Sample Efficient Offline RL via T-symmetry Enforced Latent State-Stitching

Anonymous Author(s)
Affiliation
Address
email

Abstract

Offline reinforcement learning (RL) has achieved notable progress in recent years. However, most existing offline RL methods require a large amount of training data to achieve reasonable performance and offer limited generalizability in out-of-distribution (OOD) regions due to conservative data-related regularizations. This seriously hinders the usability of offline RL in solving many real-world applications, where the available data are often limited. In this study, we introduce TELS, a highly sample-efficient offline RL algorithm that enables state-stitching in a compact latent space regulated by the fundamental time-reversal symmetry (T-symmetry) of dynamical systems. Specifically, we introduce a T-symmetry enforced inverse dynamics model (TS-IDM) to derive well-regulated latent state representations that greatly facilitate OOD generalization. A guide-policy can then be learned entirely in the latent space to optimize for the reward-maximizing next state, bypassing the conservative action-level behavioral regularization adopted in most offline RL methods. Finally, the optimized action can be extracted using the learned TS-IDM, together with the optimized latent next state from the guide-policy. We conducted comprehensive experiments on both the D4RL benchmark tasks and a real-world industrial control test environment, TELS achieves superior sample efficiency and OOD generalization performance, significantly outperforming existing offline RL methods in a wide range of challenging small-sample tasks.

1 Introduction

Offline reinforcement learning (RL) has seen rapid progress in recent years. It bypasses the reliance on environment interactions of online RL, directly utilizing pre-collected datasets for policy learning, thus being ideal for many real-world tasks that lack high-fidelity simulators or have environment interaction restrictions [1, 2, 3]. However, offline RL is also known to be prone to value overestimation, caused by extrapolation error when evaluating out-of-distribution (OOD) samples and amplified through the bootstrapped update procedure in RL [4, 5].

In the past few years, quite a few offline RL methods have been proposed, which commonly adopt the pessimism principle using strategies such as adding explicit or implicit policy constraints to prevent the selection of OOD actions [4, 5, 6, 7], penalizing value function on unseen samples [8, 9, 10, 11], or adopting in-sample learning to implicit regularize policy optimization [12, 13, 14]. What's in common with these methods is the use of some kind of action-level constraints to avoid exploitation on OOD actions. Although this could stabilize offline value and policy learning, it inevitably leads to over-conservatism and crippled OOD generalization performance [15, 16]. Most of the existing offline RL methods only perform well when trained with sufficiently large amounts of offline data and reasonable state-action space coverage (e.g., 1 million samples for simple D4RL benchmark tasks [17]). This forms a stark contrast to the reality in most real-world scenarios, such as industrial

control [2, 3], robotics [18], and healthcare [19], where the real-world operational data are often scarce, and scaling up data collection can be rather costly.

Enhancing sample efficiency and OOD generalization capability is essential to making offline RL widely applicable to real-world applications. This is particularly important for small dataset settings, as most of the state-action space will become OOD regions. Several recent attempts have been made to improve the generalization performance of offline RL, which mainly follow three directions. The first direction builds upon the empirical observation that deep value functions interpolate well but struggle to extrapolate, thus allowing exploitation on interpolated OOD actions to promote generalization [15]. However, this method has a smoothness assumption on the offline dataset geometry and only applies to continuous action space. The second class of methods avoids the conservative action-level constraint and instead performs reward maximization on the state-space [20, 21], which allows exploitation of OOD actions as long as the corresponding state transitions are reachable (also referred to as "*state-stitching*" [20]). Although such methods offer some promising generalization capabilities, they still require the state-action space to have reasonable data coverage to enable valid state-stitching. The last and also the most explored direction is to learn compact and robust latent representations to enhance sample efficiency [22, 23, 24, 25, 16]. Most of these methods only focus on extracting statistical-level information from the data, using techniques such as contrastive learning [22, 23, 24, 26]. Due to the lack of in-depth modeling of the underlying dynamics inside the sequential data, these methods still struggle to provide generalizable information beyond data distribution. Some recent methods [25, 16, 3] propose to extract fundamental symmetries of dynamics to facilitate policy learning, such as the time-reversal symmetry (T-symmetry) [16, 3], i.e., the underlying physical laws should not change under the time-reversal transformation. By leveraging such universally held symmetries in the dataset, it is possible to maximally promote OOD generalization without being restrained by data distribution-related information. Although promising, these methods are built upon offline RL backbone algorithms with action-level constraints (e.g., CQL [8] or TD3+BC [7]), which still suffer from the over-conservatism issue.

In this paper, we find that enabling state-stitching in a coherent, fundamental symmetry-enforced latent space can lead to a surprisingly strong sample-efficient offline RL algorithm. We refer to our method as Offline RL via **T**-symmetry **E**nforced **L**atent **S**tate-Stitching (TELS). Specifically, we introduce a T-symmetry enforced inverse dynamics model (TS-IDM) that can not only learn well-behaved and OOD generalizable latent representations, but also facilitate effective action inference. Within the learned latent state space, we can optimize a T-symmetry regularized guide-policy to output the next latent state that maximizes the accumulated reward, bypassing the conservative action-level behavioral regularization as adopted in most offline RL algorithms. Lastly, the optimized action can be easily extracted by plugging the output of the guide-policy as the goal state in the learned TS-IDM. We evaluate TELS on both the challenging reduced-size D4RL benchmark tasks and a real-world industrial control test environment [3]. Through comprehensive experiments, we show that TELS achieves state-of-the-art (SOTA) sample efficiency and OOD generalization capability, significantly outperforming existing offline RL algorithms on small datasets. Our method greatly pushes the performance limit of offline RL under low data regimes, offering a new opportunity to tackle many previously unsolvable tasks with data size restrictions.

2 Preliminaries

Offline RL. We consider the standard Markov decision process (MDP) setting [27], which is represented as a tuple $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, r, \mathcal{P}, \rho, \gamma\}$, and a dataset $\mathcal{D}$, which consists of trajectories $\tau = \{s_0, a_0, s_1, a_1, ..., s_T, a_T\}$. Here $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, $r(s, a)$ is a scalar reward function, $\mathcal{P}(s'|s, a)$ and ρ denote the transition dynamics and initial state distribution respectively, and $\gamma \in (0, 1)$ is a discount factor. Our goal is to learn a policy $\pi(a|s)$ based on dataset $\mathcal{D}$ by maximizing the expected return in the MDP: $\mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t \cdot r(s_t, a_t)]$.

Offline policy optimization in the state space. Instead of adopting conservative action-level constraints for offline policy learning, Policy-guided Offline RL (POR) [20] proposes an alternative scheme, which decomposes the conventional reward-maximizing policy into a guide-policy and an execute policy. The guide-policy only works in the state space to find the optimal next state that maximizes the state-value function, and the execute-policy is learned as an inverse dynamics model [20] or a goal-conditioned imitative policy [21]. Such methods only need to learn a state-only value function V using the IQL-style expectile regression [12], or the sparse value learning objective

as discussed in [13]. We present the former as follows:

$$V = \arg\min_{V} \mathbb{E}_{(s,r,s')\sim\mathcal{D}} \left[L_2^\tau \left(r(s) + \gamma \bar{V}(s') - V(s) \right) \right] \tag{1}$$

where $L_2^\tau(x) = |\tau - \mathbb{1}(x < 0)|x^2$ is the asymmetric expectile regression loss and $\bar{V}$ denotes the target value network. Based on the learned state-value function, we can learn a guide-policy $\pi_g(s'|s)$ to serve as a prophet by telling which state the agent should (high reward) and can (logical generalization) go to, without being constrained to state-action transitions seen in the dataset. This can be achieved by leveraging an advantage weighted regression (AWR) objective [28, 29] to maximize the value while implicitly constraining π_g to $s \to s'$ transitions observed in the dataset (i.e., *state-stitching*):

$$\pi_g = \arg\max_{\pi_g} \mathbb{E}_{(s,r,s')\sim\mathcal{D}} \Big[\exp(\alpha \cdot A(s,s')) \log \pi_g(s' \mid s) \Big] \tag{2}$$

where the advantage $A(s,s') = r + \gamma V(s') - V(s)$ serves as the behavior cloning weight, and α is the temperature parameter to prioritize value maximization over state-wise imitation.

For the execute-policy π_e, POR employs a supervised learning framework and trains π_e by maximizing the likelihood of the actions given the states and next states: $\max_{\pi_e} \mathbb{E}_{(s,a,s')\sim\mathcal{D}}[\log \pi_e(a \mid s, s')]$. During evaluation phase, given the current state s, we can sample the optimized next state s' from $\pi_g(s'|s)$, and get final action simply as $a^* = \pi_e(a \mid s, \pi_g(s'|s))$.

Time-reversal symmetry for generalizable offline RL. Recently, leveraging fundamental, universally held symmetries of dynamics such as T-symmetry discovered in classical and quantum mechanics [30, 31] has been shown to be a promising approach to enhance the generalization of offline RL [16, 3]. Specifically, if we model the system dynamics with measurements $\mathbf{x}$ as a set of non-linear first-order differential equations (ODEs) expressed as $\frac{d\mathbf{x}}{dt} = F(\mathbf{x})$, a dynamical system is said to exhibt *time-reversal symmetry* if there is an invertible transformation Γ that reverses the direction of time: i.e., $d\Gamma(\mathbf{x})/dt = -F(\Gamma(\mathbf{x}))$. For the discrete-time MDP setting, the T-symmetry can be extended as learning a pair of ODE forward dynamics $F(s,a) \to \dot{s}$ and reverse dynamics $G(s',a) \to -\dot{s}$, and require them to satisfy $F(s,a) = -G(s',a)$ [16], where the time-derivative of state $\dot{s} = \frac{ds}{dt}$ is approximated as $s' - s$.

Based on this intuition, TSRL [16] constructed an encoder-decoder structured *T-symmetry enforced dynamics model* (TDM) for representation learning, which embeds a pair of latent ODE forward and reverse dynamics to enforce T-symmetry. TSRL achieves impressive performance under small-sample settings, and its variant has been successfully deployed for real-world industrial control [3], but it still has some limitations. First, TSRL only uses the learned encoder from TDM to derive the latent representations, without fully exploiting the rich dynamics-related information for downstream policy learning. Second, its representation learning scheme uses both state and action as inputs, forcing TSRL to involve policy-induced actions during policy optimization, which inevitably requires adding a conservative action-level behavioral constraint as in TD3+BC [7] to stabilize training. Moreover, involving action as an input for representation learning is also prone to capturing biased behaviors in the behavioral policy, which could impede learning fundamental, distribution-agnostic dynamics patterns in data. Please refer to Appendix A for a more detailed comparison and discussion.

3 Offline RL via T-symmetry Enforced Latent State-Stitching

We now present our proposed method, TELS, which comprises a T-symmetry enforced inverse dynamics model (TS-IDM) integrated with an effective offline policy optimization procedure operated in latent state space (illustrated in Figure 1). TS-IDM overcomes multiple drawbacks of TDM in TSRL [16], which not only extracts the generalizable, T-symmetry preserving representations from the limited data, but also can be seamlessly used as an execute-policy for optimal action extraction.

3.1 T-symmetry enforced inverse dynamic model

As illustrated in Figure 1, if inspecting the input and output of our proposed TS-IDM, it functions similarly to an inverse dynamics model that takes current and next state (s, s') as input and outputs the predicted action a. However, TS-IDM's architecture is special in several aspects. In its interior, it comprises a state encoder $\phi_s(s) = z_s$ and a corresponding decoder $\psi_s(z_s) = \hat{s}$; a latent inverse dynamics model $h_{inv}(z_s, z_{s'}) = z_a$ followed by an action decoder $\psi_a(z_a) = \hat{a}$; and most importantly,

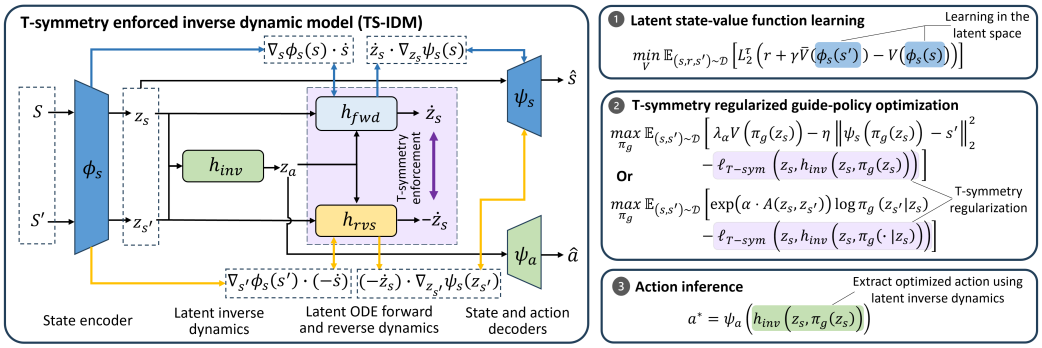


Figure 1: Overview of **T**-symmetry **E**nforced **L**atent **S**tate-Stitching (TELS) framework.

a pair of T-symmetry enforced latent ODE forward and reverse dynamics predictors $h_{fwd}(z_s, z_a) = \dot{z}_s$ and $h_{rvs}(z_{s'}, z_a) = -\dot{z}_s$. In the following content, we will dive into the design intuitions and learning objectives of these components.

Encoding and decoding. As previously discussed, constructing an informative and well-structured latent space is critical for sample-efficient offline policy optimization. To this end, we introduce a state encoder $\phi_s(s) = z_s$ to map a state s into corresponding latent representation z_s, and also a state decoder $\psi_s(z_s) = s$ to reconstruct the original state from its latent embedding, ensuring that the learned latent representations remain faithful to the original state space and avoid excessive distortion.

We then construct a latent inverse dynamics model $h_{inv}(z_s, z_{s'}) = z_a$, which infers the latent action z_a from the latent state transitions $(z_s, z_{s'})$. By inferring actions from state transitions, the learned latent space implicitly encodes the underlying dynamics of the environment. Moreover, the inverse dynamic model h_{inv} can be integrated with a pair of latent ODE dynamic models to derive the T-symmetry property of the system, which we will introduce in more detail shortly. Finally, to ensure that the inferred actions are both meaningful and interpretable, we employ an action decoder $\psi_a(z_a) = \hat{a}$ to map the latent action back to its original action space. We can thus formulate the reconstruction loss for the states and actions as follows:

$$\ell_{\text{rec}}(s, a, s') = \underbrace{\|\psi_s(\phi_s(s)) - s\|_2^2}_{\text{reconstruction loss of states}} + \underbrace{\|\psi_a(h_{inv}(z_s, z_{s'})) - a\|_2^2}_{\text{reconstruction loss of actions}} \tag{3}$$

Latent ODE forward and reverse dynamics. Drawing inspiration from previous research that integrates physics-informed insights into dynamical systems modeling [32, 33, 31, 16], we embed a pair of latent ODE forward and reverse dynamics $h_{fwd}(z_s, z_a) = \dot{z}_s$ and $h_{rvs}(z_{s'}, z_a) = -\dot{z}_s$ to separately capture the forward and reverse time evolution in the latent states. We are interested in modeling ODE systems because it encourages learning parsimonious models helpful to uncover fundamental properties from the data that can maximally promote generalization [32, 33]. Note that based on the chain rule, we can derive the supervision signal for the latent dynamics models with $\dot{z}_s = \frac{dz}{dt} = \frac{dz_s}{ds} \cdot \frac{ds}{dt} = \nabla_s z_s \cdot \dot{s} = \nabla_s \phi_s(s) \cdot \dot{s}$ to enforce the ODE property. Therefore, we introduce the following training losses for h_{fwd} and h_{rvs}:

$$\begin{aligned}\ell_{\text{dyn}}(s, s') = & \underbrace{\|(\nabla_s z_s)\dot{s} - \dot{z}_s\|_2^2}_{\text{latent ODE forward dynamics}} + \underbrace{\|(\nabla_{s'} z_{s'})(-\dot{s}) - (-\dot{z}_s)\|_2^2}_{\text{latent ODE reverse dynamics}} \\ = & \|\nabla_s \phi_s(s)\dot{s} - h_{fwd}(z_s, z_a)\|_2^2 + \|\nabla_{s'} \phi_s(s')(-\dot{s}) - h_{rvs}(z_{s'}, z_a)\|_2^2,\end{aligned} \tag{4}$$

where the latent action z_a is obtained from the latent inverse dynamics model $h_{inv}(z_s, z_{s'})$.

ODE property enforcement on state decoder. Note that in $\ell_{\text{dyn}}(s, s')$, we actually implicitly enforced the ODE property on the state encoder ϕ_s, the same should also apply to the state decoder ψ_s to ensure compatibility with the T-symmetry formalism, i.e. the time-derivative of the state encoder $\frac{d\phi_s(s)}{dt}$ and decoder $\frac{d\psi_s(z_s)}{dt}$ should behave in the same way as $\dot{z}_s$ and $\dot{s}$. Similar to the previous treatment on the state encoder, as $\dot{s} = \frac{d\psi_s(z_s)}{dt} = \frac{d\psi_s(z_s)}{dz_s} \cdot \frac{dz_s}{dt} = \nabla_{z_s} \psi_s(z_s) \cdot \dot{z}_s$, we can use the following objective to enforce the ODE property for the state decoder ψ_s:

$$\ell_{\text{ode}}(s, s') = \underbrace{\|\nabla_{z_s}\psi_s(z_s)\cdot\dot{z}_s - \dot{s}\|_2^2}_{\text{enforce ODE of } \psi_s \text{ on } h_{fwd}} + \underbrace{\|\nabla_{z_{s'}}\psi_s(z_{s'})\cdot(-\dot{z}_s) - (-\dot{s})\|_2^2}_{\text{enforce ODE of } \psi_s \text{ on } h_{rvs}}$$
$$= \|\nabla_{z_s}\psi_s(z_s)\cdot h_{fwd}(z_s, z_a) - \dot{s}\|_2^2 + \|\nabla_{z_{s'}}\psi_s(z_{s'})\cdot h_{rvs}(z_{s'}, z_a) + \dot{s}\|_2^2 \quad (5)$$

Again, the latent action z_a is obtained from $h_{inv}(z_s, z_{s'})$.

Notably, the ODE property enforcement in Eq. (5) is not considered in the T-symmetry enforced dynamics model (TDM) proposed by TSRL [16]. In other words, TDM only enforces the ODE properties for encoders but not for decoders. This can cause inconsistency between the learned dynamics and the underlying ODE structure, leading to inaccurate or misaligned ODE representations.

T-symmetry enforcement. To further regularize the learned latent representations, we incorporate the extended version of T-symmetry [16] by requiring $h_{fwd}(z_s, z_a) = -h_{rvs}(z_{s'}, z_a)$, which corresponds to the following T-symmetry consistency loss:

$$\ell_{\text{T-sym}}(z_s, z_a) = \|h_{fwd}(z_s, z_a) + h_{rvs}(z_s + h_{fwd}(z_s, z_a), z_a)\|_2^2 \quad (6)$$

where we use the fact that $z_{s'} = z_s + \dot{z}_s = z_s + h_{fwd}(z_s, z_a)$ and $h_{rvs}(z_s + h_{fwd}(z_s, z_a), z_a) = -\dot{z}_s = -h_{fwd}(z_s, z_a)$ to further couple the learning process of h_{fwd} and h_{rvs}. Moreover, given a latent state-action pair (z_s, z_a), the above T-symmetry consistency loss can also serve as an evaluation metric to assess their agreement with the learned TS-IDM. A large T-symmetry loss indicates that the latent state-action representation (z_s, z_a) induced by some (s, s') may not satisfy the fundamental dynamics pattern, making it more likely to be a problematic or non-generalizable sample.

Overall learning objective. Finally, the complete training loss function of TS-IDM is as follows:

$$\mathcal{L}_{\text{TS-IDM}} = \sum_{(s,a,s')\in\mathcal{D}} \Big[\ell_{\text{rec}} + \beta\cdot(\ell_{\text{dyn}} + \ell_{\text{ode}} + \ell_{\text{T-sym}})\Big](s, a, s') \quad (7)$$

where β is a hyperparameter that balances extracting fundamental dynamics properties and ensuring the interpretability of the learned representation. Note that we use the same β for ℓ_{dyn}, ℓ_{ode}, and $\ell_{\text{T-sym}}$ terms, this is to ensure that the ODE property and T-symmetry regularization are enforced in the same scale for state encoder ϕ_s, decoder ψ_s, latent inverse dynamics h_{inv}, latent ODE forward and reverse dynamics h_{fwd} and h_{rvs}, as all of them are strongly coupled. This forms a strongly consistent, T-symmetry preserving ODE system to capture the fundamental dynamics properties in the offline dataset, while also helping reduce unnecessary hyperparameters of the learning process. In Appendix B.3, we provide detailed results on the impact of β on the learning process of TS-IDM.

3.2 Latent space offline policy optimization

Once we have learned TS-IDM, we can extract three highly useful components from it to facilitate sample-efficient downstream offline policy optimization, including: 1) a state encoder $\phi(s)$ that provides an ideal, well-behaved latent space for state-stitching; 2) T-symmetry consistency as an additional regularizer to prevent erroneous generalization when learning a guide-policy in the latent state space; and 3) the TS-IDM itself can serve as an execute-policy as in POR [20] to extract optimized action given a learned guide-policy.

Latent state-value functions learning. Based on the state encoder $\phi_s(s)$ from the learned TS-IDM, we can convert the entire offline policy optimization process into the latent state space, which enjoys both a stable learning process and generalizability due to more compact and well-behaved representations. Specifically, we can use a similar expectile regression loss as in Eq. (1) to learn a state-value function $V(z_s)$, but in the latent state space:

$$\min_V \mathbb{E}_{(s,r,s')\sim\mathcal{D}}\Big[L_2^\tau\left(r + \gamma\bar{V}\left(\phi_s(s')\right) - V\left(\phi_s(s)\right)\right)\Big] \quad (8)$$

T-symmetry regularized guide-policy optimization. A key benefit of learning within the T-symmetry preserving latent space is that, as T-symmetry captures what is essential and invariant about the dynamical system, it can provide generalizable information even for OOD samples beyond the offline dataset. This naturally favors learning a reward-maximizing guide-policy π_g in the latent space, which can enjoy more effective state-stitching. Moreover, different from POR [20], by leveraging the

T-symmetry consistency term $\ell_{\text{T-sym}}(\cdot)$ in Eq. (6) as an additional regularizer, we can prevent π_g from outputting problematic and non-generalizable latent next state, thereby further enhancing logical state-wise OOD generalization. In TELS, we provide two instantiations for guide-policy optimization, depending on the choice of using deterministic policy $\pi_g(z_s)$ or stochastic policy $\pi_g(z_{s'}|z_s)$:

- Deterministic policy:

$$\max_{\pi_g} \mathbb{E}_{(s,s')\sim\mathcal{D}}\Big[\lambda_\alpha V(\pi_g(z_s)) - \eta\|\psi_s(\pi_g(z_s)) - s'\|_2^2 - \ell_{\text{T-sym}}\left(z_s, h_{ivs}\left(z_s, \pi_g(z_s)\right)\right)\Big] \tag{9}$$

- Stochastic policy:

$$\max_{\pi_g} \mathbb{E}_{(s,s')\sim\mathcal{D}}\Big[\exp(\alpha\cdot A(z_s, z_{s'}))\log \pi_g(z_{s'} \mid z_s) - \ell_{\text{T-sym}}(z_s, h_{ivs}(z_s, \pi_g(\cdot|z_s))\Big] \tag{10}$$

where $z_s = \phi_s(s)$, $z_{s'} = \phi_s(s')$, and $A(z_s, z_{s'}) = r + \gamma V\left(z_{s'}\right) - V\left(z_s\right)$. For the deterministic policy $\pi_g(z_s)$, we optimize the guide-policy by maximizing the latent state-value function V weighted by a normalization term λ_α, together with two extra regularization terms. The first regularizes the next state decoded from the guide-policy using state decoder ψ_s should not deviate too much from the next state s' in the dataset. The second term regularizes the guide-policy induced latent state-action pair (i.e., $(z_s, z_a) = (z_s, h_{inv}(z_s, \pi_g(z_s)))$) to comply with the T-symmetry consistency specified in the learned TS-IDM. For the stochastic guide-policy $\pi_g(z_{s'}|z_s)$, we adopt a similar AWR-style objective as in Eq. (2), while also incorporating the T-symmetry consistency regularization as in the deterministic version. In our experiments, we find that the deterministic version objective Eq. (9) works well for the MuJoCo locomotion tasks, while the stochastic version Eq. (10) works better for more complex Antmaze tasks [17], potentially due to more stochastic nature of the task environment.

Action inference. After learning the guide-policy π_g, we can further use it to extract the optimized action for control. To do this, we can simply use the optimized latent next state $z^*_{s'}$ obtained from guide-policy $\pi_g(z_s)$ or $\pi_g(\cdot|z_s)$ as the goal state, and plug it into the learned latent inverse dynamics model $h_{inv}(z_s, z_{s'})$ in TS-IDM to replace $z_{s'}$. The final action can be extracted by decoding the resulting latent action from h_{inv} using the action decoder ψ_a :

$$a^* = \psi_a\left(h_{inv}\left(z_s, \pi_g(z_s)\right)\right) \tag{11}$$

Note that there is no training process needed for this stage. Moreover, throughout our policy optimization process, actions are completely not involved, allowing TELS to directly bypass the conservatism issue caused by the action-level regularization. Please refer to Algorithm 1 in Appendix C for the complete training and inference procedure of TELS.

4 Experiments

In this section, we present the evaluation results of TELS on the D4RL benchmark tasks [17] against behavior cloning (BC), and existing offline RL methods: TD3+BC [7], CQL [8], IQL [34], DOGE [15], POR [20], model-based methods MOPO [35] and COMBO [36], diffusion-based method IDQL [37], and TSRL [16], the current SOTA method in small-sample settings. To demonstrate the effectiveness of TELS in solving real-world tasks, we also validate TELS in a real-world industrial control environment, which is a data center (DC) cooling control testbed built by a recent work [38]. Moreover, we conduct additional experiments to further evaluate the OOD generalizability of TELS on a challenging task, and the strengths of the representations learned with TS-IDM in improving small-sample performance. More results and implementation details can be found in Appendix B, C.

4.1 Comparative evaluation on small-sample setting

Evaluation on D4RL benchmarks. In Table 1, we evaluate TELS against baseline methods on challenging reduced-size D4RL datasets (5k$\sim$100k samples, about 0.5$\sim$10% of their original sizes)[1]. These small-sample tasks are particularly challenging for offline RL algorithms, as the data only sparsely cover the state-action space and require strong OOD generalization capability for algorithms to achieve reasonable performance. Results on full D4RL datasets can be found in Appendix B.1.

[1]We use the same reduced-size MuJoCo datasets from the TSRL paper [38], and randomly sub-sample 100k Antmaze datasets for experiments. We use the original Adroit datasets for evaluation, as they are already small.

Table 1: Normalized scores on reduced-size D4RL datasets (averaged over the final 10 evaluations with 5 seeds)

Task	Size (ratio)	BC	TD3+BC	MOPO	COMBO	CQL	IQL	DOGE	IDQL	POR	TSRL	TELS
Hopper-m	10k (1%)	29.7±11.7	40.1±18.6	5.5 ± 2.3	30.2 ± 28.0	43.1±24.6	46.7±6.5	44.2 ± 10.2	44.2±12.1	46.4 ± 1.7	62.0±3.7	**77.3 ± 10.7**
Hopper-mr	10k (2.5%)	12.1±5.3	7.3±6.1	6.8 ± 0.3	10.6 ± 13.1	2.3±1.9	13.4±3.1	17.9 ± 4.5	21.7±7.0	17.4 ± 6.2	21.8±8.2	**43.2 ± 3.5**
Hopper-me	10k (0.5%)	27.8±10.7	17.8±7.9	5.8 ± 5.8	13.9 ± 22.0	29.9±4.5	34.3±8.7	50.5 ± 25.2	43.2±4.4	37.9 ± 6.1	50.9±8.6	**100.9 ± 6.8**
Halfcheetah-m	10k (1%)	26.4±7.3	16.4±10.2	-1.1 ± 4.1	16.5 ± 2.4	35.8±3.8	29.9±0.12	36.2 ± 3.4	36.4±1.5	33.3±3.2	38.4±3.1	**40.8 ± 0.6**
Halfcheetah-mr	10k (5%)	14.3±7.8	17.9±9.5	11.7 ± 5.2	11.8 ± 15.3	8.1±9.4	22.7±6.4	23.4 ± 3.6	26.7±1.0	27.5±3.6	28.1±3.5	**33.2 ± 1.0**
Halfcheetah-me	10k (0.5%)	19.1±9.4	15.4±10.7	-1.1 ± 1.4	5.2 ± 6.1	26.5±10.8	10.5±8.8	26.7 ± 6.6	38.8±1.9	34.7±2.6	39.9±21.1	**40.7 ± 1.2**
Walker2d-m	10k (1%)	15.8±14.1	7.4±13.1	3.1 ± 4.7	3.6 ± 1.1	18.8±18.8	22.5±3.8	45.1 ± 10.2	31.7±14.2	22.2±3.6	49.7±10.6	**62.4 ± 5.3**
Walker2d-mr	10k (3.3%)	1.4±1.9	5.7±5.8	3.3 ± 2.7	4.2 ± 15.6	8.5±2.19	10.7±11.9	13.5 ± 8.4	12.2±10.5	14.8±4.2	26.0±11.3	**54.8 ± 6.0**
Walker2d-me	10k (0.5%)	21.7±8.2	7.9±9.1	0.6 ± 2.7	0.1 ± 0.1	19.1±14.4	26.5±8.6	35.3 ± 11.6	21.8±14.5	20.1±8.6	46.4±17.4	**87.4 ± 13.3**
Antmaze-u	10k (1%)	44.7 ± 42.1	0.7 ± 1.2	0.0	0.0	5.5 ± 2.3	65.1 ± 19.4	56.3 ± 24.4	67.5 ±12.4	6.1 ± 7.3	76.1 ± 15.6	**88.7 ± 7.7**
Antmaze-u-d	10k (1%)	24.1 ± 22.2	16.27 ± 16.4	0.0	0.0	0.5 ± 0.1	34.6 ± 18.5	41.7 ± 18.9	55.1 ± 36.8	42.1 ± 14.2	52.2 ± 22.1	**60.9 ± 16.9**
Antmaze-m-d	100k (10%)	0.0	0.0	0.0	0.0	0.0	4.8 ± 5.9	0.0	9.0 ±3.4	0.0	0.0	**47.2 ± 17.3**
Antmaze-m-p	100k (10%)	0.0	0.0	0.0	0.0	0.0	12.5 ± 5.4	0.0	9.4 ± 14.7	0.0	0.0	**62.9 ± 17.8**
Antmaze-l-d	100k (10%)	0.0	0.0	0.0	0.0	0.0	3.6 ± 4.1	0.0	16.1 ± 8.4	0.0	0.0	**39.8 ± 14.1**
Antmaze-l-p	100k (10%)	0.0	0.0	0.0	0.0	0.0	3.5 ± 4.1	0.0	9.7 ±8.5	0.0	0.0	**47.3 ± 13.1**
Pen-human	5k (100%)	34.4	8.4	9.7	27.7	37.5	71.5	42.6 ± 16.3	67.9 ± 17.3	64.1 ± 25.3	80.1 ± 18.1	77.4 ± 17.2
Hammer-human	5k (100%)	1.5	2.0	0.2	0.2	**4.4**	1.4	-1.2 ± 0.2	2.7 ± 1.3	0.2 ± 0.1	0.2 ± 0.3	3.6 ± 1.5
Door-human	5k (100%)	0.5	0.5	-0.2	-0.3	9.9	4.3	-1.1 ± 0.2	10.5 ± 1.5	0.1 ± 0.1	0.5 ± 0.3	**11.8 ± 1.6**
Relocate-human	5k (100%)	0.0	-0.3	-0.2	-0.3	0.2	0.1	0.1 ± 0.2	0.2 ± 0.1	0.1 ± 0.1	0.1 ± 0.1	0.3 ± 0.2

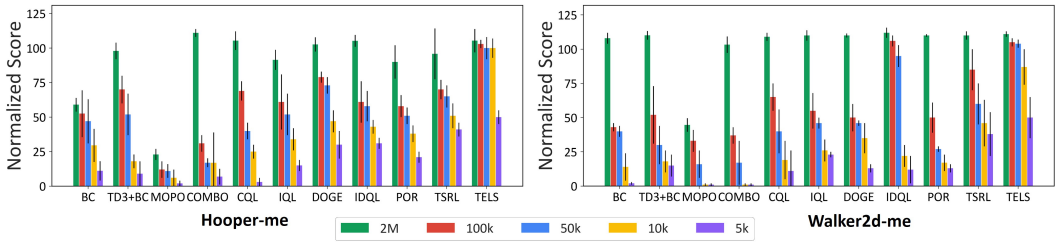


Figure 2: Performance of TELS against baselines under different data sizes

As shown in Table 1, most baselines fail to learn reasonable policies under small datasets, especially in the challenging 100k Antmaze-medium/large datasets. For example, conventional offline RL methods like TD3+BC and CQL perform poorly on small datasets, primarily due to their over-conservative data-related policy constraints. Model-based methods also perform badly due to insufficient samples to learn accurate dynamics models and the use of problematic model rollout data. Baselines that have generalization promotion designs, such as DOGE and TSRL, perform slightly better but still fail miserably in the challenging Antmaze-m/l tasks, as they still adopt conservative action-level constraints to stabilize policy learning. Recent diffusion-based methods like IDQL, although perform well on large datasets, struggle to learn when given limited data. By contrast, TELS dominates the chart and outperforms all other baselines in all tasks, sometimes by a large margin. This is attributed to the leverage of fundamental, data distribution-agnostic T-symmetry property for policy learning, which greatly improves the OOD generalization performance. This is evident when observing the huge performance difference between POR and TELS, as the former shares a similar policy optimization procedure but does not use the T-symmetry enforced representation and policy regularization.

We also evaluate the performance of the algorithms across different dataset sizes in Figure 2. The results show that TELS can robustly maintain reasonable performance even with only 5k samples, surpassing all the other methods, while most baseline methods suffer from significant performance drop when training samples are decreased.

Evaluation on real-world industrial control test environment. To further demonstrate the effectiveness of TELS in solving real-world industrial control tasks, we deploy TELS in a real-world DC cooling control testbed [38] and compare against CQL, IQL, and TSRL. This testbed comprises 22 servers with oscillating server loads and an Air-Cooling Unit (ACU) for cooling control. A small historical operational dataset (43k real-world samples collected over 61 days) with 105 state-action features is used for policy learning. The goal is to improve the energy efficiency of the DC's cooling systems (minimizing the Air-side Cooling Load Factor (ACLF), calculated as the ratio of energy consumption of ACU to servers), while satisfying thermal safety constraints (no overheating). We follow the same real-world experiment setup as in [38] and present the details in Appendix D.

As shown in Table 2, under a similar server energy consumption level (around 40 kWh), TELS learns the best control policy, achieving 20.17% ACLF while maintaining zero thermal safety violations. CQL learns a naive policy that achieves lower ACLF but with significant thermal safety violations. This shows TELS's effectiveness in solving real-world complex industry control tasks.

Table 2: Evaluation results in the real-world DC cooling control testbed (6-hour length experiments)

Testbed	CQL	IQL	TSRL	TELS
Server energy consumption (kWh)	41.44	39.80	40.30	40.61
ACU energy consumption (kWh)	4.16	16.27	10.95	8.19
Energy efficiency measure: ACLF (the lower the better)	**10.3**%	40.89%	27.16%	**20.17%** ↓
Percentage of thermal safety violation (the lower the better)	40.99%	**0.00**%	**0.00**%	**0.00**%

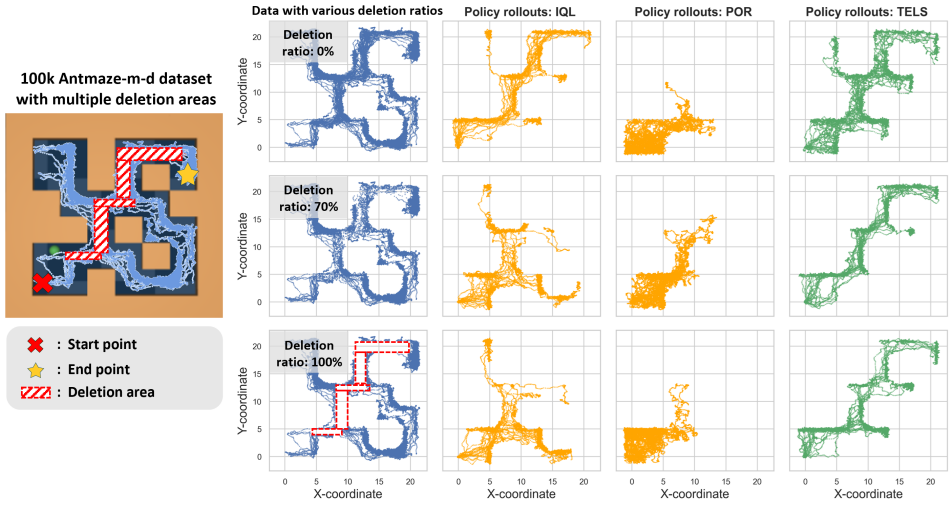


Figure 3: **Left:** Illustration of the 100k Antmaze-m-d task with multiple deletion areas, where the red cross denotes the start point, the yellow star denotes the goal locations, and the red shaded areas denote the data deletion regions. **Right:** Visualization of the training dataset and policy rollout trajectories generated by trained policies from various algorithms under varying deletion ratios.

4.2 Analysis and ablation of TELS

OOD generalization capability. To further examine the OOD generalizability of TELS, we construct a very challenging task based on the reduced-size 100k Antmaze-m-d dataset, as illustrated in Figure 3. Specifically, we randomly remove samples within 5 critical regions along the critical paths from the start to the goal locations. This task requires extremely strong OOD generalization capability to solve, as the vital information for the optimal trajectory is extremely scarce or completely OOD. We train IQL, POR, and TELS on the remaining data and plot their policy rollouts over 20 episodes for performance evaluation and behavior analyses (due to page limit, we also include results for IDQL, DOGE, TSRL in Appendix B.2). As shown in Figure 3, IQL can only achieve some success when the deletion ratio is 0%, and POR fails to reach the goal in all cases. By contrast, TELS consistently learns optimal policy even with 70% and 100% deletion rates. It can effectively utilize the limited information provided in the sparse remaining data samples at the boundaries of the deletion areas for policy learning. These highlight the extraordinary OOD generalization capability of TELS in extremely challenging low-data regimes.

Effectiveness of the learned representations. To verify the effectiveness of the learned latent representation in TS-IDM, we use TS-IDM's state encoder $\phi_s(s)$ as the representation learning module on top of two conventional offline RL methods: IQL and TD3+BC. Figure 4 (left) reveals significant performance improvements and variance reduction when IQL and TD3+BC are trained within the latent state space induced by $\phi_s(s)$, suggesting that TS-IDM indeed learns compact and generalizable representations that benefit policy learning. To further evaluate the quality of TS-IDM's representations, in Figure 4 (right), we replace TS-IDM in TELS with other representation learning methods, including autoencoder ("AE-rep"), variational autoencoder ("VAE-rep") [39], and contrastive learning method SimCLR ("Contras-rep") [40]. The results show that the TS-IDM representation achieves substantially better performance as compared to AE, VAE, and contrastive representations, due to the information-rich and well-behaved latent space learned in TS-IDM.

Ablations on the design components of TS-IDM. To examine the impact of each component in TS-IDM, we evaluate various variants of TS-IDM, starting with a vanilla latent inverse dynamics model with encoder and decoders, denoted as " $\phi/\psi+ h_{inv}$", gradually adding latent forward and reverse dynamics "h_{fwd}, h_{rvs}", ODE property enforcement "ℓ_{ode}", and eventually the T-symmetry consistency loss "$\ell_{\text{T-sym}}$", resulting in the full TS-IDM. Results on 10k datasets are shown in Table 3.

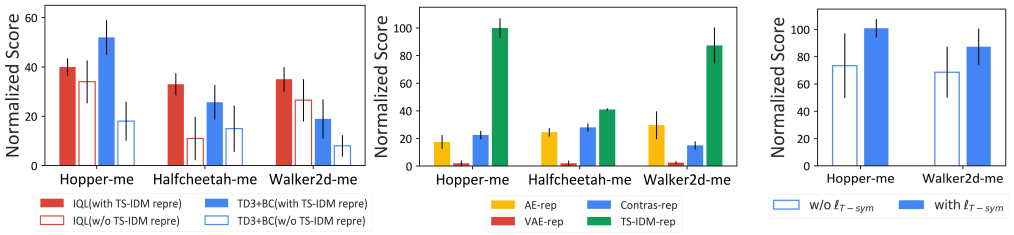


Figure 4: **Left:** Performance of IQL and TD3+BC on 10k datasets with or without using the representation from TS-IDM. **Right:** The performance of TELS with different representation models on 10k datasets.

Figure 5: Impact of $\ell_{\text{T-sym}}$ on policy optimization.

We observe that the naïve autoencoder-based inverse dynamics model fails to provide reasonable representations. Incorporating dynamics-related information via latent dynamics is helpful, but the performance gain remains mild. Enforcing ODE properties on decoders greatly enhances the quality of learned representations. Lastly, enforcing T-symmetry consistency proves to be the strongest performance improvement factor, which greatly enhances the quality of the learned representations for downstream policy learning.

Table 3: Ablation on the design components of TS-IDM.

	Hopper-me	Halfcheetah-me	Walker2d-me
$\phi/\psi + h_{inv}$	17.2 $\pm$ 7.0	29.7 $\pm$ 3.6	24.5 $\pm$ 10.1
$\uparrow + h_{fwd}, h_{rvs}$	35.5 $\pm$ 7.3	31.3 $\pm$ 1.1	33.6 $\pm$ 9.2
$\uparrow + \ell_{\text{ode}}$	61.4 $\pm$ 23.7	31.2 $\pm$ 1.2	58.5 $\pm$ 18.1
$\uparrow + \ell_{\text{T-sym}}$	**100.9** $\pm$ **6.8**	**40.7** $\pm$ **1.2**	**87.4** $\pm$**13.3**

Ablations on regularizer terms in policy optimization. We also conduct ablation experiments in Figure 5 to validate the effectiveness of the T-symmetry consistency regularizer term $\ell_{\text{T-sym}}$ during the guide-policy optimization process of TELS. The results demonstrate that incorporating this term can effectively enhance performance while reducing variance, highlighting the importance of utilizing T-symmetry consistency regularization to promote OOD generalization and learning stability.

5 Related Work

Offline RL faces unique challenges in mitigating the risk of OOD exploitation. Evaluating value functions in OOD regions often results in inaccurate estimates, which can lead to severe value overestimation and misguiding policy learning. To mitigate this, most offline RL methods leverage data-related regularizations to stabilize the learning process. These include explicit behavior constraint techniques that penalize action divergence [6, 4, 7, 41], value regularization schemes to discourage policies from selecting OOD actions via modifying Bellman update [8, 9, 10, 11] or introducing uncertainty penalities [42, 43, 10], and in-sample learning methods [44, 12, 13, 14], which stabilize training by only using in-sample data for value and policy learning. While these methods perform reasonably well on datasets with sufficient state-action coverage, they often struggle in small-sample settings where exploiting OOD generalization is vital for achieving good performance. Recently, leveraging expressive model architectures such as Transformers and diffusion models [45, 46, 47, 48, 37, 49, 50, 51] have gained popularity in offline RL, due to their strong capability to fit complex data distributions. However, these models are overly heavy and require extensive amounts of data to learn, making them hardly usable for the small-sample setting.

6 Conclusion

We propose a highly sample-efficient offline RL algorithm that learns an optimized policy within the latent space regulated by the fundamental T-symmetry property. Specifically, we develop a T-symmetry enforced inverse dynamics model, TS-IDM, to construct a well-behaved and generalizable latent space, effectively mitigating the challenges of OOD generalization. By learning a T-symmetry regularized guide-policy within this latent space, we can obtain the reward-maximizing next state to serve as the goal state input in the learned TS-IDM for optimal action extraction. Through extensive experiments, we show that TELS achieves strong OOD generalization capability and SOTA small-sample performance. Moreover, we empirically show that TS-IDM can also function as a representation learning model to provide informative representations and enhance the performance of existing methods under the small-sample setting. One potential limitation of TELS is that strong ODE and T-symmetry property regularizations, although helpful for capturing fundamental patterns in data, sometimes could limit the model's expressive power (see Appendix B.3). Future studies can explore improved designs to perfectly balance fundamental pattern extraction and model expressivity.

References

[1] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[2] Xianyuan Zhan, Haoran Xu, Yue Zhang, Xiangyu Zhu, Honglei Yin, and Yu Zheng. Deepthermal: Combustion optimization for thermal power generating units using offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[3] Xianyuan Zhan, Xiangyu Zhu, Peng Cheng, Xiao Hu, Ziteng He, Hanfei Geng, Jichao Leng, Huiwen Zheng, Chenhui Liu, Tianshun Hong, Yan Liang, Yunxin Liu, and Feng Zhao. Data center cooling system optimization using offline reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, 2025.

[4] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pages 11761–11771, 2019.

[5] 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.

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

[7] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[8] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[9] Haoran Xu, Xianyuan Zhan, and Xiangyu Zhu. Constraints penalized q-learning for safe offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[10] Chenjia Bai, Lingxiao Wang, Zhuoran Yang, Zhi-Hong Deng, Animesh Garg, Peng Liu, and Zhaoran Wang. Pessimistic bootstrapping for uncertainty-driven offline reinforcement learning. In *International Conference on Learning Representations*, 2021.

[11] Jiafei Lyu, Xiaoteng Ma, Xiu Li, and Zongqing Lu. Mildly conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:1711–1724, 2022.

[12] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.

[13] 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. In *The Eleventh International Conference on Learning Representations*, 2023.

[14] Liyuan Mao, Haoran Xu, Weinan Zhang, and Xianyuan Zhan. Odice: Revealing the mystery of distribution correction estimation via orthogonal-gradient update. In *The Twelfth International Conference on Learning Representations*, 2024.

[15] 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.

[16] Peng Cheng, Xianyuan Zhan, Wenjia Zhang, Youfang Lin, Han Wang, Li Jiang, et al. Look beneath the surface: Exploiting fundamental symmetry for sample-efficient offline rl. *Advances in Neural Information Processing Systems*, 36, 2023.

[17] 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.

[18] Samarth Sinha, Ajay Mandlekar, and Animesh Garg. S4rl: Surprisingly simple self-supervision for offline reinforcement learning in robotics. In *Conference on Robot Learning*, pages 907–917. PMLR, 2022.

[19] Shengpu Tang, Maggie Makar, Michael Sjoding, Finale Doshi-Velez, and Jenna Wiens. Leveraging factored action spaces for efficient offline reinforcement learning in healthcare. *Advances in neural information processing systems*, 35:34272–34286, 2022.

[20] Haoran Xu, Jiang Li, Jianxiong Li, and Xianyuan Zhan. A policy-guided imitation approach for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2022.

[21] Seohong Park, Dibya Ghosh, Benjamin Eysenbach, and Sergey Levine. Hiql: Offline goal-conditioned rl with latent states as actions. *Advances in Neural Information Processing Systems*, 36, 2024.

[22] Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International conference on machine learning*, pages 5639–5650. PMLR, 2020.

[23] Rishabh Agarwal, Marlos C Machado, Pablo Samuel Castro, and Marc G Bellemare. Contrastive behavioral similarity embeddings for generalization in reinforcement learning. In *International Conference on Learning Representations*, 2021.

[24] Mengjiao Yang and Ofir Nachum. Representation matters: offline pretraining for sequential decision making. In *International Conference on Machine Learning*, pages 11784–11794. PMLR, 2021.

[25] Matthias Weissenbacher, Samarth Sinha, Animesh Garg, and Kawahara Yoshinobu. Koopman q-learning: Offline reinforcement learning via symmetries of dynamics. In *International Conference on Machine Learning*, pages 23645–23667. PMLR, 2022.

[26] Masatoshi Uehara, Xuezhou Zhang, and Wen Sun. Representation learning for online and offline rl in low-rank mdps. *arXiv preprint arXiv:2110.04652*, 2021.

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

[28] Gerhard Neumann and Jan Peters. Fitted q-iteration by advantage weighted regression. *Advances in neural information processing systems*, 21, 2008.

[29] 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.

[30] Jeroen SW Lamb and John AG Roberts. Time-reversal symmetry in dynamical systems: a survey. *Physica D: Nonlinear Phenomena*, 112(1-2):1–39, 1998.

[31] In Huh, Eunho Yang, Sung Ju Hwang, and Jinwoo Shin. Time-reversal symmetric ode network. *Advances in Neural Information Processing Systems*, 33:19016–19027, 2020.

[32] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

[33] Kathleen Champion, Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences*, 116(45):22445–22451, 2019.

[34] 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.

[35] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[36] Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967, 2021.

[37] Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

[38] Xianyuan Zhan, Xiangyu Zhu, Peng Cheng, Xiao Hu, Ziteng He, Hanfei Geng, Jichao Leng, Huiwen Zheng, Chenhui Liu, Tianshun Hong, et al. Data center cooling system optimization using offline reinforcement learning. *International Conference on Learning Representations*, 2025.

[39] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR 2014 : International Conference on Learning Representations (ICLR) 2014*, 2014.

[40] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[41] Tenglong Liu, Yang Li, Yixing Lan, Hao Gao, Wei Pan, and Xin Xu. Adaptive advantage-guided policy regularization for offline reinforcement learning. In *Forty-first International Conference on Machine Learning*, volume 235, pages 31406 – 31424. PMLR, 2024.

[42] Yue Wu, Shuangfei Zhai, Nitish Srivastava, Joshua M Susskind, Jian Zhang, Ruslan Salakhutdinov, and Hanlin Goh. Uncertainty weighted actor-critic for offline reinforcement learning. In *International Conference on Machine Learning*, pages 11319–11328. PMLR, 2021.

[43] 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.

[44] 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.

[45] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

[46] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

[47] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

[48] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[49] Liyuan Mao, Haoran Xu, Xianyuan Zhan, Weinan Zhang, and Amy Zhang. Diffusion-dice: In-sample diffusion guidance for offline reinforcement learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[50] Yinan Zheng, Ruiming Liang, Kexin ZHENG, Jinliang Zheng, Liyuan Mao, Jianxiong Li, Weihao Gu, Rui Ai, Shengbo Eben Li, Xianyuan Zhan, and Jingjing Liu. Diffusion-based planning for autonomous driving with flexible guidance. In *The Thirteenth International Conference on Learning Representations*, 2025.

[51] Tenglong Liu, Jianxiong Li, Yinan Zheng, Haoyi Niu, Yixing Lan, Xin Xu, and Xianyuan Zhan. Skill expansion and composition in parameter space. In *The Thirteenth International Conference on Learning Representations*, 2025.

[52] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, pages 12519–12530, 2019.

[53] Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, 2020.

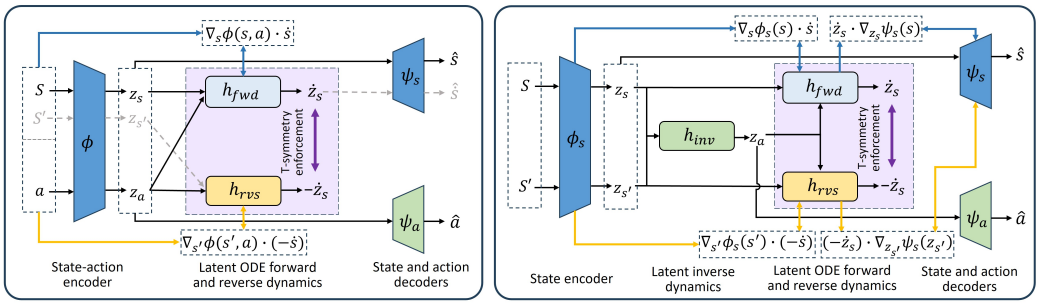
[54] Jianhao Wang, Wenzhe Li, Haozhe Jiang, Guangxiang Zhu, Siyuan Li, and Chongjie Zhang. Offline reinforcement learning with reverse model-based imagination. *Advances in Neural Information Processing Systems*, 34:29420–29432, 2021.

[55] Marc Rigter, Bruno Lacerda, and Nick Hawes. Rambo-rl: Robust adversarial model-based offline reinforcement learning. *Advances in neural information processing systems*, 35:16082–16097, 2022.

Appendix

A Additional Discussion on Related Works

In this section, we present a detailed discussion of the connections and differences between our proposed method, TELS with TSRL [16], POR [20], and conventional model-based approaches [52, 35, 53, 36, 54, 2].



(a) T-symmetry enforced dynamic model (TDM) in TSRL

(b) Our proposed T-symmetry enforced inverse dynamic model (TS-IDM)

Figure 6: Comparison of the architecture between TDM in TSRL and our proposed TS-IDM in TELS.

Connection and differences with TSRL. As illustrated in Figure 6, both TSRL and TELS leverage the T-symmetry consistency enforcement to construct the latent space. Specifically, in Figure 6 (a), TSRL employs a T-symmetry-enforced dynamics model (TDM), which models system dynamics by incorporating paired latent ODE forward and reverse dynamics to enforce T-symmetry. In contrast, Figure 6 (b) illustrates our proposed T-symmetry-enforced inverse dynamics model (TS-IDM), which integrates T-symmetry constraints into both forward and reverse dynamics while incorporating an inverse dynamics model. We emphasize the main differences between TELS and TSRL as follows:

- **Architecture:** As presented in Figure 6 (a), TDM jointly encodes state-action pairs to form the latent space, which may capture behavioral biases from the dataset (e.g., expert-specific action patterns) and impede learning fundamental, distribution-agnostic dynamics patterns in data. In contrast, Figure 6 (b) illustrates that TS-IDM overcomes these limitations by adopting a state-only modeling approach, focusing on the underlying latent state variations. Additionally, the only useful component of the learned TDM for downstream policy learning is its encoder $\phi(s, a)$, wasting the dynamics-related information captured by the model. In contrast, TS-IDM trains an inverse dynamics model within the T-symmetry-enforced latent space, which can be reused as an execute-policy to extract optimal actions.
- **Detailed model design:** As shown in Figure 6 (a), TDM only enforces the ODE property for its encoder but not the decoder, which could lead to inconsistency between the learned dynamics and the underlying ODE structure, resulting in inaccurate or misaligned ODE representations. To address this problem, we introduce the loss term ℓ_{ode} in Eq. (5) specifically to achieve this goal. This design is very important as it can greatly enhance the coupling among the different elements in the model and results in a more stable learning process.
- **Training procedure:** In TSRL, the TDM encoder and decoders must be pre-trained before joint training on other components to avoid stability issues. In contrast, our proposed TS-IDM does not require pre-training; all components can be learned jointly in a single stage. Additionally, TDM requires adding L1-norm regularization to the parameters of the latent forward and reverse dynamics models to stabilize the learning process. This is unnecessary in TS-IDM (see Eq. (7)), as the design of our proposed TS-IDM enables strongly coupled and consistent relationships among all its internal components. The learning curves of TS-IDM can be found in Appendix F.
- **Policy optimization:** Since TDM requires both state and action as inputs to derive the latent representations, it is constrained to Q-function maximization for policy optimization. Consequently, TSRL adopts the TD3+BC framework as its backbone for policy optimization, which inherently suffers from over-conservative action-level constraints, particularly in small dataset settings. In contrast, TELS performs policy optimization entirely within the compact and generalizable latent state space derived from TS-IDM, enabling state-level optimization that avoids the limitations of action-space constraints.

Table 4: Normalized scores on full-size D4RL datasets (averaged over the final 10 evaluations with 5 seeds).

Task	BC	TD3+BC	MOPO	COMBO	CQL	IQL	DOGE	IDQL	POR	TSRL	TELS (ours)
Hopper-m	52.9	59.3	28.0	97.2	58.5	66.3	**98.6 ± 2.1**	63.1	78.6 ± 7.2	86.7±8.7	94.3 ± 2.8
Hopper-mr	18.1	60.9	67.5	89.5	95.0	94.7	76.2±17.7	82.4	**98.9 ± 2.1**	78.7±28.1	**99.5 ± 2.3**
Hopper-me	52.5	98.0	23.7	**111.1**	**105.4**	91.5	102.7± 5.2	105.3	90.0 ± 12.1	95.9±18.4	105.4 ± 8.5
Halfcheetah-m	42.6	48.3	42.3	**54.2**	44.0	47.4	45.3± 0.6	**49.7**	48.8 ± 0.5	48.2 ±0.7	44.3 ± 0.4
Halfcheetah-mr	**55.2**	44.6	53.1	55.1	45.5	44.2	42.8 ±0.6	45.1	43.5±0.9	42.2 ± 3.5	41.1 ± 0.1
Halfcheetah-me	55.2	90.7	63.3	90.0	91.6	86.7	78.7±8.4	94.4	**94.7±2.2**	92.0±1.6	87.1 ± 2.9
Walker2d-m	75.3	83.7	17.8	81.9	72.5	78.3	**86.8 ± 0.8**	80.2	81.1 ± 2.3	77.5 ±4.5	81.3± 5.1
Walker2d-mr	26.0	81.8	39.0	56.0	77.2	73.9	**87.3 ± 2.3**	79.8	76.6 ± 6.9	66.1±12.0	**86.0 ± 3.3**
Walker2d-me	107.5	110.1	44.6	103.3	108.8	109.6	**110.4±1.5**	**111.6**	109.1 ± 0.7	109.8±3.12	**110.7 ± 1.4**
Antmaze-u	65.0	78.6	0.0	80.3	84.8	85.5	**97.0 ± 1.8**	93.8	90.6 ± 7.1	81.4 ± 19.2	94.5 ± 10.3
Antmaze-u-d	45.6	71.4	0.0	57.3	43.4	66.7	63.5 ± 9.3	62.0	71.3 ± 12.1	76.5 ± 29.7	**79.7 ± 15.3**
Antmaze-m-d	0.0	0.0	0.0	0.0	54.0±11.7	74.6±3.2	77.6±6.1	**86.6**	79.2±3.1	0.0	82.4 ± 4.5
Antmaze-m-p	0.0	0.0	0.0	0.0	65.2±4.8	70.4±5.3	80.6±6.5	83.5	84.6 ±5.6	0.0	**86.7 ± 5.7**
Antmaze-l-d	0.0	0.0	0.0	0.0	31.6±9.5	45.6±7.6	36.4 ±9.1	56.4	**73.4 ±8.5**	0.0	41.7 ± 14.2
Antmaze-l-p	0.0	0.0	0.0	0.0	18.8±15.3	43.5±4.5	48.2±8.1	57.0	58.0 ± 12.4	0.0	**60.7 ± 13.3**

Connection and differences with POR. As discussed in Section 2, while both POR and TELS share similarities in utilizing a state-stitching approach in state space for policy optimization, they exhibit the following fundamental differences:

- **Original state-space vs. latent state-space optimization:** POR relies on policy optimization in the original state space, which inherently requires sufficient state-action coverage for valid state-stitching. In contrast, TELS mitigates this limitation by constructing a compact and generalizable latent space via TS-IDM.
- **Unregularized T-symmetry vs. T-symmetry regularized policy optimization:** POR optimizes the guide-policy solely through an AWR formulation [28, 29], constraining π_g to stay close to the dataset via state-stitching as in Eq. (2), but lacks additional regularization to ensure generalizable state transitions. In contrast, TELS enforces an additional T-symmetry consistency regularization $\ell_{\text{T-sym}}$, which plays a critical role in preventing π_g from outputting problematic and non-generalizable latent next states, thereby enhancing its OOD generalizability.

Differences from model-based approaches. We emphasize that our proposed TELS framework fundamentally differs from MBRL methods [52, 35, 53, 36, 54, 2, 55]. Conventional MBRL methods prioritize learning forward dynamics models to predict future states and generate rollouts for policy learning. In contrast, our proposed TS-IDM is primarily designed for state representation learning and action extraction via inverse dynamics, rather than for data generation. Furthermore, as evidenced by Table 1, in the small-sample setting, limited data samples are insufficient for the model-based approach to learn an accurate dynamics model, causing high approximation errors during model rollouts, which significantly deteriorates policy learning performance.

B Additional Results

B.1 Evaluation on the full datasets

We also evaluate the performance of TELS on the original full datasets of D4RL tasks, and the results are presented in Table 4. Our proposed method achieves comparable or better performance than existing offline RL methods. Note that although TSRL also adopts a similar T-symmetry regularized representation learning scheme as ours, it performs poorly in Antmaze medium and large datasets. Primarily due to its use of the conservative TD3+BC backbone for policy optimization.

Moreover, we observe that as the dataset size increases and its state-action space coverage broadens, the stringent T-symmetry regularization in the TS-IDM can be proportionally reduced. Since the dataset can provide enough information for policy learning, it relieves the need to extract fundamental

Table 5: TELS performance on 10k datasets across various TS-IDM with different β.

	$\beta = 10$	$\beta = 1$	$\beta = 0.1$
Hopper-m	77.3 ± 5.4	**77.3 ± 10.7**	61.4 ± 5.6
Hopper-mr	15.3 ± 6.6	**43.2 ± 3.5**	19.7 ±3.4
Hopper-me	37.6 ± 17.9	**100.9 ± 6.8**	64.7 ± 3.3
Halfcheetah-m	32.9 ± 2.3	40.8 ± 0.6	**41.2 ± 1.1**
Halfcheetah-mr	8.6 ± 1.8	33.2 ± 1.0	**34.0 ± 2.2**
Halfcheetah-me	7.5 ± 2.2	**40.7 ±1.2**	39.5 ± 2.1
Walker2d-m	37.2 ± 7.9	**62.4 ± 5.3**	54.6 ± 8.2
Walker2d-mr	17.1±2.9	**54.8 ± 6.0**	39.2 ± 8.6
Walker2d-me	20.4 ± 10.4	**87.4 ± 13.3**	44.7 ± 9.8

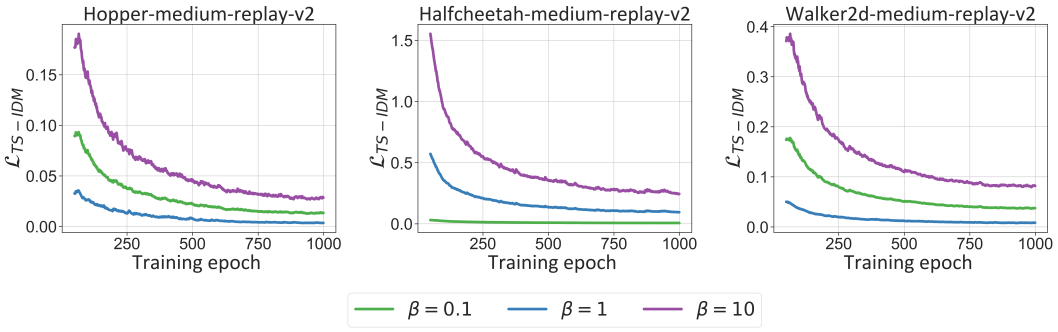


Figure 7: The learning curves for training TS-IDM on 10k dataset with different β hyperparameter.

features within the data. Consequently, we balance this trade-off by prioritizing model expressiveness over strict generalization guarantees (i.e., deploying a lower β in Eq. (7)). For instance, in the Antmaze full dataset setting, we use the regularization hyperparameter $\beta = 0.01$ to train the TS-IDM. Additional ablation studies analyzing the impact of β are detailed in Appendix B.3.

B.2 Additional OOD generalizability validation experiments

We further investigate the generalization capabilities of DOGE [15], IDQL [37], and TSRL [16] under the variation deletion degrees in the Antmaze environment. Specifically, we train each algorithm on the modified dataset after the deletion operation. We then evaluate their behaviors by visualizing rollouts over 20 evaluation episodes.

As illustrated in Figure 8, only IDQL occasionally succeeds in reaching the goal under the 0% deletion setting, while both DOGE and TSRL fail consistently. As the deletion ratio increases to 70% and 100%, none of the three methods achieves meaningful policy learning. These results highlight the inherent challenges of this setting, which requires both a compact yet expressive latent representation space and a highly generalizable policy capable of operating with extremely sparse and limited data. While TSRL integrates TDM to distill underlying patterns from the dataset, the scarcity of available data undermines its action-level constraints approach, preventing it from deriving a viable policy.

B.3 Additional ablation experiments

Impact of T-symmetry regularization on TS-IDM. To investigate the impact of T-symmetry regularization strength controlled by the hyperparameter β in Eq. (7), we conduct additional ablation experiments by varying the value of β to assess how T-symmetry regularization influences the representation learning quality and downstream policy's performance. Specifically, we train TS-IDM on reduced-size 10k D4RL MuJoCo datasets with $\beta = \{0.1, 1, 10\}$, representing different T-symmetry regularization strengths. The learning curves of TS-IDM's overall learning loss "$\mathcal{L}_{\text{TS-IDM}}$"

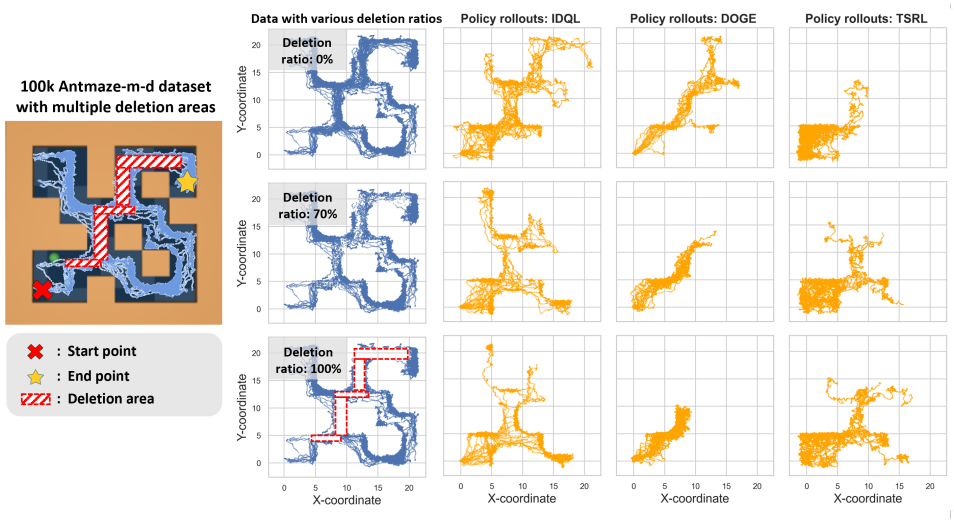


Figure 8: **Left:** Illustration of the 100k Antmaze-m-d task with multiple deletion areas, where the red cross denotes the start point, the yellow star denotes the goal locations, and the red shaded areas denote the data deletion regions. **Right:** Visualization of the training dataset and policy rollout trajectories generated by trained policies from various algorithms under varying deletion ratios.

in Eq. (7) are presented in Figure 7. The final policy learning performances with different TS-IDM models are presented in Table 5.

From Figure 7, we observe that choosing a proper β value impacts the learning quality of TS-IDM. A large β (e.g., $\beta = 10$) could impose overly strong regularization and hurt model expressiveness, which is reflected in the high learning loss at convergence. However, when the regularization strength is lowered, maintaining a proper scale of β is important to ensure both the quality and generalizability of the learned representations. As we can see in Figure 7, in the Hopper and Walker2d tasks, choosing $\beta = 1$ provides the lowest "$\mathcal{L}_{\text{TS-IDM}}$" loss; whereas in the Halfcheetah task, "$\mathcal{L}_{\text{TS-IDM}}$" is the lowest when choosing $\beta = 0.1$. If we check the final policy's performance under different TS-IDMs in Table 5, we can see a clear correlation with what we have observed in Figure 7. TELS achieves the highest score on Hopper and Walker2d tasks when $\beta = 1$, but the scores are higher for Halfcheetah tasks when $\beta = 0.1$. This matches exactly with the learning performance of TS-IDM under different β values. The strong correlation between TS-IDM's learning performance and the final policy's performance of TELS shows that we can select the best β hyperparameter values by simply looking at TS-IDM's training loss and using the one that provides the lowest training loss. This avoids the need to perform potentially unsafe online policy evaluations or unstable offline policy evaluations, which is favorable in real-world deployments.

Impact of regularizer terms η in policy optimization. The hyperparameter η governs the strength of regularization in TELS, balancing exploration and adherence to dataset states during policy updates. To evaluate the robustness of TELS, we test multiple η values ($\eta = \{1, 5, 10\}$) to examine its sensitivity to the state-level behavioral constraint in Eq. (9). Higher η values impose stronger constraints on the guide-policy, requiring generated states s' to align closely with dataset states. As shown in Figure 9, TELS demonstrates consistent robustness across η settings, achieving reliable performance under varying constraint strengths.

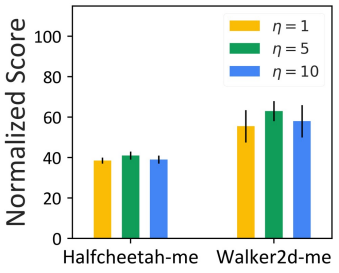


Figure 9: TELS with various η.

Impact of each component in TS-IDM for policy optimization. To validate the impact of the T-symmetry regularizer $\ell_{\text{T-sym}}$ in Eq. (10), we conduct ablation studies on 100k-sample Antmaze tasks. From the evaluation results presented in Table 6, the naïve auto-encoder based inverse dynamics model "$\phi/\psi + h_{inv}$" fails to form a reasonable latent space, yielding 0 average normalized scores across all Antmaze tasks. The introduction of latent dynamics models "h_{fwd}" and "h_{rvs}" provides marginal improvements by capturing partial system dynamics yet remains insufficient for effective policy

Table 6: Ablations on the components of TS-IDM in Antmaze tasks.

	Antmaze-m-d	Antmaze-m-p	Antmaze-l-d	Antmaze-l-p
$\phi/\psi + h_{inv}$	0	0	0	0
$\uparrow + h_{fwd}, h_{rvs}$	23.6 ± 18.4	30.4 ± 9.3	14.4 ± 5.6	7.8 ± 3.4
$\uparrow + \ell_{\text{ode}}$	34.1 ± 15.7	48.7 ± 13.3	20.1 ± 8.9	22.6 ± 16.7
$\uparrow + \ell_{\text{T-sym}}$	$\mathbf{47.2 \pm 17.3}$	$\mathbf{62.9 \pm 17.8}$	$\mathbf{39.8 \pm 14.1}$	$\mathbf{47.3 \pm 13.1}$

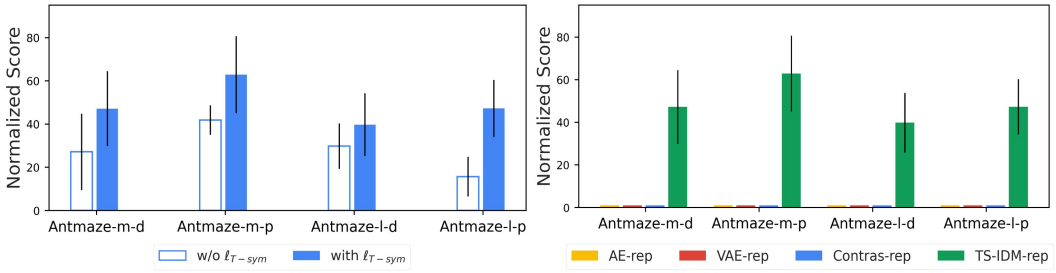


Figure 10: **Left:** Impact of $\ell_{\text{T-sym}}$ on policy optimization with 100k Antmaze datasets. **Right:** Performance of TELS with different representation models on Antmaze 100k datasets.

learning. Notably, enforcing ODE properties on decoders and applying T-symmetry consistency emerge as the most significant factors driving performance improvements, substantially enhancing the reliability of learned representations for downstream guide-policy optimization.

Impact of T-symmetry regularizer term in stochastic policy optimization. We further conduct ablation experiments in Figure 10 (left) to validate the effectiveness of the T-symmetry consistency regularization term $\ell_{\text{T-sym}}$ during the stochastic guide-policy optimization process of TELS. The results demonstrate that in stochastic policy optimization schemes, integrating this term significantly improves performance while reducing variance, underscoring the critical role of T-symmetry consistency regularization in enhancing OOD generalization and training stability.

Effectiveness of learned representations for stochastic policy optimization. As illustrated in Figure 10 (right), we evaluate TELS across diverse representation learning approaches in Antmaze tasks. The results demonstrate that baseline models struggle to construct meaningful latent spaces as task complexity increases and data scarcity intensifies (with only 100k usable samples). In contrast, TS-IDM uniquely learns a compact, well-structured latent space that remains informative and generalizable, providing a more reliable latent space for policy learning.

C Implementation Details

C.1 Implementation details of TS-IDM

- **Network structure.** For all MuJoCo locomotion and Antmaze tasks, we deployed 3-layer feed-forward neural networks for the state encoder ϕ_s, latent inverse dynamics model h_{inv}, forward and reverse dynamics models h_{fwd} and h_{rvs}, and decoder models ψ_s and ψ_a for the latent states and actions. The activation function is ReLU and uses Adam optimizer to update the parameters. We present the hyperparameters details of training TS-IDM in Table 7, including the details of the structure we have implemented as well as the deployed hyperparameters.
- **ODE property enforcement on ϕ_s and ψ_s.** We adopt a similar approach to TSRL [16] to train the ODE enforced forward and reverse dynamic models. Specifically, we compute the time-derivative of the state encoder $\phi_s(s)$ by calculating its jacobian matrix through `vmap()` function in Functorch [2]. This allows us to derive the supervision values $\frac{d\phi_s(s)}{ds} \cdot \dot{s}$ and $\frac{d\phi_s(s')}{ds'} \cdot (-\dot{s})$ for the forward dynamics model and reverse dynamics model respectively as in Eq. (4). This approach implicitly enforces the ODE property on the state encoder ϕ_s as the encoder is required to produce state representations that satisfy the ODE constraints. Unlike TSRL, which enforces ODE properties only on the encoders and not on the decoders, our method further regularizes the

[2] https://pytorch.org/functorch/stable/functorch.html

Table 7: Hyperparameters of TS-IDM.

	Hyperparameters	Value
TS-IDM Architecture	State encoder hidden units	512×256
	State encoder activation function	ReLU
	Latent forward model hidden units	256×256
	Latent forward model activation function	ReLU
	Latent reverse model hidden units	256×256
	Latent reverse model activation function	ReLU
	latent inverse model hidden units	1024×1024
	Latent inverse model activation function	ReLU
	Latent inverse model dropout	True
	Latent inverse model dropout rate	0.1
	State decoder hidden units	256×512
	State decoder activation function	ReLU
	Action decoder hidden units	512×512
	Action decoder activation function	ReLU
Hyperparameters	Optimizer type	Adam
	Weight of ℓ_{rec}	1
	Learning rate	3e-4
	Batch size	256
	Training epoch	1000
	State normalize	True
	Weight of β	1 (MuJoCo locomotion 10k setting) 0.1 (MuJoCo Antmaze 10k&100k setting) 0.1 (MuJoCo locomotion full dataset setting) 0.01 (MuJoCo Antmaze full dataset setting) 0.01 (MuJoCo adroit-human task) 0.01 (Real-world DC cooling control testbed task)
	Weight decay	0 (MuJoCo locomotion 10k setting) 1e-5 (MuJoCo locomotion&Antmaze full dataset setting) 1e-5 (MuJoCo adroit-human tasks) 1e-5 (Real-world DC cooling control testbed task)

Table 8: Structure and training parameters of guide-policy optimization.

	Hyperparameters	Value
Guide-policy structure	Value network hidden units	1024×1024
	Value network activation function	ReLU
	Policy network hidden units	1024×1024
	Policy network hidden units	ReLU
Training Perparameters	Optimizer type	Adam
	Target Value network moving average	0.05
	Batch size	256
	Training steps	100000
	State normalize	True

state decoder ψ_s. Specifically, ψ_s is trained to decode the predicted latent state variables generated by $h_{fwd}(z_s, z_a) = \dot{z}_s$ and $h_{rvs}(z_{s'}, z_a) = -\dot{z}_s$ ensuring that it also satisfies the ODE constraints in Eq. (5). To achieve this, we apply the same approach to compute $\frac{d\psi_s(z_s)}{dt}$ and train the state decoder accordingly.

C.2 Implementation details of T-symmetry regularized guide-policy

- **Network structure.** For all D4RL MuJoCo-v2 and Antmaze-v1 tasks, we deployed 2-layer feed-forward neural networks for the guide-policy π_g and the value function V. The activation function is ReLU and uses Adam optimizer to update the parameters. The parameter details are presented in Table 8.

Algorithm 1 Offline RL via T-symmetry Enforced Latent State-Stitching (TELS).

Require: Offline dataset $\mathcal{D}$.
1: *// TS-IDM learning*
2: Learning the state encoder ϕ_s, state decoder ψ_s, action decoder ψ_a, latent inverse dynamics h_{inv}, latent forward and reverse dynamics h_{fwd} and h_{rvs} using the TS-IDM learning objective Eq. (7).
3: Initialize V_θ, $V_{\theta'}$, π_σ
4: *// Policy training*
5: **for** $t = 1, \cdots, M$ training steps **do**
6: Sample transitions $(s, r, s') \sim \mathcal{D}$ and compute their representations $(z_s, z_{s'})$ using the state encoder ϕ_s.
7: Use $(z_s, r, z_{s'})$ to update the latent state-value function V using Eq.(8).
8: Use $(z_s, z_{s'})$ to update the latent guide-policy π_g using Eq. (9) or (10).
9: **end for**
10: *// Evaluation*
11: Get initial state s from environment
12: **while** not done **do**
13: Get optimized next state $z^*_{s'}$ using guide-policy π_g.
14: Extract action a using Eq. (11).
15: **end while**

- **Hyperparameters for policy optimization.** Under both small-sample and full datasets settings, we employ a deterministic policy update strategy for MuJoCo locomotion tasks, as defined in Eq. (9), with learning rates of 1e-4 for both value and policy functions. The normalization term λ is computed as $\lambda_\alpha = \alpha/[\sum_{s_i} |V(\phi_s(s_i))|/N]$, where α controls the trade-off between value maximization and policy regularization and N denotes the number of samples in the training batch. For Antmaze tasks, we utilize a stochastic policy optimization strategy, as outlined in Eq. (10), with learning rates of 1e-3 for value and policy functions.

  Full dataset setting: We set $(\tau, \alpha, \eta) = (0.7, 0.01, 10)$ for all MuJoCo locomotion tasks and Adorit tasks, for all MuJoCo Antmaze tasks, we deploy $(\tau, \alpha) = (0.9, 10)$ as the training parameters.

  Small-sample setting: For Halfcheetah and Walker2d tasks, we set $(\tau, \alpha, \eta) = (0.5, 0.01, 5)$ and incorporate policy dropout to mitigate overfitting. These tasks share identical state and action dimensions (17 states and 6 actions), enabling the use of the same parameter set for guide-policy training. In contrast, Hopper tasks with a smaller state-action space (11 states and 3 actions) are comparatively simpler given the same amount of training data (e.g., 10k samples). Consequently, we adopt a more aggressive learning strategy for Hopper, setting $(\tau, \alpha, \eta) = (0.7, 0.1, 10)$ to prioritize value maximization. For Antmaze tasks, we use an identical set of parameters $(\tau, \alpha) = (0.9, 10)$ as in the full dataset setting to train the guide-policy. For the real-world DC cooling control testbed task, we find using the $(\tau, \alpha, \eta) = (0.5, 0.01, 5)$ can derive the best performance.

Training resources. To train a TS-IDM, we utilize one NVIDIA GeForce RTX 4090 with an AMD Ryzen 9 7950X 16-Core Processor and 16GB of memory for approximately 30 minutes, running on Ubuntu 22.04.2 LTS 64-bit. We employ the same resource configurations for approximately 6 hours for the guide-policy training.

D Detailed Experiment Setups

Reduced-size dataset generation. To create reasonably reduced-size D4RL datasets for a fair comparison, we use the identical small samples as in the TSRL paper [16] for the locomotion tasks training. For Antmaze tasks, we adopt a similar approach by randomly sub-sampling trajectories from the original dataset to construct smaller training datasets. Specifically, for the "Antmaze-umaze" tasks, we randomly sample 10k data points for training, and for the "Antmaze-medium" and "Antmaze-large" tasks, we utilize 100k random samples as the training dataset of TELS.

The rationale behind this adjustment is the "medium" and "large" environments are significantly more expansive than the "umaze" environment. Sampling only 10k data points would likely result in trajectories that lack the fundamental information necessary to describe the task. Therefore, we relax the small-sample constraints for these environments to ensure that the reduced datasets at least contain enough successful trajectories for effective training.

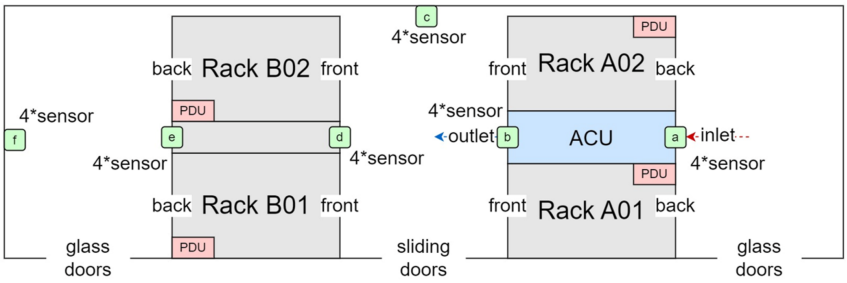


Figure 11: The layout illustration of the real-world DC cooling control testbed environment (figure from [38]).

Experiment setups for OOD generalization tasks in Antmaze. In Section 4.2, we conduct a more challenging scenario to verify the OOD generalizability of the algorithm. Specifically, based on the 100k "Antmaze-medium-diverse-v2" dataset, we manually selected five critical intervals and erased the data points within these intervals by randomly deleting them. The selection of intervals was determined based on the XY-axis coordinates. In this dataset, the first two dimensions of the state represent the vertical and horizontal coordinates, respectively. Based on this information, we randomly deleted 70% and 100% of the data in the chosen intervals. We then trained IQL [12], DOGE [15], IDQL [37], POR [20], TSRL [16], and TELS using this modified dataset to evaluate their performance.

Experiment details of real-world industrial control test environment. We adapted the figure from [3] to illustrate the layout structure of the real-world DC cooling control testbed. As shown in the figure D, the testbed comprises 22 server units and an inter-rack air conditioning unit (ACU) positioned between Rack 1 and Rack 2, supplemented by 24 temperature and humidity sensors (organized into six monitoring sets) to capture spatial thermal dynamics within the environment. Notably, the ACU employs compressor-driven cooling, with fan operation and compressor workload constituting the primary sources of energy expenditure. The thermal regulation is achieved by modulating the ACU's entering air temperature (EAT) setpoint to maintain the rack exhaust temperature (CAT) below a predefined safety threshold. The energy-saving objective is to improve the energy efficiency of the DC's cooling systems (minimizing the ACLF) while satisfying thermal safety constraints.

We leverage a dataset of 43k real-world operational samples recorded at 2-minute intervals over 61 days with 105 state-action features. During the training process, we utilize the identical reward function and follow the same experimental protocols outlined in [3]. To ensure rigorous benchmarking, we adopt the same challenging thermal constraint (set the CAT threshold as 22°C) for comparative evaluation of TELS performance.

Experiment setups for various representation learning. To validate the effectiveness of the representations learned by TS-IDM, we integrate it as the representation module in two offline RL frameworks (IQL and TD3+BC), verifying the usability of the learned latent space as illustrated in Figure 4 (left). Specifically, we process the original states s and next states s' from the dataset using the pre-trained state encoder ϕ_s of TS-IDM to derive the latent representations: $\phi_s(s) \to z_s$ and $\phi_s(s') \to z_{s'}$. Then, train IQL and TD3+BC within the latent space to evaluate their performance under the small-sample setting.

Furthermore, in Figure 4(right), we benchmark TELS against three established representation learning baselines ("AE-rep", "Contras-rep" and "VAE-rep") to rigorously assess TS-IDM's representation quality. Implementation details for all baseline models are provided below:

- **"AE-rep"**: We implement a naïve autoencoder-based inverse dynamics framework, consisting of a state encoder and decoders ϕ_s and ψ_s to construct the latent state space. As in TELS, the inverse dynamics model h_{inv} is built within this latent space, serving as the execute-policy. For a fair comparison, we use the same network parameters for the encoder, decoder, and inverse dynamics model as in TS-IDM. The "AE-rep" model is trained with a reconstruction loss to capture the essential features of the input, and the inverse dynamics model is simultaneously trained on the latent representations to predict actions.
- **"VAE-rep"**: The variational autoencoder (VAE) [39] is built based on the "AE-rep" model by introducing additional KL divergence loss terms. Specifically, the encoder outputs parameters of a Gaussian distribution in the latent space, and the latent representations are sampled using the

reparameterization trick. The VAE is trained using a combined loss function that includes both the reconstruction loss and the KL divergence loss, which regularizes the latent space to follow a prior distribution. The inverse dynamic model is trained simultaneously with the VAE, sharing the latent space and optimizing for both the reconstruction of the input data and the prediction of actions.

- **"Contras-rep"**: We utilize the NT-Xent loss (Normalized Temperature-Scaled Cross Entropy Loss) used in SimCLR [40] within the latent representation space on top of the "AE-rep" model. The overall loss function combines the contrastive loss with the reconstruction loss, ensuring that the latent space not only captures the structure of the data but also learns semantically meaningful representations that are robust to variations. The inverse dynamic model is trained simultaneously within the latent space to predict actions.

E Border Impact

While training reinforcement learning (RL) agents on large-scale offline datasets has been extensively studied, real-world applications often face prohibitive data scarcity and collection costs. This necessitates offline RL methods that achieve reliable performance in small-sample regimes. To address this challenge, we introduce a highly sample-efficient offline RL algorithm to learn high-performing policies from extremely limited data. We empirically validate its efficacy through deployment on a real-world data center cooling control testbed, establishing its practical viability. Our approach highlights a promising pathway for advancing sample-efficient offline RL in resource-constrained settings. A potential limitation is the inherent risk of unreliable or unsafe actions within historical datasets, which may mislead policy learning.

F Learning Curves

The following are the learning curves of TS-IDM and the T-symmetry regularized guide-policy optimization in TELS on the reduced-size D4RL MuJoCo and Antmaze datasets. We evaluate the policy with 10 episodes over 5 random seeds.

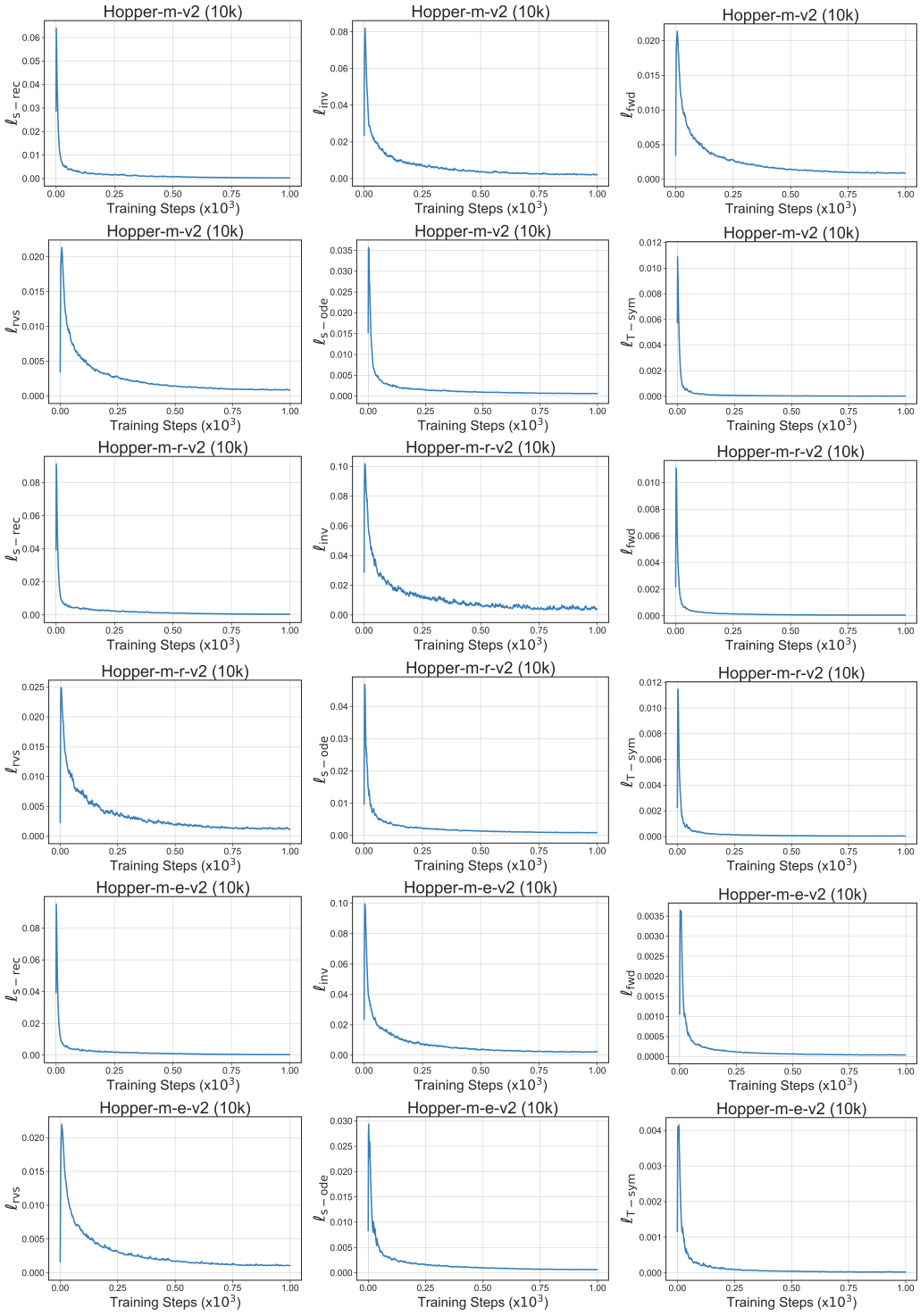


Figure 12: Learning curves of the overall and each individual loss terms in TS-IDM for Hopper tasks.

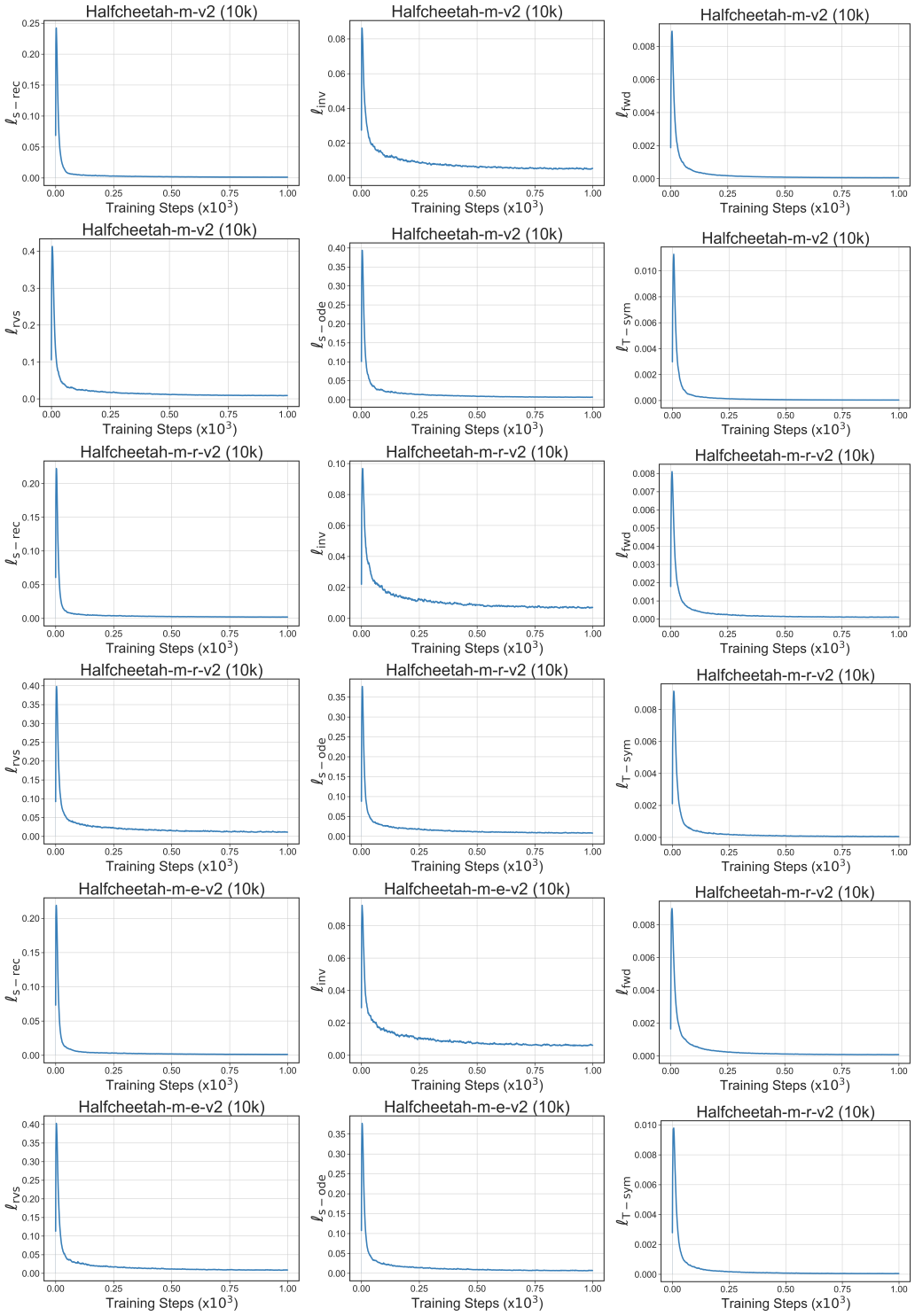


Figure 13: Learning curves of the overall and each individual loss terms in TS-IDM for Halfcheetah tasks.

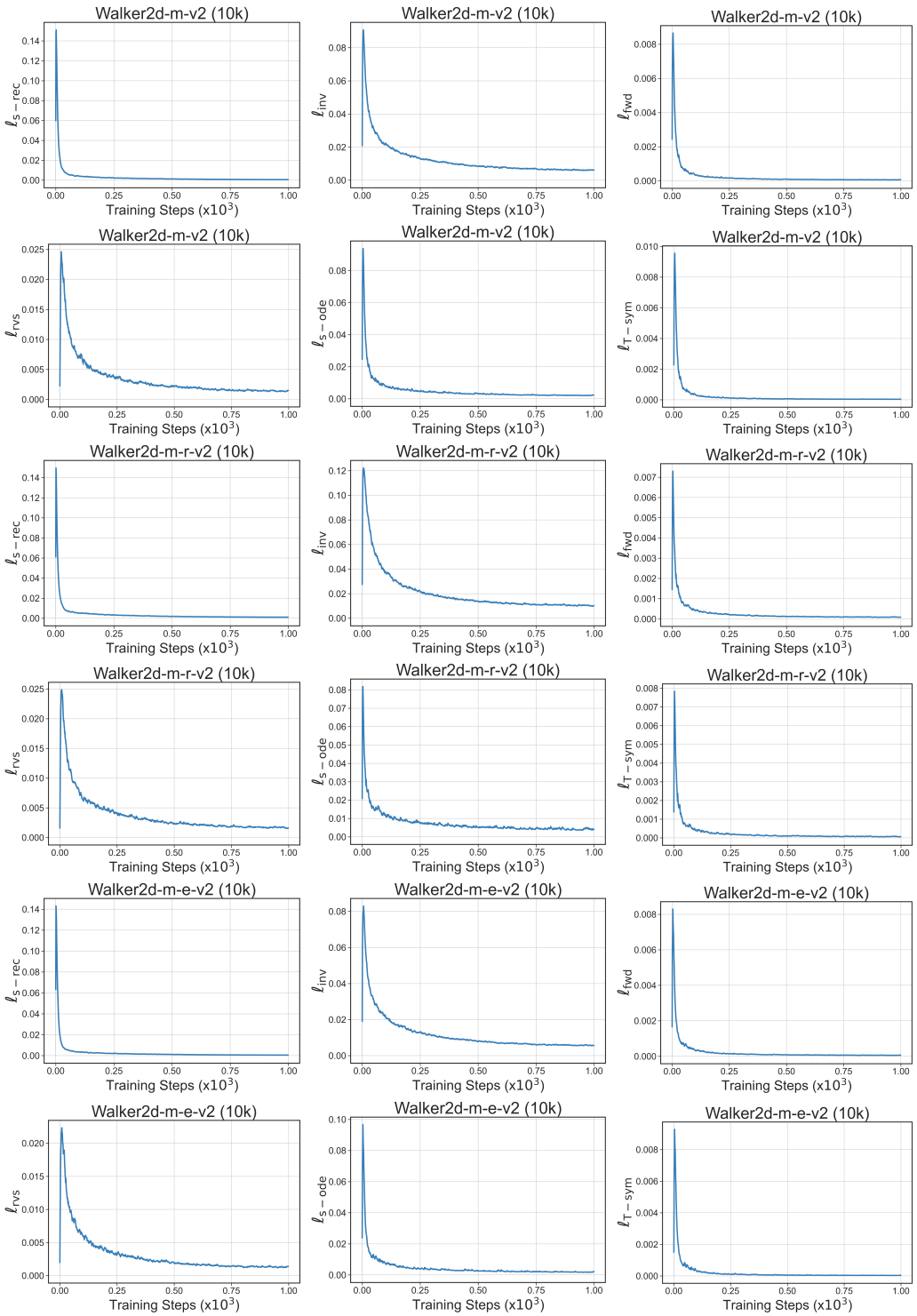


Figure 14: Learning curves of the overall and each individual loss terms in TS-IDM for Walker2d tasks.

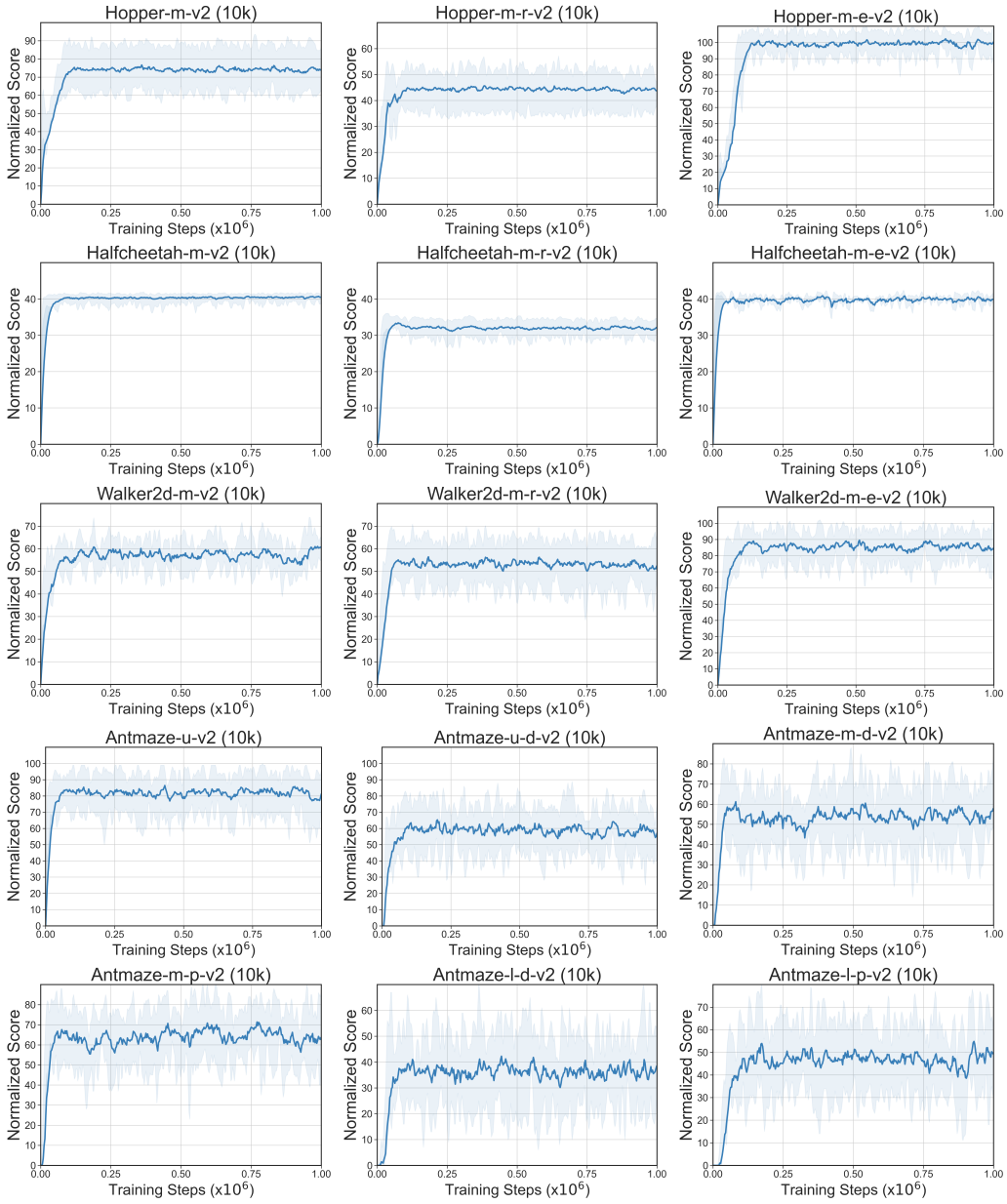


Figure 15: Learning curves of policy optimization in TELS for D4RL MuJoCo and Antmaze tasks with reduced-size datasets. We evaluate the policy within 10 episodes over 5 random seeds.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We introduce the contributions and scope in the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discussed the limitations of our work in Section 6.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: This is not a theoretical paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We explained our problem settings in Section 4 and implementation details in Appendix C.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: For anonymity reasons, we have not made our code public. Upon acceptance, we will release our code on GitHub.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: All the training and model details are specified in Appendix C and the experiment setup details are presented in Appendix D.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We presented the standard error in the reported results and learning curves, with an average of over five random seeds.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have presented the training resources requirement in Appendix C.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We read and followed the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss both potential positive societal impacts and negative societal impacts of the work in Appendix E.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: N/A

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification: N/A

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: N/A

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: N/A

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: N/A

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: N/A

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.