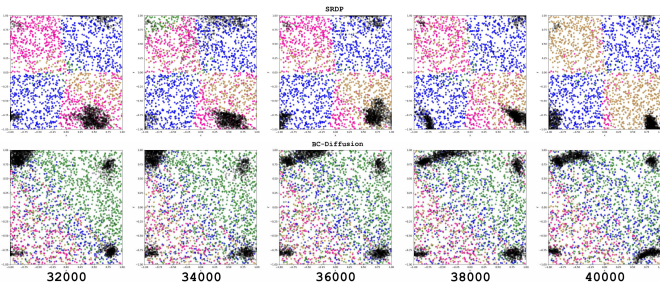


Fig. 2. Top to bottom: SRDP ($\lambda = 1.25$) and BC-Diffusion [7]. Dataset for training is constructed from Fig. 1 (a) ($s_{train} \sim \mathcal{U}([-0.05, 0.05] \times [-0.05, 0.05])$) and ground truth is illustrated in Fig. 1 (c). The number of training iterations in each column increases incrementally from left to right by 2000, starting from 32000, as the convergence for the more restricted dataset takes longer.

In the next set of experiments, the training dataset includes $s_{train} \sim \mathcal{U}([-0.08, 0.08] \times [-0.08, 0.08])$ and a_{train} generated by Eq. 6. Fig. 3 and Fig. 7 show that only SRDP can represent multimodal expert distributions and partition the state space into visible quadrants at earlier iterations, improving training stability and reducing computation costs. Consistent with the experiments in Fig. 2, BC-Diffusion only learns the action distributions in the dataset, excluding the state information in OOD states. As an ablation study, we provide illustrations with

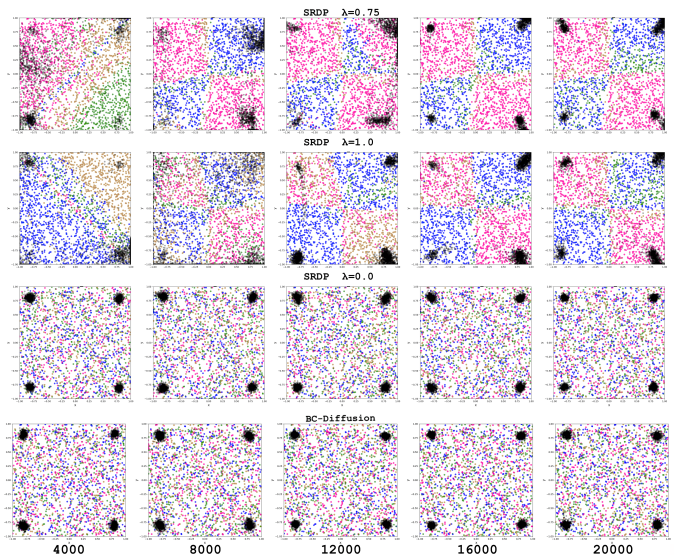


Fig. 3. Top to bottom rows show the results from SRDP (proposed model) with the scaling parameter $\lambda = 0.75$, $\lambda = 1.0$, $\lambda = 0.0$ (meaning no state reconstruction loss) and BC-Diffusion [7]. The training dataset is constructed from $s_{train} \sim \mathcal{U}([-0.08, 0.08] \times [-0.08, 0.08])$. The number of training iterations increases incrementally by 4000 from left to right.

TABLE I
ABLATION STUDY FOR SRDP SCALING PARAMETERS OVER TRAINING ITERATIONS AT 4000 INTERVALS WITH Chamfer Distance.

Method	4000	8000	12000	16000	20000
SRDP($\lambda = 0.0$)	1.62 ± 0.31	1.51 ± 0.06	1.63 ± 0.26	1.61 ± 0.34	1.63 ± 0.32
SRDP($\lambda = 0.25$)	1.49 ± 0.25	1.03 ± 0.59	0.91 ± 0.62	0.82 ± 0.64	0.81 ± 0.67
SRDP($\lambda = 0.5$)	1.1 ± 0.3	1.07 ± 0.55	1.01 ± 0.72	0.91 ± 0.68	0.79 ± 0.55
SRDP($\lambda = 0.75$)	1.62 ± 0.24	0.88 ± 0.51	0.83 ± 0.43	0.54 ± 0.28	0.39 ± 0.16
SRDP($\lambda = 1.0$)	1.24 ± 0.48	0.95 ± 0.35	0.68 ± 0.55	0.47 ± 0.43	0.44 ± 0.41
BC-Diffusion	1.41 ± 0.01	1.51 ± 0.16	1.54 ± 0.18	1.6 ± 0.14	1.62 ± 0.15

different scaling parameters that control the weight of state reconstruction loss in Fig. 3. We use a dual head architecture with hidden layer sizes ($16^{shared}, 16^{shared}, 16$) for SRDP and three hidden layers with size 16 for BC-Diffusion. Notice that SRDP with scaling parameter 0 performs similarly to the BC-Diffusion baseline in Fig. 3 as this reduces SRDP to BC-Diffusion. We compute Chamfer distances between the ground truth actions and the actions generated by SRDP with varying scaling parameters (λ) and BC-Diffusion for each group of states. Table I, shows the sum of Chamfer distances over each group of states across five random seeds, at 4000 intervals. Chamfer distance is particularly suitable for our case because it measures the distance between two sets of points. BC-Diffusion is prone to overfitting since the chamfer distance increases as the training progresses. The results show that SRDP performs significantly better and converges faster than BC-Diffusion with appropriate scaling parameter selection.

B. Multimodal UR10 Experiments

We design a multimodal UR10 robot experiment to illustrate the application of our method in the real world and highlight the importance of OOD generalization. Our hardware setup consists of a 6-Dof UR10 manipulator robot mounted with a Robotiq gripper illustrated in Fig. 4. Similar to the experiments in [4] designed for a visuomotor policy learning task, we define the actions for our policy as the space positional commands

of the end-effector. In the setting visualized in Fig. 4, the robot learns to reach the vegetables or the coffee maker from a random initial state. If the gripper is opened in states in region A (red/magenta) or D (yellow/light brown) (enabling the use of the pink coffee stirrer), it should reach the coffee maker for stirring. Conversely, if the gripper is enclosed (allowing the use of the green knife) in region B (blue) or C (green), it should reach the vegetables for chopping. The training dataset is constructed from the true space positional commands of the end-effector, $s_{train} \sim \mathcal{U}([-1.0, -0.9] \times [0.03, 0.17])$. We sample states from the uniform distribution $\mathcal{U}([-1.2, -0.7] \times [-0.25, 0.45])$ to evaluate the policies in OOD states. The action distributions are generated with the same procedure outlined in Eq. 6 though the origin and GMM distributions are shifted for real UR10 space. Specifically, for states with respect to the center position at $(-0.95, 0.1)$, action distributions for the right upper, left upper, left bottom, and right bottom regions are produced with corresponding mean $\mu_1(s) = (-0.8, 0.35)$, $\mu_2(s) = (-1.1, 0.35)$, $\mu_3(s) = (-1.1, -0.15)$, $\mu_4(s) = (-0.8, -0.15)$ and diagonal covariance matrix Σ with $\sigma = 0.015$. We evaluate SRDP and baselines trained with 75 diffusion timesteps across five random seeds. Chamfer distance results of SRDP($\lambda = 1.0$) compared to the baseline with respect to the ground truth contextual action distribution in the dataset is (0.071 ± 0.02) . For Diffusion-QL and SRDP($\lambda = 0.0$) the chamfer distances are (0.27 ± 0.01) , (0.28 ± 0.02) respectively. We use a dual head architecture with hidden layer sizes $(32^{shared}, 16^{shared}, 32)$ for SRDP($\lambda = 0.0$) and SRDP($\lambda = 1.0$) and $(16, 16, 16)$ for BC-Diffusion. SRDP($\lambda = 0.0$) is BC-Diffusion with a bottleneck layer in the middle without a state reconstruction loss. The action generation time for SRDP across 10 trials is 8.34 ± 0.28 ms on an NVIDIA 4090 GPU. Results indicated that the architecture change did not imply a better OOD generalization; hence state reconstruction signal is essential in learning multimodal expert behavior from OOD data.

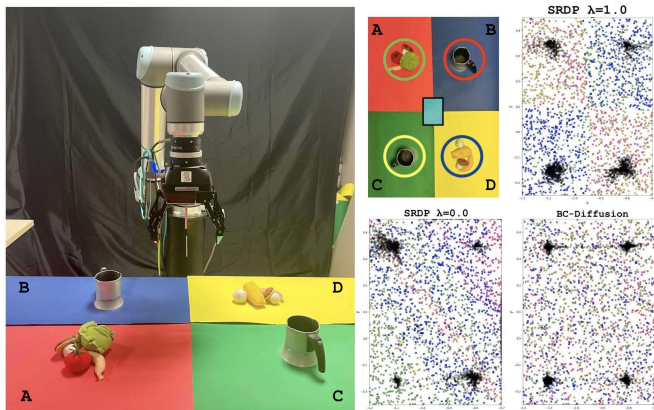


Fig. 4. Real robot setup and the comparison of SRDP($\lambda = 1.0$) with BC-Diffusion and SRDP($\lambda = 0.0$). State samples in the training dataset are within the light blue region. The color-coded regions are labeled as follows: A (red/magenta), B (blue), C (green), and D (yellow/light brown).

C. Missing Data Maze Environment

Maze2d navigation environments were introduced as benchmarks for offline RL in D4RL [1]. In this environment, a

TABLE II
COMPARISON OF SRDP WITH DIFFUSION-QL BASELINE.

maze2d-missing-data-large-v1	Normalized Score
SRDP($\lambda = 1.0$)	35.0 ± 28.2
SRDP($\lambda = 0.75$)	23.9 ± 20.4
SRDP($\lambda = 0.25$)	20.0 ± 37.9
SRDP($\lambda = 0.0$)	1.7 ± 2.3
Diffusion - QL	13.1 ± 15.3
maze2d-large-v1	Normalized Score
SRDP($\lambda = 0.75$)	203.6 ± 19.7
Diffusion - QL	189.1 ± 15.3

2D force-actuated ball robot must navigate through a closed maze to reach a fixed goal location during evaluation. The maze2d dataset consists of trajectories collected by a planner agent, which uses a PD controller to reach a random goal from a random initial state location. The actions are defined as the linear force applied in x-y directions, whereas the observations are defined as the concatenation of the position and the linear velocity of the ball in the x-y direction. To assess the performance of SRDP in OOD states, we use the sparse reward maze2D offline RL dataset “maze2d-missing-data-large-v1” by [37] illustrated in Fig. 5(a) where the data collected in the vicinity of three circles, one encapsulating the goal, is missing. This task is used to evaluate the performance of online and offline finetuning after offline pretraining in [37]. In our context, we focused on results without online finetuning, aligning with the more challenging experimental framework adopted throughout this study. Notably, the agent starts from a random initial state during evaluation, and a significant portion of state data is absent near the goal and random initial state locations. An ablation study on this environment demonstrating the advantage of SRDP in Table II shows the mean and standard deviation of normalized scores obtained across five random seeds. More specifically, the results indicate the superiority of SRDP($\lambda = 1.0$) compared to Diffusion-QL [7] with $(256, 256, 256)$. It is important to note that SRDP with a dual head architecture with hidden layer sizes $(512^{shared}, 256^{shared}, 512)$ and scaling parameter $\lambda = 0.0$ becomes Diffusion-QL without a state reconstruction decoder. In addition, SRDP performs superior to the baseline for “maze2d-large-v1” sparse environment in D4RL without missing data.

D. D4RL Benchmark

D4RL benchmark [38], comprising extensive robotics datasets, has been widely used as a benchmark for offline RL algorithms. In AntMaze environments, the objective of an 8-DoF quadruped ant robot is to navigate to a 2D goal location in various mazes. Correspondingly, in Gym-MuJoCo ([38], [39]) half-cheetah, hopper, and walker2d environments, the objective is to achieve forward locomotion. Since offline RL differs from imitation learning, each dataset in Table III includes various amounts of suboptimal data. For these experiments, we follow the details of the online implementation used in Diffusion-QL, where they assume that the policies can be evaluated at fixed intervals with few online interactions. This is a form of early stopping in RL introduced in [40] where

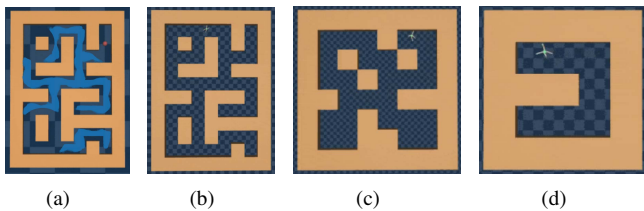


Fig. 5. (a) Maze2d environment with missing data “maze2d-missing-data-large-v1” from [37]. D4RL AntMaze Environments with three layouts: (b) Antmaze-large-v0, (c) Antmaze-medium-v0, (d) Antmaze-umaze-v0.

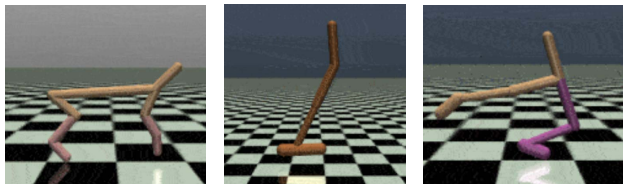


Fig. 6. D4RL Gym-MuJoCo Environments

TABLE III
COMPARISON OF SRDP WITH THE DIFFUSION-QL BASELINE.

AntMaze	Diffusion-QL	SRDP(ours)
antmaze-umaze-v0	96.0	96.4 ± 1.5
antmaze-umaze-diverse-v0	84.0	89.8 ± 5.1
antmaze-medium-play-v0	79.8	78.4 ± 8.1
antmaze-medium-diverse-v0	82.0	90.6 ± 6.0
antmaze-large-play-v0	49.0	63.0 ± 8.0
antmaze-large-diverse-v0	61.7	62.6 ± 3.5
Average	75.4	80.1
Gym	Diffusion-QL	SRDP(ours)
halfcheetah-medium-v2	51.5	51.9 ± 1.5
hopper-medium-v2	96.6	96.8 ± 3.1
walker2d-medium-v2	87.3	88.1 ± 0.5
halfcheetah-medium-replay-v2	48.3	48.1 ± 0.3
hopper-medium-replay-v2	102.0	102.1 ± 0.1
walker2d-medium-replay-v2	98.0	98.1 ± 1.2
halfcheetah-medium-expert-v2	97.2	97.5 ± 0.1
hopper-medium-expert-v2	112.3	112.6 ± 0.4
walker2d-medium-expert-v2	111.2	111.1 ± 0.4
Average	89.3	89.6

the hyperparameter, the number of training epochs, is tuned for Diffusion-QL and the proposed model, SRDP. We report the average normalized scores of undiscounted returns for SRDP and Diffusion-QL in Table III, where 100 corresponds to expert-level behavior compared to the normalized scores for Diffusion-QL from [7].

Antmaze datasets are generated following non-Markovian and suboptimal policies, sparse rewards, and multitask data design procedures [1]. Fig. 5 (b), (c), (d) shows the AntMaze environments where each maze layout, large, medium, umaze, has different levels, such as play and diverse. The performance scores across five random seeds in Table III demonstrate that the proposed SRDP model is superior for all cases except “antmaze-medium-play-v0”. OOD generalization is beneficial in multitask settings; hence, SRDP can enhance performance significantly.

Gym-MuJoCo environments, comprising continuous control tasks in MuJoCo [39], have been commonly used in deep RL

[1], [36], [41]–[43]. In D4RL, datasets are collected using the online RL interaction data of a Soft Actor-Critic (SAC) agent [1], [42]. To evaluate an offline-RL algorithm in narrow data distribution and suboptimal behavior policy settings, “medium”, “medium-replay”, and “medium-expert” datasets are generated. Medium datasets are generated by collecting the rollouts from a medium-performing policy, whereas medium-replay datasets are generated by keeping a replay buffer of rollouts until the RL policy reaches a medium-level performance. Medium-expert datasets include expert trajectories by 50% in addition to medium-level rollouts. Results presented in Table III show that our proposed model, SRDP, performs better than Diffusion-QL [7] across five random seeds.

VI. CONCLUSION

In this work, we propose SRDP for OOD generalization in offline RL, a representation learning-based method built on top of the recent class of diffusion policies introduced by [7]. To show the multiple skills learned by the model when evaluated in OOD states, we designed a 2D Multimodal Contextual Bandit environment and applied it to a real robotic scenario. Compared to prior work, our results indicate that SRDP can represent multimodal policies, partition the state space more accurately, and converge faster in real-robot and simulation environments. In addition, results show that SRDP can learn superior models compared to previous work in Antmaze and Gym-MuJoCo environments in D4RL benchmarks [1] with various levels and agents. Finally, on a sparse continuous control navigation task where critical regions of the state space are completely removed from the offline RL dataset, our method performs significantly better than the standard diffusion policy-based RL. For future work, we plan to integrate a vision module into SRDP, reduce the inference time in diffusion policies, and extend our approach to trajectory-level diffusion probabilistic models [8].

APPENDIX

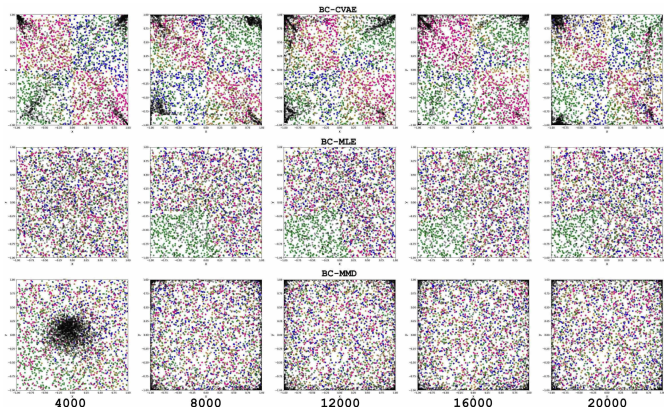


Fig. 7. Additional results for Section V-A. Top to bottom Row: BC-CVAE represents the policy by a conditional variational autoencoder (CVAE) [33], BC-MLE [36] uses a Multivariate Gaussian to model the policy with a diagonal covariance matrix, BC-MMD [14] uses a CVAE that learns the behavior policy to constrain a Gaussian policy. The training dataset is constructed from $s_{train} \sim \mathcal{U}([-0.08, 0.08] \times [-0.08, 0.08])$. The number of training iterations increases incrementally by 4000 from left to right.

ACKNOWLEDGMENT

This research was supported by Grant-in-Aid for Scientific Research – the project with number 22H03670, the project JPNP16007 commissioned by the New Energy and Industrial Technology Development Organization (NEDO), the Scientific and Technological Research Council of Turkey (TUBITAK, 118E923), and INVERSE project (101136067) funded by the European Union. The numerical calculations reported in this paper were partially performed at TUBITAK ULAKBIM, High Performance and Grid Computing Center (TRUBA resources). The authors would like to thank the anonymous reviewers and Editor Aleksandra Faust for their valuable feedback and constructive suggestions.

REFERENCES

- [1] 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.
- [2] S. Levine, A. Kumar, G. Tucker, and J. Fu, “Offline reinforcement learning: Tutorial, review, and perspectives on open problems,” *CoRR*, vol. abs/2005.01643, 2020.
- [3] J. Li, C. Tang, M. Tomizuka, and W. Zhan, “Dealing with the unknown: Pessimistic offline reinforcement learning,” in *5th Annual Conference on Robot Learning*, 2021.
- [4] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, “Diffusion policy: Visuomotor policy learning via action diffusion,” in *Robotics: Science and Systems (RSS)*, 2023.
- [5] 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 *5th Annual Conference on Robot Learning*, 2021.
- [6] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, “Learning fine-grained bimanual manipulation with low-cost hardware,” *arXiv preprint arXiv:2304.13705*, 2023.
- [7] Z. Wang, J. J. Hunt, and M. Zhou, “Diffusion policies as an expressive policy class for offline reinforcement learning,” in *The Eleventh International Conference on Learning Representations*, 2023.
- [8] M. Janner, Y. Du, J. B. Tenenbaum, and S. Levine, “Planning with diffusion for flexible behavior synthesis,” in *International Conference on Machine Learning*, 2022.
- [9] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, “Deep unsupervised learning using nonequilibrium thermodynamics,” *CoRR*, vol. abs/1503.03585, 2015.
- [10] Y. Song and S. Ermon, “Generative modeling by estimating gradients of the data distribution,” in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019.
- [11] J. Ho, A. Jain, and P. Abbeel, “Denosing diffusion probabilistic models,” in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 6840–6851.
- [12] T. Denouden, R. Salay, K. Czarnecki, V. Abdelzad, B. Phan, and S. Vernekar, “Improving reconstruction autoencoder out-of-distribution detection with mahalanobis distance,” *arXiv preprint arXiv:1812.02765*, 2018.
- [13] U. Mutlu and E. Alpaydin, “Training bidirectional generative adversarial networks with hints,” *Pattern Recognition*, vol. 103, p. 107320, 2020.
- [14] A. Kumar, J. Fu, G. Tucker, and S. Levine, “Stabilizing off-policy q-learning via bootstrapping error reduction,” *CoRR*, vol. abs/1906.00949, 2019.
- [15] Y. Wu, G. Tucker, and O. Nachum, “Behavior regularized offline reinforcement learning,” *CoRR*, vol. abs/1911.11361, 2019.
- [16] A. Nair, M. Dalal, A. Gupta, and S. Levine, “Accelerating online reinforcement learning with offline datasets,” *CoRR*, vol. abs/2006.09359, 2020.
- [17] I. Kostrikov, A. Nair, and S. Levine, “Offline reinforcement learning with implicit q-learning,” *CoRR*, vol. abs/2110.06169, 2021.
- [18] A. Kumar, A. Zhou, G. Tucker, and S. Levine, “Conservative q-learning for offline reinforcement learning,” *CoRR*, vol. abs/2006.04779, 2020.
- [19] S. Zhang, B. Liu, and S. Whiteson, “GradientDICE: Rethinking generalized offline estimation of stationary values,” in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, H. D. III and A. Singh, Eds., vol. 119. PMLR, 13–18 Jul 2020, pp. 11 194–11 203.
- [20] N. Jiang and L. Li, “Doubly robust off-policy evaluation for reinforcement learning,” *CoRR*, vol. abs/1511.03722, 2015.
- [21] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Zou, S. Levine, C. Finn, and T. Ma, “MOPO: model-based offline policy optimization,” *CoRR*, vol. abs/2005.13239, 2020.
- [22] T. Yu, A. Kumar, R. Rafailov, A. Rajeswaran, S. Levine, and C. Finn, “COMBO: conservative offline model-based policy optimization,” *CoRR*, vol. abs/2102.08363, 2021.
- [23] M. Janner, Q. Li, and S. Levine, “Reinforcement learning as one big sequence modeling problem,” *CoRR*, vol. abs/2106.02039, 2021.
- [24] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch, “Decision transformer: Reinforcement learning via sequence modeling,” *CoRR*, vol. abs/2106.01345, 2021.
- [25] J. Yue, L. Fang, S. Xia, Y. Deng, and J. Ma, “Dif-fusion: Towards high color fidelity in infrared and visible image fusion with diffusion models,” *CoRR*, vol. abs/2301.08072, 2023.
- [26] M. Wei, Y. Shen, Y. Wang, H. Xie, and F. L. Wang, “Raindiffusion: When unsupervised learning meets diffusion models for real-world image deraining,” *CoRR*, vol. abs/2301.09430, 2023.
- [27] D. Eschweiler and J. Stegmaier, “Denosing diffusion probabilistic models for generation of realistic fully-annotated microscopy image data sets,” *CoRR*, vol. abs/2301.10227, 2023.
- [28] F. Bao, C. Li, Y. Cao, and J. Zhu, “All are worth words: a vit backbone for score-based diffusion models,” *CoRR*, vol. abs/2209.12152, 2022.
- [29] J. C. White and R. Cotterell, “Schrödinger’s bat: Diffusion models sometimes generate polysemous words in superposition,” *CoRR*, vol. abs/2211.13095, 2022.
- [30] L. Yang, Z. Zhang, Y. Song, S. Hong, R. Xu, Y. Zhao, Y. Shao, W. Zhang, B. Cui, and M.-H. Yang, “Diffusion models: A comprehensive survey of methods and applications,” 2022.
- [31] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT Press, 2018.
- [32] H. Van Hasselt, A. Guez, and D. Silver, “Deep reinforcement learning with double q-learning,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, no. 1, 2016.
- [33] S. Fujimoto, D. Meger, and D. Precup, “Off-policy deep reinforcement learning without exploration,” in *International Conference on Machine Learning*, 2018.
- [34] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016.
- [35] Z. Wang, “Diffusion-policies-for-offline-rl,” 2022, <https://github.com/Zhendong-Wang/Diffusion-Policies-for-Offline-RL>.
- [36] S. Fujimoto and S. Gu, “A minimalist approach to offline reinforcement learning,” in *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021.
- [37] M. S. Mark, A. Sharma, F. Tajwar, R. Rafailov, S. Levine, and C. Finn, “Offline retraining for online rl: Decoupled policy learning to mitigate exploration bias,” 2023.
- [38] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, “Openai gym,” *arXiv preprint arXiv:1606.01540*, 2016.
- [39] E. Todorov, T. Erez, and Y. Tassa, “Mujoco: A physics engine for model-based control,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.
- [40] S. E. Ada, E. Ugur, and H. L. Akin, “Generalization in transfer learning: robust control of robot locomotion,” *Robotica*, vol. 40, no. 11, p. 3811–3836, 2022.
- [41] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch, “Decision transformer: Reinforcement learning via sequence modeling,” *arXiv preprint arXiv:2106.01345*, 2021.
- [42] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor,” *arXiv preprint arXiv:1801.01290*, 2018.
- [43] D. Brandfonbrener, W. F. Whitney, R. Ranganath, and J. Bruna, “Offline RL without off-policy evaluation,” in *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021.