NASDAQ: Normalized Observation Space Dynamics-Augmented Q-Learning

Xinwei Liu[1] **Junyuan Liang**[1*] **Zicong Hong**[2] **Jianting Zhang**[3] **Wuhui Chen**[1]
[1]Sun Yat-sen University, China
[2]EPFL, Switzerland
[3]Purdue University, USA
{liuxw73, liangjy53}@mail2.sysu.edu.cn

Abstract

Augmenting model-free reinforcement learning (RL) with representations learned through observation dynamics prediction (observation-predictive RL) can improve sample efficiency and performance, with minor modifications and limited additional computation. However, this approach still struggles in challenging tasks with low-dimensional observations. In this paper, we identify a key factor behind this problem: unbalanced reconstruction losses across observation dimensions, where dimensions with larger value ranges dominate the loss. This encourages the agent to neglect dimensions with relatively small ranges, leading to degraded performance. To address this issue, we propose a novel normalization method tailored to online RL, which normalizes low-dimensional observations and balances the resulting losses and gradients. Beyond balancing reconstruction losses, observation normalization enables dynamics prediction to be performed in a normalized observation space, thereby providing a unified treatment of low- and high-dimensional inputs (e.g., physical states and images). Building on this idea, we further introduce Normalized Observation Space Dynamics-Augmented Q-learning (NASDAQ), a framework for observation-predictive RL applicable across diverse domains. NASDAQ learns state-action representations by coupling value learning with two auxiliary tasks: short-term value prediction and next *normalized* observation prediction. Extensive experiments demonstrate that NASDAQ achieves competitive or superior performance compared with state-of-the-art model-based and self-predictive RL methods, while requiring significantly less training wall-time. Our implementation is publicly available at `https://github.com/sumwailiu/NASDAQ`.

1 Introduction

Reinforcement learning (RL) [1] has proven to be a powerful framework for solving various sequential decision-making problems. However, model-free RL [2–4] is notoriously sample inefficient. Model-based RL improves sample efficiency and performance by introducing world models [5, 6] and planning [7–9], but incurs substantial computational costs. Augmenting model-free RL with representations learned through observation dynamics prediction [10–12], which we call observation-predictive RL, can also improve sample efficiency and performance, with less algorithmic and computational complexity. However, these approaches still struggle with challenging low-dimensional control tasks. In contrast, self-predictive RL [13–17], which learns representations by predicting latent embeddings of future observations instead, has recently shown promising results across a broader range of domains [17]. This discrepancy raises the following questions:

1. What is the bottleneck in observation-predictive RL?

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

2. Is it possible to enable observation-predictive RL to match or surpass model-based and self-predictive RL across diverse domains, while maintaining high computational efficiency?

To answer these questions, we start by examining the training metrics of OFENet [12], a representative observation-predictive RL method, and identify the cause of its limited performance in challenging low-dimensional tasks. The root cause lies in unbalanced reconstruction losses across observation dimensions arising from differences in their value ranges. Consequently, the gradients induced by large losses dominate the overall gradient, encouraging the learned representations to neglect dimensions with relatively small ranges. To address this issue, we propose Shift-Adaptive Robust Observation Normalization (SARON), a novel normalization method for low-dimensional settings. SARON is tailored to online RL, accounting for distributional shifts and distributional variability induced by exploration. It normalizes streaming observations and balances the resulting reconstruction losses and gradients across dimensions.

Beyond mitigating this imbalance, observation normalization allows dynamics prediction to be performed in a normalized observation space, thereby providing a unified treatment of low- and high-dimensional observations (e.g., physical states and images). Building on this insight, we introduce Normalized Observation Space Dynamics-Augmented Q-learning (NASDAQ), a unified framework for observation-predictive RL applicable across diverse domains. Unlike complex self-predictive methods [14–17] that iteratively predict latent embeddings of future observations over extended horizons, NASDAQ learns state-action representations by combining value learning with two auxiliary tasks: short-term value prediction and next *normalized* observation prediction.

We evaluate NASDAQ (with SARON for low-dimensional settings) across 87 environments from four widely used benchmarks. Extensive experiments demonstrate that our method achieves competitive or superior performance compared with state-of-the-art model-based [6, 8] and self-predictive approaches [15–17], while requiring substantially less training wall-time. Interestingly, even without auxiliary tasks, NASDAQ with SARON remains a strong model-free RL baseline, trailing the state-of-the-art self-predictive method only slightly in low-dimensional settings. We hope this work can inspire further research on observation-predictive and model-free methods.

2 Related work

Data-efficient RL A large body of work aims to improve sample efficiency and performance via dynamics modeling. Model-based methods incorporate world models into RL [5–9]. While these methods achieve impressive performance, they incur significant algorithmic and computational complexity. An alternative line of work [10–17] augments model-free RL with representations learned through dynamics prediction without explicitly using world models. We refer to this class of approaches as *dynamics-augmented methods*, which can be divided into self-predictive RL [14–17] and observation-predictive RL [10–13]. Self-predictive RL learns representations by predicting latent embeddings of future observations. Recently, the self-predictive method MR.Q [17] has achieved performance competitive with model-based approaches, DreamerV3 [6] and TD-MPC2 [8], across diverse domains. Inspired by model-based methods, MR.Q also includes model-based objectives, such as reward and termination predictions, in its representation learning. Observation-predictive RL performs dynamics modeling directly in the observation space. Our framework, NASDAQ, falls into this category. The most related work to ours is OFENet [12], which learns state-action representations in a decoupled manner for downstream RL by predicting the next *raw* observation. However, OFENet is designed for low-dimensional inputs and struggles in challenging tasks. In contrast, NASDAQ adopts coupled representation and value learning and addresses this bottleneck via a novel observation normalization method, while being broadly applicable across domains.

Normalization for RL Prior studies have investigated applying batch normalization [12], layer normalization [17], spectral normalization [18, 19], CrossNorm and SelfNorm [20] in RL. These methods are applied to internal components of neural networks. While observation normalization is commonly used in visual RL [17, 21, 22], it is often overlooked for low-dimensional observations. Its effects in such settings thus remain underexplored. A simple normalization method for low-dimensional tasks is implemented in the Stable-Baselines3 (SB3) library [23]. The approach normalizes each dimension using the mean and standard deviation computed over all historical observations. In contrast, our proposed method, SARON, computes statistics by accounting for the characteristics of online RL.

Self-predictive vs. observation-predictive RL Recent work has compared the two types of approaches. Ni et al. [24] empirically show that self-predictive methods outperform observation-predictive ones in noisy or distracting tasks, whereas the opposite holds in sparse-reward tasks. Voelcker et al. [25] theoretically show that features learned via observation prediction are generally superior to those learned via latent self-prediction, but the latter are more robust to observation perturbations. While these studies identify conditions under which observation-predictive RL is weak, they do not propose methods to address them. In this work, we empirically identify a key factor limiting observation-predictive RL and propose a solution.

3 Background

RL problems are formulated as Markov Decision Processes (MDPs). An MDP can be described as a tuple $(\mathcal{S}, \mathcal{A}, p, R, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ respectively denote the state and action spaces, $p(s_{t+1}|s_t, a_t)$ is the transition dynamics, R is a reward function, and γ is a discount factor. The objective is to find a policy $\pi : \mathcal{S} \to \mathcal{A}$ that maximizes $\sum_{t=0}^{\infty} \gamma^t r_t$, the cumulative discounted return when following the policy. Value-based RL methods learn a value function $Q^\pi(s, a) := \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s, a_0 = a]$ that models the expected return starting from an initial state s and action a.

4 Understanding the bottleneck in observation-predictive RL

In this section, we identify a key bottleneck in OFENet [12], a strong observation-predictive RL method that learns state-action representations separately from downstream model-free RL algorithms (e.g., TD3 [4], PPO [3], and SAC [26]). We consider the OFENet+TD3 variant in our experiments. The method is evaluated on a compact diagnostic set of six low-dimensional tasks, comprising five OpenAI Gym tasks [27] used in OFENet and the *dog-run* task from the DeepMind Control suite (DMC) [28]. Experimental setup and implementation details are deferred to Appendix B.1 and B.4, respectively.

Table 1: Final performance and training metrics of OFENet+TD3 at the end of training over 5 seeds. The [bracketed values] represent a 95% bootstrap confidence interval. The performance of MR.Q is included for comparison. **Bold numbers** indicate better results.

Tasks	MR.Q	OFENet+TD3	Auxiliary loss
Ant-v5	7514 [6977, 7911]	**8156** [8047, 8280]	0.015 [0.013, 0.016]
Hopper-v5	2699 [2245, 3179]	**2853** [1837, 3637]	4.8e-4 [3.6, 5.9]e-4
Walker2d-v5	5431 [4793, 5998]	**6009** [5812, 6206]	0.054 [0.044, 0.067]
HalfCheetah-v5	**13823** [13412, 14251]	13548 [12867, 14309]	0.16 [0.13, 0.19]
Humanoid-v5	**7207** [5950, 8147]	6063 [5853, 6273]	139 [133, 149]
dog-run	**300** [281, 319]	66 [22, 110]	272 [130, 436]

In Table 1, we examine the auxiliary loss (observation reconstruction loss) of OFENet+TD3 and compare its performance with that of MR.Q [17], a state-of-the-art self-predictive RL method. As reported in OFENet, smaller auxiliary losses weakly correlate with better performance. Notably, auxiliary losses on challenging tasks (*Humanoid-v5* and *dog-run*) are far larger than those on the other tasks. To investigate the underlying cause, we further analyze auxiliary losses across observation dimensions. As shown in Figure 1, auxiliary losses across dimensions exhibit a skewed distribution with a heavy tail in challenging tasks. This indicates that the large auxiliary loss is primarily driven by a small subset of dimensions with disproportionately high error contributions. Moreover, the auxiliary loss distribution mirrors that of standard deviation, suggesting a positive correlation between a dimension's value range and its auxiliary loss. A quantitative assessment of this correlation is provided in Figure 10 of Appendix C.7. Taken together, these findings suggest that a key bottleneck in observation-predictive RL is unbalanced reconstruction losses across dimensions due to their value ranges. As a result, the overall gradient is dominated by gradients from large-loss dimensions, encouraging the learned representations to ignore dimensions of relatively small ranges.

To some extent, our findings are consistent with prior theoretical results that observation-predictive methods are more vulnerable than self-predictive ones when observation vectors are distracted by multiplying a random binary matrix [25]. When the learned representations are encouraged to focus

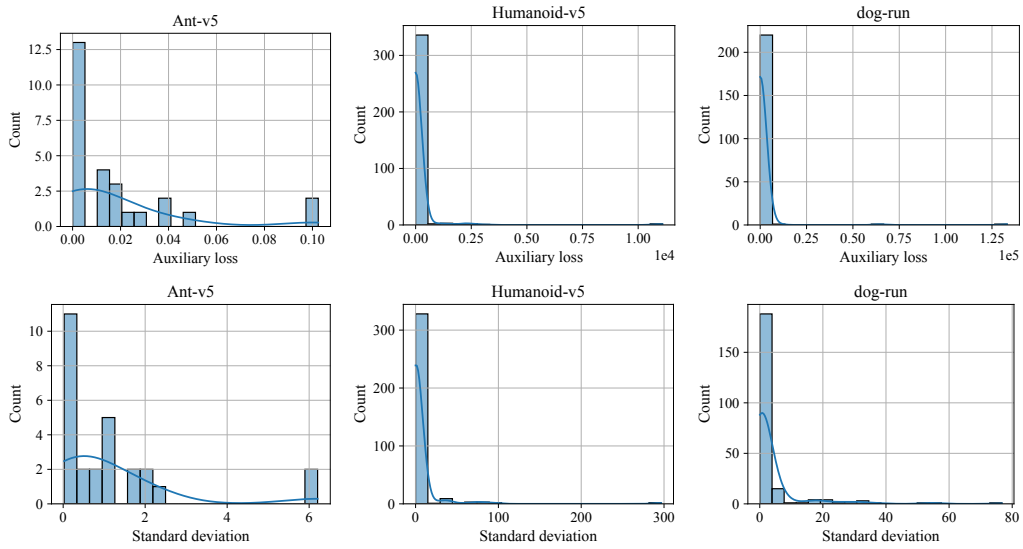


Figure 1: Histograms and kernel density–estimated PDFs of per-dimension statistics, obtained from a single run of OFENet+TD3 on three tasks. *Top*: distribution of auxiliary losses for each observation dimension, where the value for each dimension is computed by averaging the recorded auxiliary losses over the final 20k time steps. *Bottom*: distribution of per-dimension standard deviations, computed from the final 20k samples. See Appendix C.7 for results on the remaining tasks.

on a subset of observation dimensions, the effect resembles multiplying the observation vector by a diagonal binary matrix.

5 Methods

5.1 Normalization for low-dimensional observations

Motivated by the insight obtained in Section 4, we aim to balance the auxiliary losses across observation dimensions. Specifically, we normalize each dimension by its mean and standard deviation. A simple approach is to compute these statistics over all historical observations [23]. However, such a method neglects both distributional shifts during online RL and distributional variability induced by exploration. Therefore, we propose Shift-Adaptive Robust Observation Normalization (SARON), a novel normalization method designed for the online RL paradigm. SARON computes episode-wise means μ and uncentered second moments ν, and maintains their exponential moving averages (EMAs) with coefficient β to obtain the running statistics:

$$m \leftarrow \beta m + (1-\beta)\mu, \qquad s \leftarrow \beta s + (1-\beta)\nu. \tag{1}$$

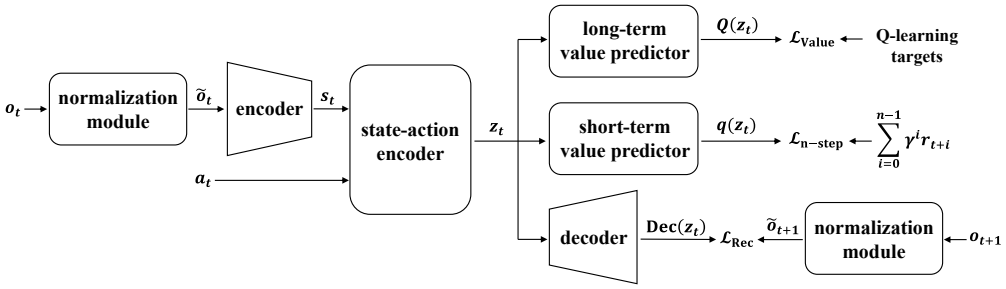
As EMA statistics (m and s) are initialized to zero, leading to biased estimates in early updates, we apply bias correction to these statistics following Adam [29]:

$$\hat{m} = \frac{m}{1-\beta^k}, \qquad \hat{s} = \frac{s}{1-\beta^k}, \tag{2}$$

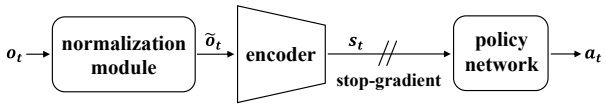
where k denotes the number of EMA updates applied so far. Raw observations o are normalized using these corrected statistics and clipped to $[-O, O]$ for numerical stability:

$$\tilde{o} = \mathrm{clip}\left(\frac{o-\hat{m}}{\hat{\sigma}}, -O, O\right), \qquad \hat{\sigma} = \sqrt{\hat{s}-\hat{m}^2}, \tag{3}$$

where $\hat{\sigma}$ is the corrected standard deviation. To obtain robust statistics from noisy online observation streams, we exclude episode-wise statistics associated with abnormally low returns from the EMA updates. These statistics, probably induced by exploration, are treated as outliers and identified using the interquartile range (IQR) over recent returns.



(a) Value network with two auxiliary prediction heads



(b) Policy network with detached inputs

Figure 2: Overview of the NASDAQ framework. (a) The value network contains three predictors for value learning (long-term value prediction), and two auxiliary tasks of predicting the short-term value and the next *normalized* observation. (b) The gradients from the policy network are not backpropagated to the observation encoder.

5.2 Normalized observation space dynamics-augmented Q-learning (NASDAQ)

Beyond balancing auxiliary losses, observation normalization enables dynamics prediction to be formulated in a normalized observation space, thereby providing a unified way to learn representations from low- and high-dimensional observations (e.g., physical states and images). Motivated by this perspective, we introduce Normalized Observation Space Dynamics-Augmented Q-learning (NASDAQ), a unified framework for observation-predictive RL applicable to diverse domains. As illustrated in Figure 2, the raw observation o is first normalized by a normalization module, e.g., min-max normalization for pixel inputs, and SARON for low-dimensional observations. The normalized observation $\tilde{o}$ is then fed to an encoder f to obtain a latent state s, which is further processed by a state-action encoder g together with the action a to produce a state-action representation z:

$$s = f(\tilde{o}), \qquad z = g(s, a). \tag{4}$$

s is fed to a policy network with gradients stopped, whereas z is fed to three predictors (Q, q, and Dec) for value learning and two auxiliary tasks.

Value learning The value target is constructed similarly to TD3 [4]. Specifically, we train two separate value networks V_1 and V_2, each with the same structure shown in Figure 2. We use numerical subscripts to refer to components within each network (e.g., the encoder f_1 of V_1). Target networks, which are used to produce stationary prediction targets, are denoted by adding a prime to their online counterparts (e.g., f_1' and V_1'). Online and target policy networks (π and π') use the detached outputs from f_1 and f_1', respectively. All target parameters are periodically synchronized with their online counterparts [16, 17]. The target action is produced by the target policy network, perturbed by clipped Gaussian noise:

$$a^{\pi'} = \begin{cases} \operatorname{argmax} a' & \text{for discrete } a' \\ \operatorname{clip}(a', -1, 1) & \text{for continuous } a' \end{cases}, \quad a' = \pi'(s') + \operatorname{clip}(\epsilon, -c, c), \quad \epsilon \sim \mathcal{N}(0, \sigma^2), \tag{5}$$

where $s' = f_1'(\tilde{o}')$. Discrete actions are represented by a one-hot encoding, with the Gaussian noise added to each dimension. Following prior work [17, 22, 30], we predict n-step returns. Given a normalized transition $\tau = (\tilde{o}_t, a_t, r_{t:t+n-1}, \tilde{o}_{t+n})$, the value target y is computed as:

$$y = \sum_{i=0}^{n-1} \gamma^i r_{t+i} + \gamma^n \min_{j \in \{1,2\}} Q_j'(z_{t+n}^j), \qquad z_{t+n}^j = g_j'\left(s_{t+n}^j, a_{t+n}^{\pi'}\right), \tag{6}$$

where $s_{t+n}^j = f_j'(\tilde{o}_{t+n})$, and $a_{t+n}^{\pi'}$ is computed based on $\tilde{o}_{t+n}$ via Equation 5. The value learning loss is computed using the Huber loss [31] instead of mean squared error (MSE) to eliminate bias from prioritized sampling [17, 32]:

$$\mathcal{L}_{\text{Value}} = \text{Huber}\left(Q_1, y\right) + \text{Huber}\left(Q_2, y\right). \tag{7}$$

Policy learning The policy network π is updated using the deterministic policy gradient [33]. Following prior work [17, 34], we add L2 regularization to the pre-activation policy outputs h^π to help avoid local minima when rewards and value estimates are sparse:

$$\mathcal{L}_{\text{Policy}} = -\frac{1}{2} \sum_{i \in \{1,2\}} Q_i(z^i) + \lambda_{\text{pre-activ}} \|h^\pi\|_2^2, \qquad a^\pi = \text{activ}(h^\pi), \tag{8}$$

where $z^i = g_i(f_i(\tilde{o}), a^\pi)$, and a^π is computed from $\tilde{o}$ using online networks and Equation 5 but without action noise. To accommodate different action spaces, the *activ* function is Tanh for continuous actions and Gumbel-SoftMax [17, 35, 36] for discrete ones.

Auxiliary tasks We introduce two auxiliary tasks in NASDAQ. The first task is short-term value prediction, where a predictor q takes state-action representations z as input and predicts the cumulative discounted reward over the next n steps. Following prior work [6, 9, 17], we optimize this task with a cross-entropy (CE) loss due to its empirical benefit in sparse-reward settings. The auxiliary loss is computed as the sum of CE losses between the predicted logits and a two-hot encoding of the target:

$$\mathcal{L}_{\text{n-step}} = \sum_{i \in \{1,2\}} \text{CE}\left(q_i(z_t^i), \text{TwoHot}(R)\right), \qquad R = \sum_{j=0}^{n-1} \gamma^j r_{t+j}, \tag{9}$$

where R is the first term of the value target in Equation 6. The support bins for the two-hot encoding are placed at non-uniform intervals induced by the transformation $\text{symexp}(x) = \text{sign}(x)(\exp(|x|) - 1)$ [6]. This auxiliary task provides training signals of lower variance than those from value estimation, thereby stabilizing representation learning when value targets exhibit high variance.

The second auxiliary task predicts the next *normalized* observation by feeding z to a decoder Dec, with MSE loss to minimize reconstruction errors:

$$\mathcal{L}_{\text{Rec}} = \sum_{i \in \{1,2\}} \left\|\text{Dec}_i(z_t^i) - \tilde{o}_{t+1}\right\|_2^2. \tag{10}$$

During training, gradients from the policy loss are not propagated to the observation encoder f_1 [12, 18, 22], whereas all components of V_1 and V_2 are jointly optimized using the combined loss $\mathcal{L}$:

$$\mathcal{L} = \mathcal{L}_{\text{Value}} + \lambda_{\text{Rec}} \mathcal{L}_{\text{Rec}} + \lambda_{\text{n-step}} \mathcal{L}_{\text{n-step}}, \tag{11}$$

where λ_{Rec} and $\lambda_{\text{n-step}}$ denote the auxiliary loss weights. Pseudocode of SARON and NASDAQ is provided in Appendix A.

6 Experiments

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

1. How does NASDAQ (with SARON) compare with state-of-the-art data-efficient RL approaches in terms of performance and computational efficiency? (See Section 6.2)
2. Can observation normalization address the bottleneck in observation-predictive RL identified in Section 4? (See Section 6.3)
3. Do the two auxiliary tasks (*normalized* observation prediction and short-term value prediction) used in NASDAQ improve RL performance? (See Section 6.4)

6.1 Experimental setup

Benchmarks We evaluate NASDAQ (with SARON for low-dimensional tasks) on four widely used RL benchmarks, spanning continuous and discrete control, as well as low- and high-dimensional observations: (1) **Gym** [27], a set of five continuous control locomotion tasks with low-dimensional observations, (2) **DMC (proprioceptive)** [28], a collection of 28 continuous control tasks with low-dimensional proprioceptive observations, (3) **DMC (visual)**, the same tasks as DMC (proprioceptive) but with image-based observations, (4) **Atari100k**, a benchmark of 26 Atari games [37] for discrete control with pixel-based inputs. See Appendix B.1 for the evaluation protocol and a complete description of the benchmarks.

RL baselines We consider the following data-efficient RL methods as baselines: (1) **OFENet+TD3** [12], an observation-predictive RL method for low-dimensional continuous control tasks, (2) **TD7** [16], a self-predictive method built on OFENet and TD3, which achieves state-of-the-art results on the Gym benchmark, (3) **SPR** [15], a self-predictive method for visual discrete control tasks, (4) **TD-MPC2** [8], a strong model-based method for continuous control tasks, (5) **DreamerV3** [6], a strong model-based RL algorithm that performs well across diverse domains, (6) **MR.Q** [17], a state-of-the-art self-predictive method that achieves performance competitive with model-based RL approaches, DreamerV3 and TD-MPC2, across diverse benchmarks, (7) **DrQ-v2** [22], a simple yet strong model-free RL baseline for visual continuous control tasks, (8) **Rainbow (enhanced)**, an improved variant of Rainbow [30] tailored for Atari100k. Details on the RL baselines are provided in Appendix B.2.

Observation normalization (ON) baseline We consider a simple observation normalization (simple ON) method implemented in SB3 [23], which normalizes observations using means and standard deviations computed over all historical observations.

Implementation overview To enable fair comparisons with other TD3-based methods (OFENet+TD3, TD7, and MR.Q), we take two control measures. First, we align the design choices of OFENet+TD3 and NASDAQ with those of MR.Q for components unrelated to representation learning (e.g., replay buffer and optimizer). Second, since these TD3-based methods share similar principles for integrating learned representations into model-free RL (whether in a coupled or decoupled manner), we control model capacity by keeping parameters of the RL components comparable across methods. Parameters dedicated to representation learning are not constrained, as they are determined by each method's design. These controls allow performance differences to be more reliably attributed to algorithmic choices rather than parameter scale or implementation discrepancies. Within each benchmark, all tasks share fixed auxiliary loss weights. A simple method for determining these weights is described in Appendix B.5. Network structures, implementation details, and the parameter control protocol are provided in Appendices B.3, B.4, and B.6, respectively.

6.2 Comparison with state-of-the-art data-efficient methods

Performance comparison As shown in Table 2, NASDAQ with SARON achieves the best performance in low-dimensional settings. For pixel-based tasks, NASDAQ outperforms other methods on DMC (visual), as shown in Table 3. On Atari100k, our method achieves performance competitive with the state-of-the-art self-predictive method MR.Q. While DreamerV3 outperforms NASDAQ on Atari100k, it employs a substantially larger model with approximately 200M parameters, which is $40\times$ larger than ours.

Table 2: Aggregate results, mean, median, and interquartile mean (IQM), on the two low-dimensional benchmarks, Gym and DMC (proprioceptive). The scores on Gym are normalized by the scores of a *deep* variant of TD3 before aggregation (see Appendix B.1). **Bold numbers** indicate the best performance. Full results are provided in Appendices C.1 and C.2.

Methods	**Gym**			**DMC (proprioceptive)**		
	Mean	Median	IQM	Mean	Median	IQM
NASDAQ with SARON (ours)	**1.79**	1.26	**1.50**	**805**	**927**	**898**
OFENET+TD3	1.48	1.24	1.44	567	696	600
TD7	1.16	**1.30**	1.14	660	810	726
MR.Q	1.50	1.12	1.36	791	918	886
TD-MPC2	0.36	0.20	0.20	783	896	868
DreamerV3	0.66	0.62	0.68	530	700	577

Computational efficiency comparison Figure 3 compares the training wall-time of NASDAQ (with SARON) with that of two general-purpose RL algorithms (MR.Q [17] and DreamerV3 [6]) and two model-free baselines (TD3 [4] and DrQ-v2 [22]). We scale down DreamerV3 to make its model size comparable to those of the other methods. All methods are implemented in PyTorch [38] and run on a single RTX 4090 GPU. In low-dimensional settings, NASDAQ with SARON has

Table 3: Aggregated results on two high-dimensional benchmarks. The results on Atari100k are normalized by human scores before aggregation (see Appendix B.1). **Bold numbers** indicate the best performance. The full results are provided in Appendices C.3 and C.4.

Methods	DMC (visual)			Atari100k		
	Mean	Median	IQM	Mean	Median	IQM
NASDAQ (ours)	**606**	**836**	**701**	0.93	0.34	0.42
MR.Q	598	809	686	0.91	0.40	0.41
DreamerV3	463	493	452	**1.25**	**0.49**	**0.54**
DrQ-v2	510	626	545	-	-	-
TD-MPC2	492	572	501	-	-	-
SPR	-	-	-	0.65	0.42	0.43
Rainbow (enhanced)	-	-	-	0.54	0.29	0.35

training wall-time closest to that of model-free baselines, indicating it incurs the least computational overhead among data-efficient methods. This efficiency stems from the fact that NASDAQ introduces only minor modifications to standard model-free RL. Notably, NASDAQ even trains faster than the model-free baseline in visual settings. The reason behind this interesting result is that NASDAQ allocates a large proportion of parameters to the observation encoder (17%), whose computations are reused across learning objectives. In contrast, the observation encoder in DrQ-v2 accounts for only 0.7% of the total parameters.

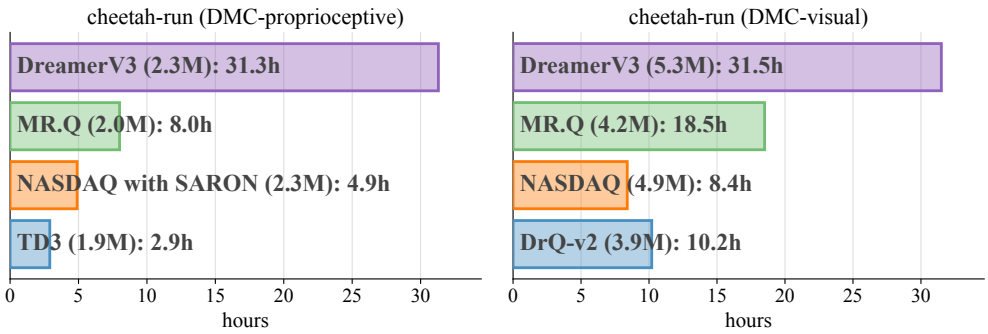


Figure 3: Training wall-time (in hours) comparisons on the *cheetah-run* task in low- and high-dimensional settings. The model size (in millions) of each method is shown in parentheses.

6.3 On the effectiveness of observation normalization in low-dimensional settings

As shown in Table 4, SARON significantly improves OFENet+TD3. It balances auxiliary losses across dimensions and addresses the identified bottleneck. The auxiliary-loss distribution of OFENet+TD3 with SARON is provided in Figure 11 of Appendix C.7. Considering both settings with and without auxiliary tasks, SARON outperforms simple ON on 9 out of 12 reported metrics, underperforms on 2, and ties on 1. This demonstrates that it is important to consider the distributional shift and perturbation on observations when normalizing online observation streams. We further examine the contributions of individual SARON components through an ablation study in Appendix D.2.

Notably, observation normalization also benefits model-free RL even without any auxiliary tasks. NASDAQ with SARON (w/o auxiliary tasks) remains competitive with MR.Q on Gym, while its performance on DMC (proprioceptive) lies between that of TD7 and TD-MPC2. Unlike MR.Q and TD7, which employ additional representation-learning objectives, and TD-MPC2, which relies on model-based planning, this variant retains a comparatively simple model-free design. We thus recommend this variant, which we call TDx, as a strong model-free RL baseline for low-dimensional settings.

Table 4: Aggregated results on two low-dimensional benchmarks. We use ↑ and ↓ to denote the performance changes of SARON relative to the simple ON baseline. **Bold numbers** indicate the best performance. Full results are provided in Appendix C.5.

Methods	Gym			DMC (proprioceptive)		
	Mean	Median	IQM	Mean	Median	IQM
OFENET+TD3	1.48	1.24	1.44	567	696	600
OFENET+TD3 with SARON	1.64	**1.29**	**1.54**	587	738	656
NASDAQ w/o ON (w/o auxiliary tasks)	1.29	1.20	1.35	593	762	645
NASDAQ w/o ON	0.99	1.00	0.98	627	870	745
NASDAQ with simple ON (w/o auxiliary tasks)	1.63	1.21	1.40	713	808	804
NASDAQ with SARON (w/o auxiliary tasks)	1.55 ↓	1.22 ↑	1.30 ↓	744 ↑	839 ↑	834 ↑
NASDAQ with simple ON	1.75	1.25	1.50	773	906	886
NASDAQ with SARON (complete)	**1.79** ↑	1.26 ↑	1.50	**805** ↑	**927** ↑	**898** ↑

6.4 Ablation study of auxiliary tasks

Table 5 summarizes the ablation results on the four benchmarks, where $\lambda > 0$ and $\lambda = 0$ denote that the auxiliary task is enabled and disabled, respectively. Details on how λ is selected when $\lambda > 0$ are provided in Appendix B.5. Across all benchmarks, normalized observation prediction brings significant improvements over the pure model-free variant ($\lambda_{\text{Rec}} = 0$, $\lambda_{\text{n-step}} = 0$). While further introducing short-term value prediction ($\lambda_{\text{Rec}} > 0$, $\lambda_{\text{n-step}} > 0$) consistently improves performance on visual RL benchmarks, it leads to a slightly negative effect in non-visual settings. We hypothesize that this reflects a bias–variance trade-off. On the one hand, the auxiliary task provides lower-variance learning signals than those from value estimation, thereby stabilizing learning. This advantage is particularly pronounced in visual RL tasks, where bootstrapped targets tend to have high variance due to high-dimensional inputs. On the other hand, it may bias the agent toward short-term value prediction and interfere with long-term value estimation. This negative effect may exist in both visual and non-visual settings, but it becomes more evident in the latter, where bootstrapped targets have relatively low variance and the benefit of variance reduction is thus diminished.

Table 5: Aggregate performance of NASDAQ (with SARON) variants across benchmarks. **Bold numbers** indicate the best performance. The full results are provided in Appendix C.6.

Versions	Gym			DMC (proprioceptive)			DMC (visual)			Atari100k		
	Mean	Median	IQM	Mean	Median	IQM	Mean	Median	IQM	Mean	Median	IQM
$\lambda_{\text{Rec}} = 0$, $\lambda_{\text{n-step}} = 0$	1.55	1.22	1.30	744	839	834	422	372	375	0.38	0.16	0.23
$\lambda_{\text{Rec}} = 0$, $\lambda_{\text{n-step}} > 0$	1.64	1.19	1.31	726	794	816	516	635	541	0.71	0.32	0.38
$\lambda_{\text{Rec}} > 0$, $\lambda_{\text{n-step}} = 0$	**1.79**	**1.26**	**1.50**	**805**	**927**	**898**	589	792	668	0.86	0.16	0.36
$\lambda_{\text{Rec}} > 0$, $\lambda_{\text{n-step}} > 0$	1.78	1.21	1.47	762	899	874	**606**	**836**	**701**	**0.93**	**0.34**	**0.42**

7 Conclusion and discussion

This paper identifies a key bottleneck in observation-predictive RL, i.e., unbalanced observation reconstruction losses across dimensions. To address this bottleneck, we propose SARON, a novel observation normalization method for low-dimensional settings. We further introduce NASDAQ, a unified framework for observation-predictive RL applicable across diverse domains. Extensive experiments demonstrate that NASDAQ (with SARON) achieves competitive or superior performance compared with state-of-the-art self-predictive and model-based approaches, while incurring significantly less computational overhead. We discuss broader impacts in Appendix F.

Generality of NASDAQ While many RL approaches showcase their generality by employing a unified network architecture with a single set of hyperparameters for tasks across domains [6, 8, 17], NASDAQ characterizes generality from an alternative perspective. Its generality lies in its principled compatibility with a wide range of existing value-based RL methods, requiring only modest modifications (e.g., value targets). We believe that this form of generality, centered on extensibility, has the potential to advance research across diverse RL domains.

Limitations and future directions Although we provide a simple and efficient method for selecting benchmark-level auxiliary loss weights in Appendix B.5, NASDAQ falls short of achieving a universal configuration of these weights, mainly due to the heterogeneity of observation spaces across benchmarks. Besides, the two auxiliary loss weights are fixed during training. An important direction for future research is to design an approach that can adaptively adjust the weights according to observation spaces, training metrics, and possibly other indicators. Another limitation is that SARON only applies to low-dimensional settings. Given its promising performance, future work could focus on designing normalization methods for high-dimensional observations (e.g., images).

Acknowledgments and Disclosure of Funding

This work was supported in part by the Major Key Project of PCL under Grant PCL2025A03, the National Natural Science Foundation of China (62472459), Guangdong S&T Program (2025B0101080001), Shandong Provincial Key Research and Development Program (No. 2022CXGC020106).

References

[1] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning - an introduction, 2nd Edition*. MIT Press, 2018. URL http://www.incompleteideas.net/book/the-book-2nd.html.

[2] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nat.*, 518(7540):529–533, 2015. doi: 10.1038/NATURE14236. URL https://doi.org/10.1038/nature14236.

[3] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

[4] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1582–1591. PMLR, 2018. URL http://proceedings.mlr.press/v80/fujimoto18a.html.

[5] Danijar Hafner, Timothy P. Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=S1lOTC4tDS.

[6] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy P. Lillicrap. Mastering diverse domains through world models. *CoRR*, abs/2301.04104, 2023. doi: 10.48550/ARXIV.2301.04104. URL https://doi.org/10.48550/arXiv.2301.04104.

[7] Nicklas Hansen, Hao Su, and Xiaolong Wang. Temporal difference learning for model predictive control. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 8387–8406. PMLR, 2022. URL https://proceedings.mlr.press/v162/hansen22a.html.

[8] Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: scalable, robust world models for continuous control. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=Oxh5CstDJU.

[9] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy P. Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nat.*, 588(7839):604–609, 2020. doi: 10.1038/S41586-020-03051-4. URL https://doi.org/10.1038/s41586-020-03051-4.

[10] Herke van Hoof, Nutan Chen, Maximilian Karl, Patrick van der Smagt, and Jan Peters. Stable reinforcement learning with autoencoders for tactile and visual data. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2016, Daejeon, South Korea, October 9-14, 2016*, pages 3928–3934. IEEE, 2016. doi: 10.1109/IROS.2016.7759578. URL https://doi.org/10.1109/IROS.2016.7759578.

[11] Amy Zhang, Harsh Satija, and Joelle Pineau. Decoupling dynamics and reward for transfer learning. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Workshop Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=H1aoddyvM.

[12] Kei Ota, Tomoaki Oiki, Devesh K. Jha, Toshisada Mariyama, and Daniel Nikovski. Can increasing input dimensionality improve deep reinforcement learning? In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 7424–7433. PMLR, 2020. URL http://proceedings.mlr.press/v119/ota20a.html.

[13] Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G. Bellemare. Deepmdp: Learning continuous latent space models for representation learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2170–2179. PMLR, 2019. URL http://proceedings.mlr.press/v97/gelada19a.html.

[14] Zhaohan Daniel Guo, Bernardo Ávila Pires, Bilal Piot, Jean-Bastien Grill, Florent Altché, Rémi Munos, and Mohammad Gheshlaghi Azar. Bootstrap latent-predictive representations for multitask reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3875–3886. PMLR, 2020. URL http://proceedings.mlr.press/v119/guo20g.html.

[15] Max Schwarzer, Ankesh Anand, Rishab Goel, R. Devon Hjelm, Aaron C. Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=uCQfPZwRaUu.

[16] Scott Fujimoto, Wei-Di Chang, Edward J. Smith, Shixiang Gu, Doina Precup, and David Meger. For SALE: state-action representation learning for deep reinforcement learning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/c20ac0df6c213db6d3a930fe9c7296c8-Abstract-Conference.html.

[17] Scott Fujimoto, Pierluca D'Oro, Amy Zhang, Yuandong Tian, and Michael Rabbat. Towards general-purpose model-free reinforcement learning. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=R1hIXdST22.

[18] Johan Bjorck, Carla P. Gomes, and Kilian Q. Weinberger. Towards deeper deep reinforcement learning with spectral normalization. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 8242–8255, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/4588e674d3f0faf985047d4c3f13ed0d-Abstract.html.

[19] Florin Gogianu, Tudor Berariu, Mihaela Rosca, Claudia Clopath, Lucian Busoniu, and Razvan Pascanu. Spectral normalisation for deep reinforcement learning: An optimisation perspective. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 3734–3744. PMLR, 2021. URL http://proceedings.mlr.press/v139/gogianu21a.html.

[20] Lu Li, Jiafei Lyu, Guozheng Ma, Zilin Wang, Zhenjie Yang, Xiu Li, and Zhiheng Li. Normalization enhances generalization in visual reinforcement learning. In Mehdi Dastani, Jaime Simão Sichman, Natasha Alechina, and Virginia Dignum, editors, *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2024, Auckland, New Zealand, May 6-10, 2024*, pages 1137–1146. International Foundation for Autonomous Agents and Multiagent Systems / ACM, 2024. doi: 10.5555/3635637.3662970. URL https://dl.acm.org/doi/10.5555/3635637.3662970.

[21] Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=GY6-6sTvGaf.

[22] Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=_SJ-_yyes8.

[23] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *J. Mach. Learn. Res.*, 22:268:1–268:8, 2021. URL https://jmlr.org/papers/v22/20-1364.html.

[24] Tianwei Ni, Benjamin Eysenbach, Erfan Seyedsalehi, Michel Ma, Clement Gehring, Aditya Mahajan, and Pierre-Luc Bacon. Bridging state and history representations: Understanding self-predictive RL. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=ms0VgzSGF2.

[25] Claas Voelcker, Tyler Kastner, Igor Gilitschenski, and Amir-massoud Farahmand. When does self-prediction help? understanding auxiliary tasks in reinforcement learning. *RLJ*, 4: 1567–1597, 2024. URL https://rlj.cs.umass.edu/2024/papers/Paper197.html.

[26] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1856–1865. PMLR, 2018. URL http://proceedings.mlr.press/v80/haarnoja18b.html.

[27] Mark Towers, Ariel Kwiatkowski, John U. Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, Rodrigo Perez-Vicente, J. K. Terry, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Hannah Tan, and Omar G. Younis. Gymnasium: A standard interface for reinforcement learning environments. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diago, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/d7ff1795e8527f6443371c3933bdb52b-Abstract-Datasets_and_Benchmarks_Track.html.

[28] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy P. Lillicrap, and Martin A. Riedmiller. Deepmind control suite. *CoRR*, abs/1801.00690, 2018. URL http://arxiv.org/abs/1801.00690.

[29] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

[30] Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Gheshlaghi Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 3215–3222. AAAI Press, 2018. doi: 10.1609/AAAI.V32I1.11796. URL https://doi.org/10.1609/aaai.v32i1.11796.

[31] Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics: Methodology and distribution*, pages 492–518. Springer, 1992.

[32] Scott Fujimoto, David Meger, and Doina Precup. An equivalence between loss functions and non-uniform sampling in experience replay. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/a3bf6e4db673b6449c2f7d13ee6ec9c0-Abstract.html.

[33] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin A. Riedmiller. Deterministic policy gradient algorithms. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 387–395. JMLR.org, 2014. URL http://proceedings.mlr.press/v32/silver14.html.

[34] Johan Bjorck, Carla P. Gomes, and Kilian Q. Weinberger. Is high variance unavoidable in rl? A case study in continuous control. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=9xhgmsNVHu.

[35] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6379–6390, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/68a9750337a418a86fe06c1991a1d64c-Abstract.html.

[36] Andre Cianflone, Zafarali Ahmed, Riashat Islam, Avishek Joey Bose, and William L Hamilton. Discrete off-policy policy gradient using continuous relaxations. *Unpublished. https://joeybose. github. io/assets/Gradient_estimator. pdf*, 2019.

[37] Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents (extended abstract). In Qiang Yang and Michael J. Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 4148–4152. AAAI Press, 2015. URL http://ijcai.org/Abstract/15/585.

[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

[39] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012*, pages 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109. URL https://doi.org/10.1109/IROS.2012.6386109.

[40] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=B1QRgziT-.

[41] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas. Dueling network architectures for deep reinforcement learning. In Maria-Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1995–2003. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/wangf16.html.

[42] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Yash Katariya, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL https://github.com/jax-ml/jax.

[43] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1511.07289.

[44] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016. URL http://arxiv.org/abs/1607.06450.

[45] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 807–814. Omnipress, 2010. URL https://icml.cc/Conferences/2010/papers/432.pdf.

[46] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In Geoffrey J. Gordon, David B. Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, volume 15 of *JMLR Proceedings*, pages 315–323. JMLR.org, 2011. URL http://proceedings.mlr.press/v15/glorot11a/glorot11a.pdf.

[47] Lu Lu, Yeonjong Shin, Yanhui Su, and George E. Karniadakis. Dying relu and initialization: Theory and numerical examples. *CoRR*, abs/1903.06733, 2019. URL http://arxiv.org/abs/1903.06733.

[48] Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, Richard E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, 1(4):541–551, 1989. doi: 10.1162/NECO.1989.1.4.541. URL https://doi.org/10.1162/neco.1989.1.4.541.

[49] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791. URL https://doi.org/10.1109/5.726791.

[50] Matthew D. Zeiler, Dilip Krishnan, Graham W. Taylor, and Robert Fergus. Deconvolutional networks. In *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, pages 2528–2535. IEEE Computer Society, 2010. doi: 10.1109/CVPR.2010.5539957. URL https://doi.org/10.1109/CVPR.2010.5539957.

[51] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL `https://openreview.net/forum?id=Bkg6RiCqY7`.

A Pseudocode

Algorithm 1 RL with SARON

Input: environment env
Initialize policy π and value function V
Initialize running means and second moments (m, s)
Initialize replay buffer $\mathcal{D}$, observation buffer $\mathcal{O}$, and return queue $\mathcal{Q}$ with capacity Q
$o \leftarrow env.\text{reset}()$, $R \leftarrow 0$
for $t = 1$ to T **do**
    if t > warmup steps **then**
        Obtain normalized observation $\tilde{o}$ based on o and (m, s) using Equation 2 and 3
        Select action $a \sim \pi(\cdot \mid \tilde{o})$ ▷ Inference with the normalized observation
    else
        Sample random action a
    end if
    Add raw observation o to $\mathcal{O}$
    $o^{\text{next}}, r, d \leftarrow env.\text{step}(a)$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(o, a, r, o^{\text{next}}, d)\}$
    $R \leftarrow R + r$ ▷ Record episodic returns
    $o \leftarrow o^{\text{next}}$
    if episode ends **then**
        if $|\mathcal{Q}| = Q$ and R is a low-return outlier identified by the IQR method **then**
            Skip EMA updates on (m, s)
        else
            Compute current episode statistics (μ, ν) from $\mathcal{O}$
            Update (m, s) by (μ, ν) using Equation 1
        end if
        Append R to $\mathcal{Q}$ ▷ Maintain recent returns for IQR-based outlier detection
        Clear $\mathcal{O}$ ▷ Clear observation buffer for new episode
        $o \leftarrow env.\text{reset}()$, $R \leftarrow 0$
    end if
    if t > warmup steps **then**
        Sample mini-batch B from $\mathcal{D}$
        Normalize the observations in B based on (m, s) using Equation 2 and 3
        Train π and V with the processed B ▷ Training with normalized observations
    end if
end for

Algorithm 2 Updates of NASDAQ

Input: online networks V_1, V_2, π
Input: target networks V_1', V_2', π'
Input: replay buffer $\mathcal{D}$, current training step t, multi-step n for *n*-step returns
Sample a minibatch of $(o_t, a_t, r_{t:t+n-1}, o_{t+1}, o_{t+n}) \sim \mathcal{D}$
Obtain $(\tilde{o}_t, a_t, r_{t:t+n-1}, \tilde{o}_{t+1}, \tilde{o}_{t+n})$ by observation normalization
Optimize V_1 and V_2 against the combined loss $\mathcal{L}$ computed via Equation 11
if t % policy update frequency $= 0$ **then**
    Optimize π against the loss $\mathcal{L}_{\text{Policy}}$ computed via Equation 8
end if
$t \leftarrow t + 1$
if t % target update frequency $= 0$ **then**
    Synchronize target networks with online networks
end if

B Experimental details

B.1 Benchmarks

Evaluation protocol All experiments are run for 5 seeds. For each seed, the evaluation score is computed as the mean over 10 episodes, measured every 5k time steps. For Atari100k, however, the final evaluation (at the end of training) follows the protocol of SPR [15], where the score is averaged over 100 episodes.

Gym This benchmark consists of five common locomotion tasks defined by OpenAI Gym [27] in the MuJoCo simulator [39], with continuous actions and low-dimensional observations. We use the `-v5` version, which is quite different from the previous versions. See the version history at `https://gymnasium.farama.org/environments/mujoco/`. Agents are trained for 1M time steps. Similar to MR.Q [17], we use the scores of a *deep* variant of TD3 [4] to normalize the scores of other methods when aggregating results:

$$\text{Deep-TD3-Normalized}(x) = \frac{x - \text{random score}}{\text{Deep-TD3 score} - \text{random score}}. \tag{12}$$

Deep-TD3 is constructed by increasing the network size of the original TD3 to match the model capacity of other TD3-based methods on this benchmark. Specifically, we increase the depth of the value networks from three to five layers, with spectral normalization [18, 19, 40] applied to the 2nd through 4th layers to stabilize training. The number of hidden units in each layer of both the value and policy networks is increased from the default setting to 450.

Table 6: The scores of Deep-TD3 and a random policy on the Gym benchmark.

Environments	Random	Deep-TD3
Ant-v5	-0.6	3908
Humanoid-v5	92.2	2897
HalfCheetah-v5	-265.0	13538
Hopper-v5	25.1	3009
Walker2d-v5	5.1	4831

DMC (proprioceptive) The DeepMind Control suite (DMC) [28] is a collection of continuous control tasks built on the MuJoCo simulator [39]. These tasks use low-dimensional proprioceptive data as observations. The maximum total reward for each episode is 1000, making it easy to aggregate results. We report results on 28 tasks evaluated in MR.Q [17] and DreamerV3 [6]. Agents are trained for 500k time steps, equivalent to 1M frames in the original environment due to an action repeat of 2.

DMC (visual) Visual DMC includes the same 28 tasks as the proprioceptive benchmark but uses image-based observations instead. Consistent with the proprioceptive setting, agents are trained for 500k time steps. The input observation is composed of the previous 3 frames, which are resized to 84×84 pixels in RGB format [17].

Atari100k This data-efficiency benchmark contains 26 Atari games [37] for discrete control. Agents are allowed only 100k steps of environment interaction with an action repeat of 4 (producing 400k frames), amounting to 2 hours of game time. We use the no sticky action setting, following the default configuration used by DreamerV3 in this benchmark (`https://github.com/danijar/dreamerv3/blob/main/dreamerv3/configs.yaml`). The observation is composed of the previous 4 frames, which are gray-scaled, resized to 84×84 pixels and set to the max between the 3rd and 4th frame [17]. When aggregating scores, we normalize with human scores reported by [41] (see Table 7):

$$\text{Human-Normalized}(x) = \frac{x - \text{random score}}{\text{Human score - random score}}. \tag{13}$$

Table 7: Human and random policy scores on Atari100k games.

Games	Random	Human
Alien	227.8	7127.7
Amidar	5.8	1719.5
Assault	222.4	742
Asterix	210	8503.3
BankHeist	14.2	753.1
BattleZone	2360.0	37187.5
Boxing	0.1	12.1
Breakout	1.7	30.5
ChooperCommand	811.0	7387.8
CrazyClimber	10780.5	35829.4
DemonAttack	152.1	1971.0
Freeway	0.0	29.6
Frostbite	65.2	4334.7
Gopher	257.6	2412.5
Hero	1027.0	30826.4
Jamesbond	29.0	302.8
Kangaroo	52.0	3035.0
Krull	1598.0	2665.5
KungFuMaster	258.5	22736.3
MsPacman	307.3	6951.6
Pong	-20.7	14.6
PrivateEye	24.9	69571.3
Qbert	163.9	13455.0
RoadRunner	11.5	7845.0
Seaquest	68.4	42054.7
UpNDown	533.4	11693.2

B.2 Baselines

Unless otherwise specified, baseline results are obtained by running the corresponding algorithms.

OFENet+TD3 The Online Feature Extractor Network (OFENet) [12] was proposed to learn state-action representations through observation dynamics prediction for decoupled model-free RL algorithms. It is an observation-predictive method for continuous control tasks with low-dimensional observations. We adopt the OFENet+TD3 variant in our experiments.

TD7 TD7 [16] is a self-predictive method built on OFENet and TD3 that achieves state-of-the-art results on the Gym benchmark.

SPR Self-Predictive Representation (SPR) [15] is a self-predictive method for visual discrete control tasks. It builds upon Rainbow and demonstrates high data efficiency in Atari100k.

TD-MPC2 TD-MPC2 [8] is a model-based method for continuous-control tasks that performs local trajectory optimization (planning) in the latent space of a learned world model. While the results for DMC (proprioceptive) were obtained from MR.Q, we re-run the authors' code to gather the results for Gym because we use the `-v5` version, which is quite different from the previous versions.

DreamerV3 DreamerV3 [6] is a strong model-based RL algorithm that masters diverse domains, where a model-free policy is optimized through the rollouts from a learned world model. The results for Gym were obtained by re-running the algorithm. The results for DMC and Atari100k were obtained from MR.Q and the original paper, respectively. Note that while DreamerV3 was originally implemented by JAX [42], we use a PyTorch [38] implementation (`https://github.com/NM512/dreamerv3-torch/`) in the experiments of computational efficiency, for a fair comparison with other methods implemented by PyTorch.

MR.Q Model-based Representations for Q-learning (MR.Q) [17] is a state-of-the-art self-predictive method that achieves performance competitive with model-based RL approaches, DreamerV3 and TD-MPC2, on diverse benchmarks with much less algorithmic complexity and computational overhead. Similar to OFENet+TD3 and TD7, MR.Q builds on TD3 and learns state-action representations for decoupled model-free RL algorithms. In addition to latent dynamics prediction, MR.Q introduces two extra auxiliary tasks of reward and terminal signal prediction.

DrQ-v2 DrQ-v2 [22] is a simple yet strong model-free RL baseline based on data augmentation for visual continuous control tasks. Results were obtained from MR.Q.

Rainbow (enhanced) SPR introduces a series of improvements to a model-free algorithm Rainbow [30], tailored to the characteristics of the Atari100k benchmark. We keep these design choices, except the core latent dynamics learning process, to construct an enhanced variant of Rainbow for the benchmark.

B.3 Network architecture of NASDAQ

Low-dimensional settings The observation encoder is simply an identity function, whereas the state-action encoder is a three-layer multilayer perceptron (MLP) with an Exponential Linear Unit (ELU) activation [43] applied after each layer. Spectral normalization (SN) [18, 19, 40] is applied to the last two layers. The short-term value predictor is a two-layer MLP, where the first layer is followed by layer normalization (LayerNorm) [44] and an ELU activation. We use a non-linear decoder instead of the linear one adopted by OFENet [12] to avoid overly strong regularization on state-action representations. The decoder is a two-layer MLP with ELU activation applied after the first layer. Following TD3 [4], the policy network is a three-layer MLP where the first two layers are followed by ReLU activations. The long-term value predictor comprises three linear layers, with SN applied to the first two layers. By default, we employ ReLU activations [45, 46] in the first two layers of the long-term value predictor, following TD3 [4]. However, we find that using ELU, as adopted in MR.Q [17], yields better performance on the *acrobot-swingup* and *pendulum-swingup* tasks from the DMC (proprioceptive) benchmark. We thus replace ReLU with ELU only for these two tasks. We conjecture that ELU performs better in these tasks due to their task-specific properties. *Pendulum-swingup* is a sparse-reward task, where ReLU may suffer from dead neurons [45, 47], while *acrobot-swingup* is a challenging task (as noted in the DeepMind Control Suite [28]), potentially requiring a more expressive activation function to model its value function. ELU may alleviate both issues. A more thorough investigation of activation function choices is left for future work. We also include a PyTorch-style [38] code snippet to illustrate the architecture, omitting non-critical implementation details for clarity.

Network Architecture in low-dimensional settings

```
import torch
import torch.nn as nn
import torch.nn.functional as F
from torch.nn.utils import spectral_norm as SN
from functools import partial


# Value network with auxiliary prediction heads
class ValueNetwork(nn.Module):
    def __init__(self, obs_dim, action_dim, hidden_dim, zsa_dim,
        rew_bins):
        self.obs_enc = nn.Identity()
        self.state_action_enc = nn.Sequential(
            nn.Linear(int(obs_dim + action_dim), hidden_dim),
            nn.ELU(),
            SN(nn.Linear(hidden_dim, hidden_dim)),
            nn.ELU(),
            SN(nn.Linear(hidden_dim, zsa_dim)),
            nn.ELU(),
        )

        self.long_term_val_predictor = nn.Sequential(
```

```
            SN(nn.Linear(zsa_dim, hidden_dim)),
            nn.ReLU(),
            nn.Linear(hidden_dim, hidden_dim),
            nn.ReLU(),
            nn.Linear(hidden_dim, 1)
        )

        self.short_term_val_predictor = nn.Sequential(
            nn.Linear(zsa_dim, hidden_dim),
            nn.LayerNorm(hidden_dim, elementwise_affine=False),
            nn.ELU(),
            nn.Linear(hidden_dim, rew_bins)
        )

        self.decoder = nn.Sequential(
            nn.Linear(zsa_dim, hidden_dim),
            nn.ELU(),
            nn.Linear(hidden_dim, obs_dim)
        )

    def forward(self, obs, action):
        s = self.obs_enc(obs)
        zsa = torch.cat([s, action], -1)
        zsa = self.state_action_enc(zsa)

        q_val = self.long_term_val_predictor(zsa)
        short_term_val = self.short_term_val_predictor(zsa)
        next_obs = self.decoder(zsa)
        return q_val, short_term_val, next_obs


class PolicyNetork(nn.Module):
    def __init__(self, obs_dim, action_dim, hidden_dim):
        self.policy = nn.Sequential(
            nn.Linear(obs_dim, hidden_dim),
            nn.ReLU(),
            nn.Linear(hidden_dim, hidden_dim),
            nn.ReLU(),
            nn.Linear(hidden_dim, action_dim)
        )

        if discrete_action_space:
            self.final_activ = partial(F.gumbel_softmax, tau=10)
        else:
            self.final_activ = torch.tanh

    def forward(self, s):
        pre_activ = self.policy(s)
        action = self.final_activ(pre_activ)
        return action
```

High-dimensional settings We adopt the same architectural design as MR.Q [17] for the observation encoder, state-action encoder, long-term value predictor, and policy network. Specifically, for the observation encoder, four convolutional (Conv) layers [48, 49] are used, each with 32 output channels, a kernel size of 3, strides of (2, 2, 2, 1), and ELU activations. The convolutional layers are followed by a linear layer taking in the flattened output, followed by LayerNorm and a final ELU activation. The state-action encoder is a three-layer MLP with LayerNorm followed by ELU after the first two layers. The long-term value predictor is a four-layer MLP with LayerNorm followed by ELU after the first three layers. We adopt a decoder similar to that used in Dreamer [5], consisting of a linear layer that projects the input vector to a feature map of size (*latent_channels*, 1, 1), followed by four transposed convolutional (TransConv) [50] layers with strides 2 and kernel sizes of (7, 6, 6, 6). The channel dimension decreases from *latent_channels* to 64, 64, 32, and finally *out_channels*. We additionally apply spectral normalization [18, 19, 40] to the first two layers of the decoder to

restrict its capacity, encouraging upstream representations to carry more predictive information. The policy network is a three-layer MLP with LayerNorm followed by ReLU activations after the first two layers. Below is the corresponding PyTorch-style code snippet, including only the components that differ from the low-dimensional setting.

Network Architecture in high-dimensional settings

```
# Value network with auxiliary prediction heads
class ValueNetwork(nn.Module):
    def __init__(self, in_channels, state_dim, action_dim, hidden_dim,
         zsa_dim, latent_channels, out_channels):
        self.obs_enc = nn.Sequential(
            nn.Conv2d(in_channels, 32, kernel_size=3, stride=2),
            nn.ELU(),
            nn.Conv2d(32, 32, kernel_size=3, stride=2),
            nn.ELU(),
            nn.Conv2d(32, 32, kernel_size=3, stride=2),
            nn.ELU(),
            nn.Conv2d(32, 32, kernel_size=3, stride=1),
            nn.ELU(), nn.Flatten(),
            nn.Linear(flatten_dim, state_dim),
            nn.LayerNorm(state_dim, elementwise_affine=False),
            nn.ELU()
        )

        self.state_action_enc = nn.Sequential(
            nn.Linear(int(state_dim + action_dim), hidden_dim),
            nn.LayerNorm(state_dim, elementwise_affine=False),
            nn.ELU(),
            nn.Linear(hidden_dim, hidden_dim),
            nn.LayerNorm(state_dim, elementwise_affine=False),
            nn.ELU(),
            nn.Linear(hidden_dim, zsa_dim),
        )

        self.long_term_val_predictor = nn.Sequential(
            nn.Linear(zsa_dim, hidden_dim),
            nn.LayerNorm(hidden_dim, elementwise_affine=False),
            nn.ELU(inplace=True),
            nn.Linear(hidden_dim, hidden_dim),
            nn.LayerNorm(hidden_dim, elementwise_affine=False),
            nn.ELU(inplace=True),
            nn.Linear(hidden_dim, hidden_dim),
            nn.LayerNorm(hidden_dim, elementwise_affine=False),
            nn.ELU(inplace=True),
            nn.Linear(hidden_dim, 1)
        )

        self.decoder = nn.Sequential(
            SN(nn.Linear(zsa_dim, latent_channels)),
            nn.Unflatten(1, (latent_channels, 1, 1)),
            SN(nn.ConvTranspose2d(latent_channels, 64, kernel_size=7,
                stride=2)),
            nn.ReLU(),
            nn.ConvTranspose2d(64, 64, kernel_size=6, stride=2),
            nn.ReLU(),
            nn.ConvTranspose2d(64, 32, kernel_size=6, stride=2),
            nn.ReLU(),
            nn.ConvTranspose2d(32, out_channels, kernel_size=6, stride
                =2)
        )


class PolicyNetork(nn.Module):
    def __init__(self, obs_dim, action_dim, hidden_dim):
```

```
        self.policy = nn.Sequential(
            nn.Linear(obs_dim, hidden_dim),
            nn.LayerNorm(hidden_dim, elementwise_affine=False),
            nn.ReLU(),
            nn.Linear(hidden_dim, hidden_dim),
            nn.LayerNorm(hidden_dim, elementwise_affine=False),
            nn.ReLU(),
            nn.Linear(hidden_dim, action_dim)
        )
```

B.4 Implementation details

Design choices alignment To reduce potential confounding factors when comparing against MR.Q [17], we keep certain design choices of OFENet+TD3 and NASDAQ consistent with those of MR.Q for components not directly related to representation learning. Specifically, we use the LAP replay buffer [32] for prioritized sampling during training, and adopt the same target network update strategy, where all target parameters are periodically synchronized with their online counterparts. The value learning loss is computed using the Huber loss [31, 32]. All networks are trained with the AdamW [51] optimizer.

Parameter sharing Similar to prior work [6, 17, 21], we share the parameters of the observation and state-action encoders between the two value networks of NASDAQ, i.e., $f_1 = f_2$, $g_1 = g_2$, in high-dimensional settings to stabilize and accelerate learning. As a result of sharing these encoders, the inputs to the auxiliary prediction heads become identical across the two networks. We therefore merge the corresponding auxiliary heads ($q_1 = q_2$, $\mathrm{Dec}_1 = \mathrm{Dec}_2$).

Benchmark-specific modifications For the Atari100k benchmark, NASDAQ clips rewards to $\{-1, 0, +1\}$ based on their sign, which is a preprocessing method commonly used for Atari games [2, 15, 30]. Following SPR [15], both NASDAQ and MR.Q perform *two* updates of all networks per training step in this benchmark. Following DrQ-v2 [22] and MR.Q [17], we apply random-shift image augmentation to pixel observations on the DMC (visual) and Atari100k benchmarks.

Observation prediction targets in image-based tasks As described in Appendix B.1, the observation in pixel-based benchmarks is composed of several previous frames. This raises the question of how to construct learning targets for next observation prediction. A straightforward approach is to use the *complete* next observation as the target. However, this target contains redundant information from the current observation, which may reduce the efficiency of learning dynamics-aware representations. We thus remove the overlapping frames between the current and next observations when constructing targets. Note that since input observations undergo random-shift augmentation, we apply the same sampled spatial shift to each corresponding prediction target to preserve input-target alignment in spatial shift.

B.5 Hyper-parameters

Table 8 and Table 9 summarize the default hyperparameters of NASDAQ and SARON in low- and high-dimensional settings, respectively.

Hyperparameter selection for NASDAQ We describe how the auxiliary loss weights, λ_{Rec} and $\lambda_{\text{n-step}}$, are determined. The two weights correspond to the two auxiliary tasks of observation prediction and short-term value prediction, respectively, as introduced in Section 5.2. For λ_{Rec}, we determine its value based on a training metric defined as the ratio between the gradient norm of the auxiliary loss $\mathcal{L}_{\text{Rec}}$ and that of the long-term value learning objective $\mathcal{L}_{\text{Value}}$, computed with respect to the shared observation encoder f_{θ_s} and state-action encoder $g_{\theta_{sa}}$:

$$\rho = \frac{\|\nabla_\Theta \mathcal{L}_{\text{Rec}}\|}{\|\nabla_\Theta \mathcal{L}_{\text{Value}}\|}, \quad \Theta = \{\theta_s, \theta_{sa}\}. \tag{14}$$

This gradient norm ratio provides a measure of the relative contributions of the auxiliary and main objectives to the shared parameters. If ρ is too large, the auxiliary objective may dominate the

gradient updates and hinder long-term value learning, and vice versa. To ensure that both objectives effectively influence the shared parameters, we determine benchmark-specific λ_{Rec} by heuristically controlling $\rho \in [0.1, 2]$ on a small subset of tasks, as shown in Figure 4. Note that while not all tasks satisfy the constraint, this heuristic offers a simple and effective method to select a decent λ_{Rec} at the benchmark level, which serves as a starting point for further task-specific hyperparameter tuning. In our experiments, however, we use only the benchmark-level auxiliary loss weights across all tasks. The specific values of λ_{Rec} and $\lambda_{\text{n-step}}$ are provided in Table 8 and Table 9.

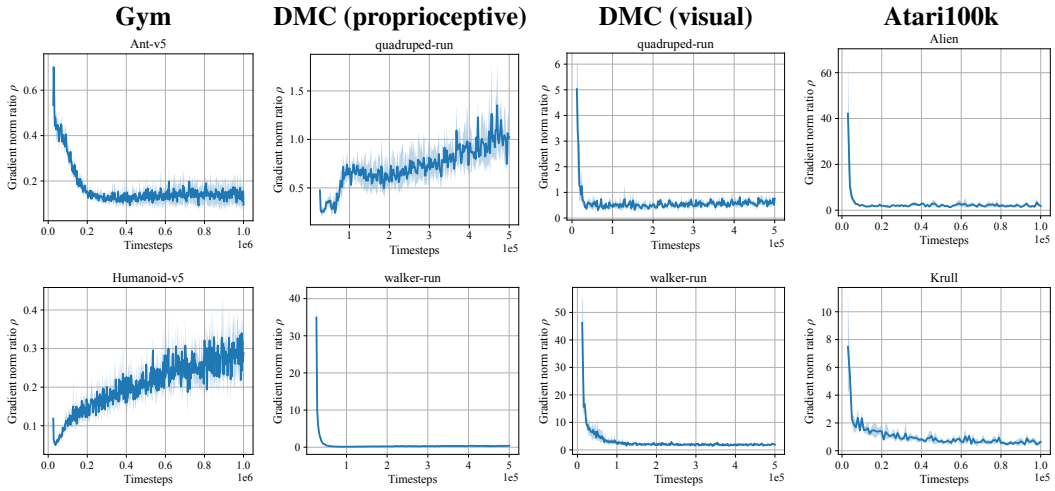


Figure 4: Gradient norm ratio ρ during training. Except for initial instability during early training, ρ predominantly lies within the interval $[0.1, 2]$.

For $\lambda_{\text{n-step}}$, we set its value to 1 for visual RL benchmarks without tuning. For benchmarks with low-dimensional observations, we simply set $\lambda_{\text{n-step}}$ to 0, indicating that short-term value prediction is not included, because we find that this auxiliary task, when $\lambda_{\text{n-step}} = 1$, leads to slight negative effects in the non-visual setting, as shown in Table 5. Although short-term value prediction benefits visual RL tasks across benchmarks, the situation is reversed in non-visual settings. We hypothesize that this phenomenon reflects a bias–variance trade-off. On the one hand, this auxiliary task provides lower-variance learning signals than those from long-term value prediction, which stabilizes the learning process. This advantage is particularly pronounced in visual RL tasks, where bootstrapped targets tend to be high-variance due to high-dimensional inputs. On the other hand, it may bias the agent toward short-term value prediction and interfere with long-term value estimation. While this negative effect may be present in both visual and non-visual settings, it becomes more evident in non-visual domains, where the variance of bootstrapped targets is relatively low, and the benefit of variance reduction is thus diminished.

Hyper-parameters selection for SARON An EMA with coefficient β can be viewed as having an effective window size of approximately $1/(1-\beta)$ [17]. For the two low-dimensional benchmarks, Gym and DMC (proprioceptive), we set the effective window size to 200k, meaning that the statistics are computed using approximately 200k historical observations. Since the maximum episode lengths of Gym and DMC are 1000 and 500 (due to an action repeat of 2), respectively, the queue sizes of historical episodes $|\mathcal{Q}|$ are thus 200 and 400, respectively. We further evaluate the sensitivity of this choice using effective window sizes ranging from 50k to 500k on four representative tasks, with results reported in Table 29 of Appendix D.3.

B.6 Parameter control protocol

We divide the parameters of the TD3-based methods (NASDAQ, MR.Q, OFENet+TD3, and TD7) into two categories: (1) RL components and (2) method-specific components. We define RL components as those involved in the RL training and inference processes, while method-specific components refer to the parameters used solely for representation learning. For a fair comparison, we keep the number of parameters in the RL components comparable across methods. Parameters used only for

Table 8: Default hyperparameters of NASDAQ and SARON in non-visual settings.

Components	**Hyperparameter**	**Value**
Value Learning	n-step returns	1
	Auxiliary loss weights $(\lambda_{\text{Rec}}, \lambda_{\text{n-step}})$	Gym: (10, 0) DMC: (2, 0)
	Discount factor γ	0.99
SARON	Queue size of historical episodes $\lvert\mathcal{Q}\rvert$	Gym: 200 DMC: 400
	EMA coefficient β	Gym: $1 - 1/200$ DMC: $1 - 1/400$
	Clip bounds $(-O, O)$	(-10, 10)
TD3	Target policy noise σ	$\mathcal{N}(0, 0.2^2)$
	Target policy noise clipping c	(-0.5, 0.5)
LAP	Probability smoothing α	0.4
	Minimum priority	1
Optimization	Optimizer	AdamW
	Learning rate	3e-4
	Weight decay	1e-4
	Mini-batch size	256
	Target update frequency	250
	Gradient updates per training step	value network: 1 policy network: 0.5
Exploration	Initial random exploration time steps	25k
	Exploration noise	$\mathcal{N}(0, 0.1^2)$
Observation Encoder	Structure	identity function
State-Action Encoder	Hidden dim	450
	z_{sa} dim	450
	Activation function	ELU
	Gradient clip norm	20
Long-term value predictor	Hidden dim	450
	Activation function	ReLU
	Gradient clip norm	20
Short-term value predictor	Hidden dim	450
	Activation function	ELU
	Reward bins	65
	Gradient clip norm	20
	Reward range	$[-10, 10]$ (effective: $[-22\text{k}, 22\text{k}]$)
Decoder	Structure	MLP
	Hidden dim	450
	Activation function	ELU
	Gradient clip norm	20
Policy Network	Hidden dim	450
	Activation function	ReLU

Table 9: Default hyperparameters of NASDAQ in visual settings. For brevity, we only list hyperparameters that differ from the non-visual settings. SARON is not used in these settings.

Components	**Hyperparameter**	**Value**
Value Learning	n-step returns	3
	Auxiliary loss weights $(\lambda_{\text{Rec}}, \lambda_{\text{n-step}})$	DMC: (0.1, 1) Atari100k: (1, 1)
TD3	Target policy noise clipping c	(-0.3, 0.3)
Optimization	Gradient updates per training step	value network: 1 (DMC), 2 (Atari100k) policy network: 1 (DMC), 2 (Atari100k)
Exploration	Initial random exploration time steps α	DMC: 10k Atari100k: 2k
	Exploration noise	DMC: $\mathcal{N}(0, 0.1^2)$ Atari100k: $\mathcal{N}(0, 0.2^2)$
Observation Encoder	Structure	Conv + MLP
	state dim	512
	Activation function	ELU
	Gradient clip norm	20
State-Action Encoder	Hidden dim	580
	z_{sa} dim	512
Long-term value predictor	Hidden dim	512
	Activation function	ELU
Short-term value predictor	Hidden dim	512
Decoder	Structure	MLP + TransConv
	Latent channels	128
	Output channels	DMC: 3 Atari100k: 1
	Activation function	ReLU
Policy Network	Hidden dim	512
	Gumbel-Softmax τ	10

representation learning are not restricted, as they are inherently determined by the characteristics of each method. Note that we exclude the target networks when counting the number of parameters, since they are maintained as historical backups of the online counterparts and do not contribute additional model capacity [4, 16, 17, 30]. Specifically, we control the number of parameters across these methods by adjusting their hidden sizes. Tables 10 and 11 summarize the numbers of parameters across TD3-based methods in low- and high-dimensional settings, respectively. Table 12 lists the specific hidden sizes for MR.Q, OFENet+TD3, and TD7. The hidden sizes of NASDAQ are provided in Tables 8 and 9.

Table 10: Parameter counts (in millions) for RL components and all components of TD3-based methods on low-dimensional benchmarks.

Domains	**RL Components**				**All the Components**			
	NASDAQ	MR.Q	OFENet+TD3	TD7	NASDAQ	MR.Q	OFENet+TD3	TD7
Gym								
Ant	1.88	1.92	1.96	1.90	2.31	2.08	1.98	1.90
Humanoid	2.32	2.41	2.16	2.35	3.04	2.60	2.50	2.35
HalfCheetah	1.86	1.91	1.93	1.89	2.28	2.08	1.94	1.89
Hopper	1.85	1.91	1.91	1.87	2.26	2.08	1.92	1.87
Walker2d	1.86	1.91	1.93	1.89	2.28	2.08	1.94	1.89
DMC (proprioceptive)								
acrobot	1.84	1.86	1.90	1.86	2.25	2.02	1.90	1.86
ball_in_cup	1.84	1.86	1.90	1.87	2.26	2.02	1.91	1.87
cartpole	1.84	1.86	1.89	1.86	2.25	2.02	1.90	1.86
cheetah	1.86	1.91	1.93	1.89	2.28	2.08	1.94	1.89
dog	2.18	2.28	2.19	2.21	2.79	2.47	2.41	2.21
finger	1.84	1.86	1.91	1.87	2.26	2.02	1.91	1.87
fish	1.87	1.92	1.94	1.89	2.30	2.08	1.96	1.89
hopper	1.85	1.91	1.92	1.88	2.27	2.08	1.93	1.88
humanoid	1.95	1.99	1.92	1.98	2.41	2.16	1.93	1.98
pendulum	1.83	1.92	1.89	1.86	2.24	2.16	1.89	1.86
quadruped	1.95	1.99	2.07	1.98	2.43	2.16	2.15	1.98
reacher	1.84	1.86	1.90	1.87	2.25	2.02	1.90	1.87
walker	1.87	1.92	1.95	1.90	2.30	2.08	1.97	1.90

Table 11: Parameter counts (in millions) for RL components and all components of TD3-based methods on high-dimensional benchmarks.

Domains	**RL Components**		**All the Components**	
	NASDAQ	MR.Q	NASDAQ	MR.Q
DMC (proprioceptive)				
acrobot	3.87	3.86	4.86	4.15
ball_in_cup	3.87	3.86	4.86	4.15
cartpole	3.87	3.86	4.86	4.15
cheetah	3.87	3.86	4.86	4.16
dog	3.91	3.88	4.90	4.18
finger	3.87	3.86	4.86	4.15
fish	3.87	3.86	4.86	4.16
hopper	3.87	3.86	4.86	4.15
humanoid	3.89	3.87	4.88	4.17
pendulum	3.87	3.86	4.86	4.15
quadruped	3.88	3.86	4.87	4.16
reacher	3.87	3.86	4.86	4.15
walker	3.87	3.86	4.86	4.16
Atari100k				
Alien	3.89	3.87	4.87	4.16
Amidar	3.88	3.86	4.86	4.16
Assault	3.87	3.86	4.86	4.16
Asterix	3.88	3.86	4.86	4.16
BankHeist	3.89	3.87	4.87	4.16
BattleZone	3.89	3.87	4.87	4.16
Boxing	3.89	3.87	4.87	4.16
Breakout	3.87	3.86	4.86	4.15
ChopperCommand	3.89	3.87	4.87	4.16
CrazyClimber	3.88	3.86	4.86	4.16
DemonAttack	3.87	3.86	4.86	4.15
Freeway	3.87	3.86	4.86	4.15
Frostbite	3.89	3.87	4.87	4.16
Gopher	3.88	3.86	4.86	4.16
Hero	3.89	3.87	4.87	4.16
Jamesbond	3.89	3.87	4.87	4.16
Kangaroo	3.89	3.87	4.87	4.16
Krull	3.89	3.87	4.87	4.16
KungFuMaster	3.88	3.86	4.87	4.16
MsPacman	3.88	3.86	4.86	4.16
Pong	3.87	3.86	4.86	4.15
PrivateEye	3.89	3.87	4.87	4.16
Qbert	3.87	3.86	4.86	4.15
RoadRunner	3.89	3.87	4.87	4.16
Seaquest	3.89	3.87	4.87	4.16
UpNDown	3.87	3.86	4.86	4.15

Table 12: Hidden size configurations of TD3-based methods across domains.

Domains	MR.Q	OFENet+TD3	TD7
Gym			
Ant	375	420	340
Humanoid	408	360	340
HalfCheetah	375	420	340
Hopper	375	420	340
Walker2d	375	420	340
DMC (proprioceptive)			
acrobot	370	360	340
ball_in_cup	370	360	340
cartpole	370	360	340
cheetah	375	360	340
dog	400	420	340
finger	370	360	340
fish	375	360	340
hopper	375	360	340
humanoid	380	360	340
pendulum	370	360	340
quadruped	380	360	340
reacher	370	360	340
walker	375	360	340
DMC (visual)			
	512	N/A	N/A
Atari100k			
	512	N/A	N/A

C Complete results

C.1 Gym

Table 13: Final performance on the **Gym** benchmark at 1M time steps, averaged over 5 seeds. The [bracketed values] represent a 95% bootstrap confidence interval. The aggregate mean, median, and interquartile mean (IQM) are computed using the Deep-TD3-normalized score (see Appendix B.1).

Tasks	OFENet+TD3	TD7	TD-MPC2	DreamerV3	MR.Q	NASDAQ with SARON
Ant-v5	8156 [8047, 8280]	7181 [6270, 8106]	1120 [555, 2117]	1457 [200, 2991]	7514 [6977, 7911]	7871 [7664, 8103]
HalfCheetah-v5	13548 [12867, 14309]	17723 [17446, 18019]	15127 [14518, 15619]	7107 [6576, 7872]	13823 [13412, 14251]	16742 [16476, 17031]
Hopper-v5	2853 [1837, 3637]	1612 [1272, 1920]	313 [222, 431]	2773 [2066, 3379]	2699 [2245, 3179]	2856 [2340, 3372]
Humanoid-v5	6063 [5853, 6273]	4400 [3872, 4924]	383 [266, 556]	2568 [1771, 3249]	7207 [5950, 8147]	9938 [9577, 10251]
Walker2d-v5	6009 [5812, 6206]	2742 [1953, 3554]	980 [611, 1555]	3021 [475, 5567]	5431 [4793, 5998]	6077 [5661, 6375]
Aggregate Results						
Mean	1.48	1.16	0.36	0.66	1.50	1.79
Median	1.24	1.30	0.20	0.62	1.12	1.26
IQM	1.44	1.14	0.20	0.68	1.36	1.50

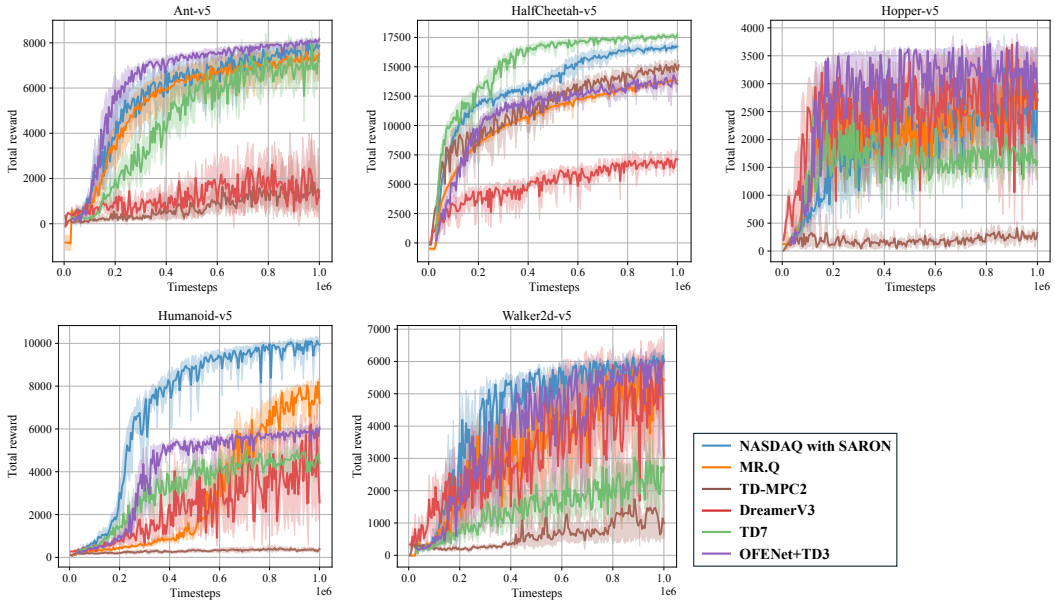


Figure 5: Learning curves on **Gym**. Results are over 5 seeds. The shaded area captures a 95% bootstrap confidence interval.

C.2 DMC (proprioceptive)

Table 14: Final performance on the **DMC (proprioceptive)** benchmark at 500k time steps (1M time steps in the original environment due to an action repeat of 2), averaged over 5 seeds. Results for TD-MPC2 and DreamerV3 are obtained from MR.Q. The [bracketed values] represent a 95% bootstrap confidence interval.

Tasks	OFENet+TD3	TD7	TD-MPC2	DreamerV3	MR.Q	NASDAQ with SARON
acrobot-swingup	40 [18, 62]	257 [228, 285]	584 [551, 615]	230 [193, 266]	234 [201, 267]	242 [200, 279]
ball_in_cup-catch	981 [975, 986]	982 [980, 986]	984 [982, 986]	968 [965, 973]	982 [980, 983]	981 [978, 983]
cartpole-balance	985 [972, 997]	996 [994, 998]	996 [995, 998]	998 [997, 1000]	998 [996, 999]	996 [994, 998]
cartpole-balance_sparse	678 [325, 1000]	1000 [1000, 1000]	1000 [1000, 1000]	999 [999, 1000]	966 [898, 1000]	1000 [1000, 1000]
cartpole-swingup	867 [858, 875]	858 [844, 871]	875 [870, 880]	736 [591, 838]	880 [879, 881]	881 [880, 882]
cartpole-swingup_sparse	168 [0, 504]	513 [183, 833]	845 [839, 849]	702 [560, 792]	757 [588, 843]	827 [814, 837]
cheetah-run	908 [903, 913]	832 [670, 918]	917 [915, 920]	699 [655, 744]	892 [877, 905]	920 [918, 921]
dog-run	66 [22, 110]	176 [146, 210]	265 [166, 342]	4 [4, 5]	300 [281, 319]	296 [272, 315]
dog-stand	347 [268, 466]	935 [924, 945]	506 [266, 715]	22 [20, 27]	946 [935, 956]	944 [933, 955]
dog-trot	98 [73, 127]	186 [105, 261]	407 [265, 530]	10 [6, 17]	593 [516, 676]	645 [552, 738]
dog-walk	144 [112, 176]	687 [444, 842]	486 [240, 704]	17 [15, 21]	773 [724, 819]	807 [761, 856]
finger-spin	977 [973, 982]	794 [415, 987]	986 [986, 988]	666 [577, 763]	981 [978, 983]	979 [974, 982]
finger-turn_easy	771 [535, 968]	826 [513, 984]	979 [975, 983]	906 [883, 927]	968 [947, 980]	933 [890, 971]
finger-turn_hard	466 [234, 698]	427 [81, 777]	947 [916, 977]	864 [812, 900]	973 [968, 978]	959 [936, 973]
fish-swim	151 [103, 190]	223 [118, 381]	659 [615, 706]	813 [808, 819]	541 [469, 609]	803 [793, 813]
hopper-hop	236 [182, 290]	247 [57, 437]	425 [368, 500]	116 [66, 165]	329 [302, 361]	284 [268, 300]
hopper-stand	930 [922, 938]	382 [1, 762]	952 [944, 958]	747 [669, 806]	948 [926, 961]	949 [944, 953]
humanoid-run	57 [13, 101]	109 [54, 142]	181 [121, 231]	0 [1, 1]	148 [128, 169]	186 [170, 204]
humanoid-stand	112 [7, 221]	656 [543, 770]	658 [506, 745]	5 [5, 6]	811 [731, 876]	860 [797, 909]
humanoid-walk	156 [8, 312]	547 [449, 658]	754 [725, 791]	1 [1, 2]	719 [630, 809]	622 [606, 639]
pendulum-swingup	380 [60, 700]	211 [19, 532]	846 [830, 862]	774 [740, 802]	832 [792, 864]	822 [808, 838]
quadruped-run	852 [745, 918]	918 [867, 947]	942 [938, 947]	130 [92, 169]	944 [938, 951]	935 [909, 942]
quadruped-walk	929 [920, 937]	948 [931, 963]	963 [959, 967]	193 [137, 243]	962 [956, 968]	960 [954, 966]
reacher-easy	977 [973, 979]	969 [931, 989]	983 [980, 986]	966 [964, 970]	952 [917, 983]	983 [982, 985]
reacher-hard	956 [935, 970]	976 [972, 979]	960 [936, 979]	919 [864, 955]	953 [927, 976]	978 [974, 983]
walker-run	714 [668, 759]	847 [843, 852]	854 [851, 859]	510 [430, 588]	790 [772, 806]	794 [786, 803]
walker-stand	979 [973, 984]	991 [988, 994]	991 [990, 994]	941 [934, 948]	990 [988, 992]	987 [985, 989]
walker-walk	959 [949, 968]	980 [978, 983]	981 [979, 984]	898 [875, 919]	978 [976, 980]	976 [975, 977]
Aggregate Results						
Mean	567	660	783	530	791	805
Median	696	810	896	700	918	927
IQM	600	726	868	577	886	898

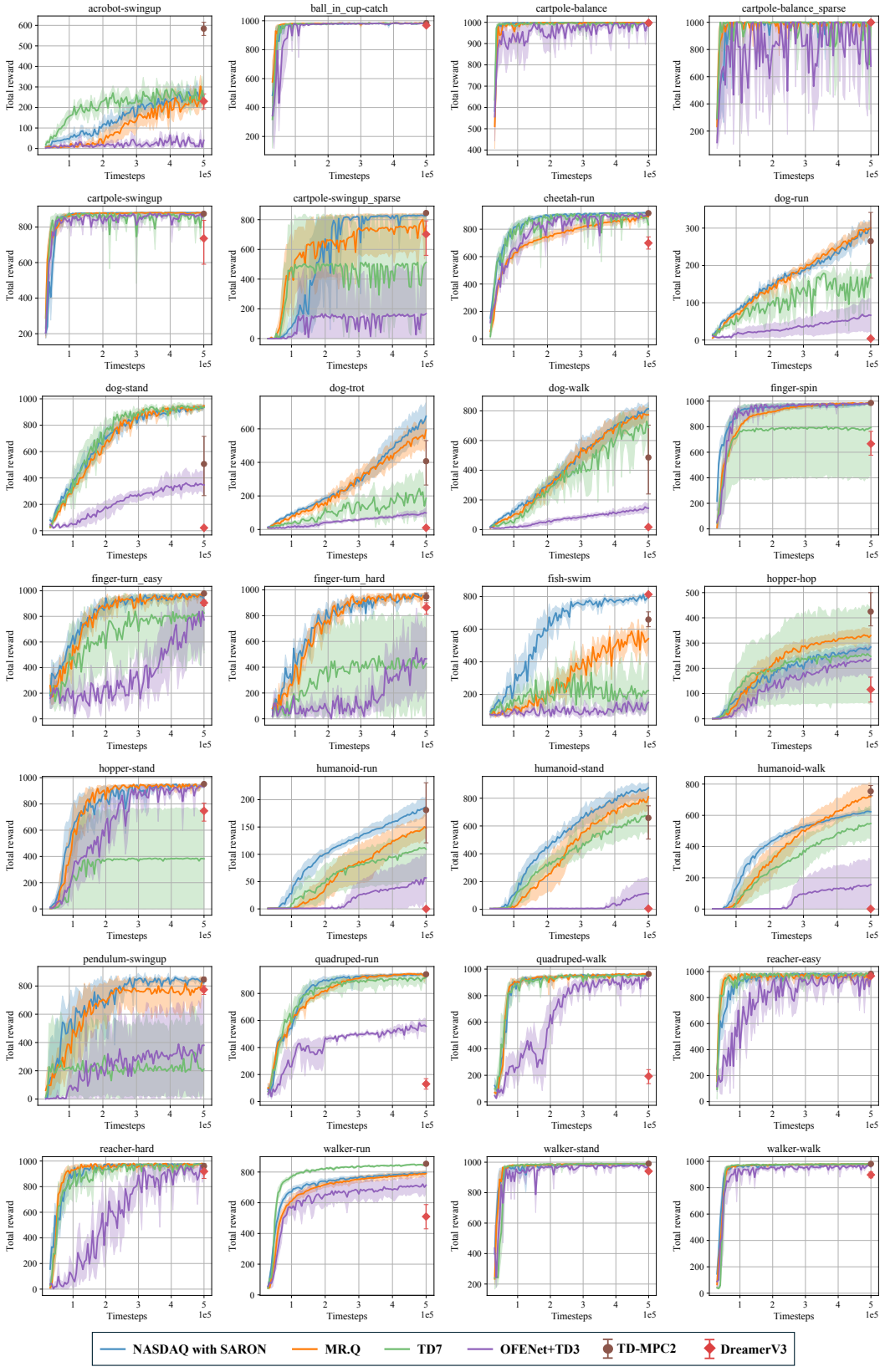


Figure 6: Learning curves on **DMC (proprioceptive)**. Solid lines indicate average performance over 5 seeds, and shaded areas indicate the 95% bootstrap confidence interval. Discrete points with 95% bootstrap confidence interval denote the final results of TD-MPC2 and DreamerV3 reported in MR.Q.

C.3 DMC (visual)

Table 15: Final performance on the **DMC (visual)** benchmark at 500k time steps (1M time steps in the original environment due to action repeat), averaged over 5 seeds. We use the results reported in MR.Q for DrQ-v2, TD-MPC2, and DreamerV3. The [bracketed values] represent a 95% bootstrap confidence interval.

Tasks	DrQ-v2	TD-MPC2	DreamerV3	MR.Q	NASDAQ
acrobot-swingup	168 [127, 219]	197 [179, 217]	121 [106, 145]	315 [268, 362]	264 [171, 356]
ball_in_cup-catch	909 [821, 973]	932 [899, 961]	971 [969, 973]	975 [971, 979]	975 [973, 978]
cartpole-balance	993 [990, 996]	972 [948, 991]	998 [997, 1000]	998 [997, 999]	997 [996, 998]
cartpole-balance_sparse	962 [887, 1000]	1000 [1000, 1000]	999 [999, 1000]	1000 [1000, 1000]	1000 [1000, 1000]
cartpole-swingup	864 [854, 873]	690 [521, 813]	725 [603, 807]	863 [857, 870]	873 [867, 878]
cartpole-swingup_sparse	774 [741, 805]	636 [404, 804]	547 [351, 726]	782 [739, 818]	835 [827, 843]
cheetah-run	728 [701, 753]	431 [267, 556]	618 [576, 661]	759 [752, 768]	853 [821, 885]
dog-run	10 [9, 12]	14 [10, 18]	9 [6, 14]	58 [46, 74]	63 [48, 79]
dog-stand	43 [37, 49]	117 [72, 148]	61 [30, 92]	239 [222, 258]	244 [216, 269]
dog-trot	14 [11, 18]	20 [14, 25]	14 [13, 16]	65 [57, 75]	63 [59, 68]
dog-walk	22 [18, 29]	22 [17, 28]	11 [11, 12]	84 [76, 95]	83 [75, 93]
finger-spin	860 [787, 922]	786 [492, 984]	656 [544, 765]	835 [682, 982]	986 [985, 987]
finger-turn_easy	503 [399, 615]	562 [317, 779]	491 [447, 542]	924 [862, 974]	915 [873, 956]
finger-turn_hard	223 [121, 340]	903 [870, 940]	494 [401, 571]	929 [891, 967]	914 [871, 958]
fish-swim	84 [65, 107]	43 [21, 64]	90 [84, 96]	70 [63, 77]	67 [64, 70]
hopper-hop	224 [170, 278]	187 [119, 238]	205 [125, 287]	254 [204, 299]	283 [256, 308]
hopper-stand	917 [903, 931]	582 [321, 794]	888 [875, 900]	925 [919, 930]	902 [850, 934]
humanoid-run	1 [1, 1]	0 [0, 1]	1 [1, 1]	1 [1, 2]	1 [1, 1]
humanoid-stand	6 [7, 7]	5 [5, 7]	5 [5, 7]	7 [7, 8]	6 [5, 7]
humanoid-walk	2 [2, 2]	1 [1, 2]	1 [2, 2]	3 [2, 5]	2 [2, 3]
pendulum-swingup	838 [813, 861]	748 [574, 850]	761 [709, 807]	852 [832, 874]	836 [816, 858]
quadruped-run	459 [412, 507]	262 [184, 330]	328 [255, 397]	493 [465, 517]	471 [458, 495]
quadruped-walk	750 [699, 796]	246 [179, 310]	316 [260, 379]	856 [800, 912]	809 [748, 855]
reacher-easy	938 [903, 973]	956 [932, 978]	735 [678, 796]	977 [974, 980]	977 [975, 979]
reacher-hard	705 [580, 831]	911 [867, 946]	338 [227, 461]	951 [913, 972]	935 [893, 977]
walker-run	546 [475, 612]	665 [566, 719]	669 [615, 708]	572 [502, 638]	676 [651, 704]
walker-stand	980 [977, 984]	937 [907, 962]	969 [966, 973]	985 [984, 986]	987 [985, 989]
walker-walk	766 [489, 957]	958 [952, 965]	942 [936, 949]	967 [962, 971]	963 [958, 968]
Aggregate Results					
Mean	510	492	463	598	606
Median	626	572	493	809	836
IQM	545	501	452	686	701

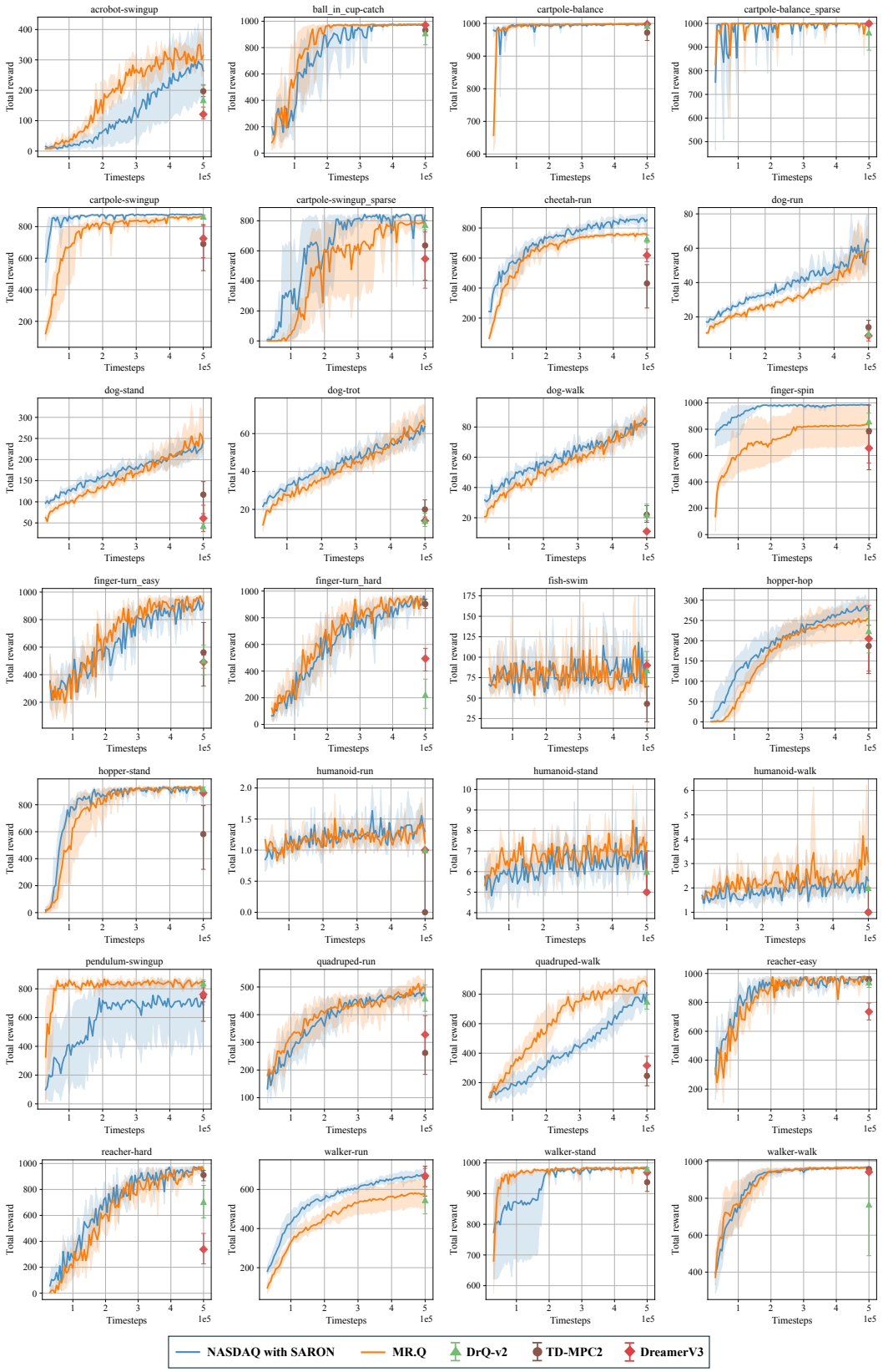


Figure 7: Learning curves on **DMC (visual)**. Solid lines indicate average performance over 5 seeds, and shaded areas indicate the 95% bootstrap confidence interval. Discrete points with 95% bootstrap confidence interval denote the final results of DreamerV3 reported in MR.Q.

C.4 Atari100k

Table 16: Final performance on the **Atari100k** benchmark at 100k time steps (400k time steps in the original environment due to an action repeat of 4), averaged over 5 seeds. The results of DreamerV3 are obtained from the original paper. The [bracketed values] represent a 95% bootstrap confidence interval. The aggregate mean, median, and interquartile mean (IQM) are computed over the human-normalized scores (see Appendix B.1).

Tasks	Rainbow (enhanced)	SPR	MR.Q	NASDAQ	DreamerV3
Alien	728 [665, 791]	923 [734, 1102]	923 [815, 1030]	1183 [951, 1480]	1118
Amidar	223 [203, 246]	213 [181, 255]	179 [149, 210]	158 [113, 203]	97
Assault	695 [600, 807]	671 [617, 726]	601 [574, 634]	676 [640, 715]	683
Asterix	924 [812, 1032]	934 [853, 1021]	1110 [1024, 1195]	1320 [1112, 1514]	1062
BankHeist	63 [44, 82]	180 [34, 432]	23 [19, 27]	35 [22, 50]	398
BattleZone	12422 [10226, 14964]	10634 [7422, 14112]	6350 [4654, 8350]	4880 [2484, 9086]	20300
Boxing	28 [20, 37]	38 [26, 53]	73 [68, 80]	75 [68, 80]	82
Breakout	14 [12, 16]	13 [11, 15]	18 [16, 21]	17 [14, 20]	10
ChopperCommand	1820 [1554, 2093]	1425 [1165, 1685]	1453 [1298, 1619]	1781 [1606, 1953]	2222
CrazyClimber	17741 [16024, 19592]	22957 [20368, 25563]	56564 [51652, 61476]	65146 [53208, 76493]	86225
DemonAttack	1525 [1365, 1677]	1519 [1305, 1732]	804 [687, 933]	2013 [1319, 2721]	577
Freeway	6 [0, 17]	21 [10, 30]	19 [6, 31]	13 [0, 25]	0
Frostbite	1949 [794, 3105]	2862 [2511, 3162]	2296 [1267, 3074]	1939 [997, 2761]	3377
Gopher	615 [482, 748]	500 [373, 644]	739 [625, 853]	806 [676, 936]	2160
Hero	7227 [6212, 8019]	7924 [7353, 8496]	8197 [7206, 9929]	5804 [4613, 6996]	13354
Jamesbond	399 [338, 472]	399 [344, 467]	391 [340, 446]	380 [316, 427]	540
Kangaroo	2262 [977, 4025]	3658 [1719, 5598]	2021 [669, 4368]	476 [259, 648]	2643
Krull	3862 [3199, 4537]	4149 [3603, 4810]	7720 [7049, 8530]	7515 [7061, 7914]	8171
KungFuMaster	13644 [7467, 22636]	14945 [11669, 17586]	17707 [13608, 21533]	15509 [12738, 17771]	25900
MsPacman	1464 [1248, 1648]	1584 [1345, 1941]	1761 [1336, 2186]	1363 [1229, 1498]	1521
Pong	-8 [-14, -2]	-5 [-11, 3]	13 [7, 19]	8 [0, 17]	-4
PrivateEye	80 [40, 100]	86 [59, 100]	3 [0, 10]	77 [37, 100]	3238
Qbert	4015 [3628, 4389]	3259 [2444, 4100]	1907 [928, 3420]	839 [588, 1192]	2921
RoadRunner	10985 [4326, 17396]	11026 [5063, 16555]	12741 [10030, 16573]	13286 [9238, 17276]	19230
Seaquest	560 [418, 670]	556 [508, 605]	576 [531, 605]	1022 [834, 1219]	962
UpNDown	6707 [4963, 8268]	8665 [4665, 13810]	5517 [4213, 6821]	11913 [5180, 24252]	46910
Aggregate Results					
Mean	0.54	0.65	0.91	0.93	1.25
Median	0.29	0.42	0.40	0.34	0.49
IQM	0.35	0.43	0.41	0.42	0.54

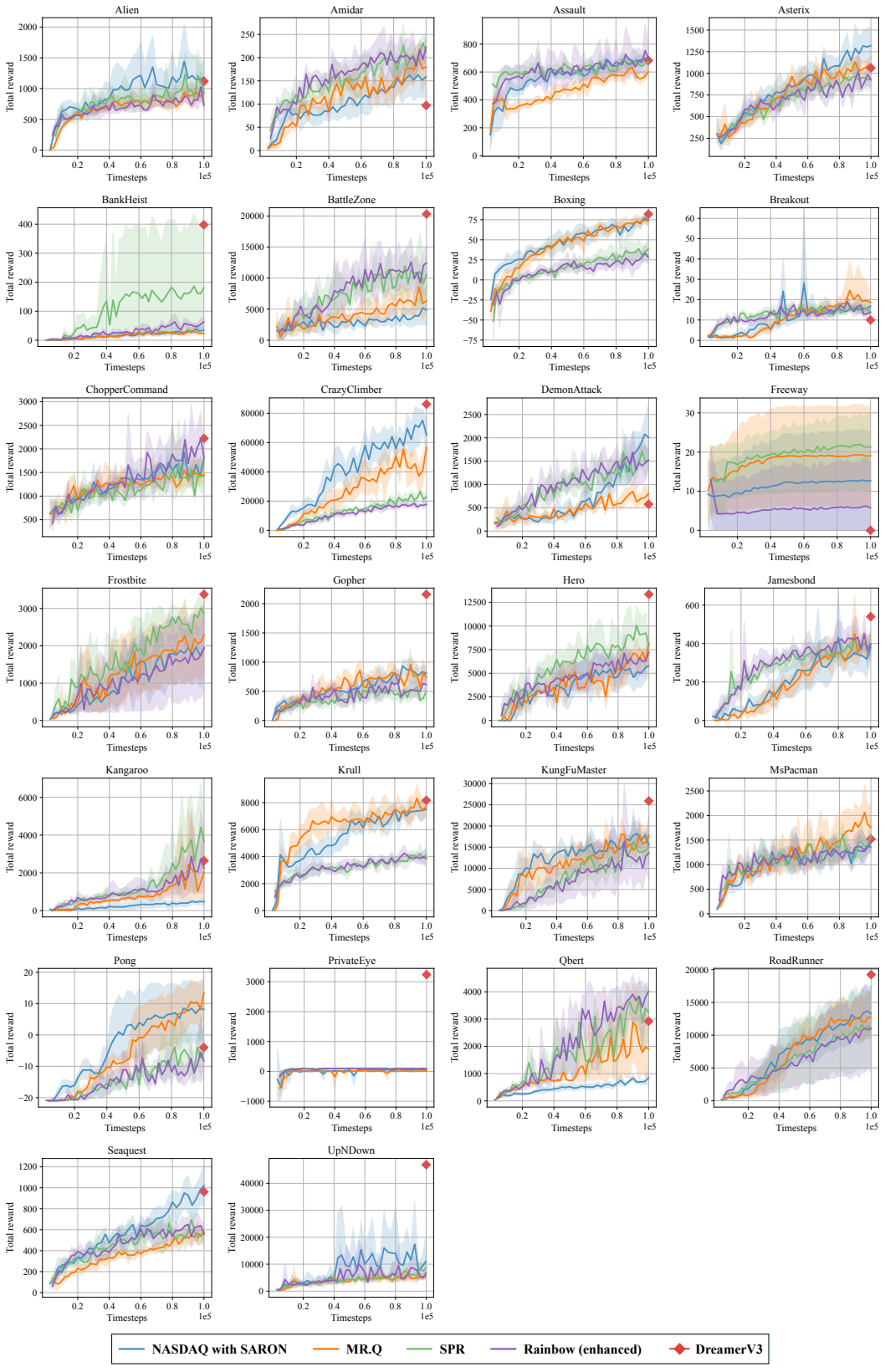


Figure 8: Learning curves on **Atari100k**. Solid lines indicate average performance over 5 seeds, and shaded areas capture the 95% bootstrap confidence interval. Discrete points denote the final results reported in DreamerV3.

C.5 Ablation results of observation normalization

Table 17: Final average results of ablation variants on **Gym** over 5 seeds. The [bracketed values] represent a 95% bootstrap confidence interval.

Tasks	OFENet+TD3	NASDAQ				
	with SARON	w/o ON, w/o Aux	w/o ON, with Aux	with simple ON, w/o Aux	with SARON, w/o Aux	with simple ON
Ant-5	8125 [6912, 9048]	6716 [6263, 7078]	7308 [7011, 7525]	7248 [6972, 7478]	6745 [5824, 7453]	7939 [7842, 8039]
Humanoid-5	8470 [8052, 8835]	4977 [4811, 5317]	502 [432, 571]	9658 [9486, 9821]	9258 [8852, 9664]	10270 [10084, 10475]
HalfCheetah-v5	16998 [16458, 17350]	16233 [16015, 16525]	16458 [16087, 16780]	16374 [16062, 16658]	16597 [16335, 16859]	16626 [16138, 17112]
Hopper-v5	1942 [1764, 2193]	1959 [1608, 2433]	2183 [1662, 2846]	1697 [1449, 1855]	1700 [1582, 1860]	1932 [1659, 2255]
Walker2d-v5	6207 [5765, 6618]	5460 [5279, 5719]	4822 [4338, 5305]	5515 [5097, 5855]	4653 [3318, 5746]	6020 [5479, 6422]
Aggregate Results						
Mean	1.64	1.29	0.99	1.63	1.55	1.75
Median	1.29	1.20	1.00	1.21	1.22	1.25
IQM	1.54	1.35	0.98	1.40	1.30	1.50

Table 18: Final average results of ablated variants on **DMC (proprioceptive)** over 5 seeds. The [bracketed values] represent a 95% bootstrap confidence interval.

Tasks	OFENet+TD3	NASDAQ				
	with SARON	w/o ON, w/o Aux	w/o ON, with Aux	with simple ON, w/o Aux	with SARON, w/o Aux	with simple ON
acrobot-swingup	30 [11, 49]	234 [201, 267]	288 [252, 327]	160 [106, 215]	155 [65, 245]	180 [68, 291]
ball_in_cup-catch	980 [977, 983]	984 [980, 988]	977 [973, 981]	978 [975, 983]	982 [981, 984]	981 [978, 982]
cartpole-balance	939 [859, 997]	997 [995, 999]	998 [997, 999]	998 [997, 999]	997 [997, 998]	998 [995, 999]
cartpole-balance_sparse	997 [992, 1000]	1000 [1000, 1000]	1000 [1000, 1000]	1000 [1000, 1000]	1000 [1000, 1000]	1000 [1000, 1000]
cartpole-swingup	872 [868, 876]	880 [879, 881]	880 [878, 882]	881 [880, 882]	882 [881, 883]	880 [878, 882]
cartpole-swingup_sparse	325 [0, 656]	169 [0, 508]	332 [0, 664]	169 [0, 508]	322 [0, 644]	174 [0, 500]
cheetah-run	911 [907, 915]	890 [845, 914]	883 [824, 916]	917 [914, 921]	918 [915, 921]	920 [919, 921]
dog-run	7 [5, 8]	56 [47, 65]	11 [9, 15]	404 [388, 431]	392 [337, 448]	313 [260, 374]
dog-stand	28 [18, 37]	246 [199, 305]	66 [43, 100]	920 [895, 940]	905 [859, 953]	953 [930, 972]
dog-trot	8 [6, 9]	75 [66, 90]	10 [7, 14]	672 [526, 817]	580 [503, 685]	619 [576, 685]
dog-walk	12 [10, 15]	92 [50, 127]	18 [10, 28]	776 [691, 838]	792 [710, 851]	618 [317, 812]
finger-spin	981 [978, 984]	952 [934, 970]	980 [974, 984]	969 [950, 982]	957 [929, 979]	983 [979, 986]
finger-turn_easy	979 [972, 985]	461 [177, 745]	979 [974, 983]	813 [624, 963]	830 [745, 917]	957 [917, 984]
finger-turn_hard	944 [905, 970]	402 [209, 594]	951 [910, 977]	644 [411, 878]	781 [619, 894]	887 [823, 951]
fish-swim	176 [112, 244]	762 [746, 778]	713 [628, 770]	799 [780, 818]	787 [772, 802]	792 [763, 818]
hopper-hop	244 [167, 361]	177 [96, 259]	261 [248, 275]	202 [157, 242]	159 [115, 202]	256 [221, 291]
hopper-stand	706 [506, 879]	610 [355, 865]	946 [943, 949]	663 [361, 945]	847 [714, 952]	926 [892, 946]
humanoid-run	152 [137, 166]	25 [1, 71]	11 [1, 30]	146 [137, 155]	135 [125, 146]	164 [154, 181]
humanoid-stand	505 [397, 663]	171 [7, 499]	7 [6, 8]	538 [327, 721]	712 [583, 827]	892 [871, 907]
humanoid-walk	310 [112, 508]	84 [2, 247]	107 [1, 221]	519 [495, 553]	534 [505, 566]	677 [603, 789]
pendulum-swingup	227 [2, 488]	841 [826, 856]	859 [842, 871]	205 [20, 516]	557 [243, 871]	870 [856, 884]
quadruped-run	582 [547, 617]	865 [834, 898]	715 [679, 745]	910 [886, 935]	940 [922, 951]	943 [936, 948]
quadruped-walk	948 [933, 961]	951 [938, 964]	918 [899, 937]	960 [954, 965]	945 [933, 955]	958 [952, 963]
reacher-easy	900 [777, 981]	986 [985, 988]	986 [985, 989]	965 [922, 988]	986 [985, 987]	985 [983, 986]
reacher-hard	963 [950, 976]	958 [917, 981]	957 [918, 979]	982 [979, 985]	981 [979, 983]	979 [977, 980]
walker-run	770 [751, 788]	762 [726, 786]	756 [734, 776]	802 [790, 812]	798 [788, 808]	770 [715, 809]
walker-stand	985 [982, 989]	987 [986, 989]	987 [982, 991]	987 [984, 990]	986 [978, 993]	988 [985, 990]
walker-walk	963 [957, 970]	976 [976, 977]	968 [963, 973]	973 [970, 976]	975 [972, 979]	977 [974, 982]
Aggregate Results						
Mean	587	593	627	713	744	773
Median	738	762	870	808	839	906
IQM	656	645	745	804	834	886

C.6 Ablation results of auxiliary tasks

Table 19: Complete ablation results of auxiliary tasks on **Gym**. The [bracketed values] represent a 95% bootstrap confidence interval. The aggregate mean, median, and interquartile mean (IQM) are computed over the Deep-TD3-normalized scores (see Appendix B.1).

Tasks	NASDAQ with SARON			
	$(\lambda_{\text{Rec}} = 0, \lambda_{\text{n-step}} = 0)$	$(\lambda_{\text{Rec}} = 0, \lambda_{\text{n-step}} > 0)$	$(\lambda_{\text{Rec}} > 0, \lambda_{\text{n-step}} = 0)$	$(\lambda_{\text{Rec}} > 0, \lambda_{\text{n-step}} > 0)$
Ant-5	6745 [5824, 7453]	6078 [4976, 7138]	7871 [7664, 8103]	7813 [7618, 8011]
Humanoid-5	9258 [8852, 9664]	9399 [8740, 9992]	9938 [9577, 10251]	10392 [10280, 10509]
HalfCheetah-5	16597 [16335, 16859]	16170 [15071, 16881]	16742 [16476, 17031]	16444 [16090, 16749]
Hopper-5	1700 [1582, 1860]	2837 [2588, 3085]	2856 [2340, 3372]	2412 [1883, 3022]
Walker2d-5	4653 [3318, 5746]	5674 [5181, 6082]	6077 [5661, 6375]	5853 [5400, 6191]
Aggregate Results				
Mean	1.55	1.64	1.79	1.78
Median	1.22	1.19	1.26	1.21
IQM	1.30	1.31	1.50	1.47

Table 20: Complete ablation results of auxiliary tasks on **DMC (proprioceptive)**. The [bracketed values] represent a 95% bootstrap confidence interval.

Tasks	**NASDAQ with SARON**			
	$(\lambda_{\text{Rec}} = 0, \lambda_{\text{n-step}} = 0)$	$(\lambda_{\text{Rec}} = 0, \lambda_{\text{n-step}} > 0)$	$(\lambda_{\text{Rec}} > 0, \lambda_{\text{n-step}} = 0)$	$(\lambda_{\text{Rec}} > 0, \lambda_{\text{n-step}} > 0)$
acrobot-swingup	155 [65, 245]	47 [14, 87]	242 [200, 279]	259 [221, 297]
ball_in_cup-catch	982 [981, 984]	981 [979, 984]	981 [978, 983]	979 [976, 984]
cartpole-balance	997 [997, 998]	992 [981, 999]	996 [994, 998]	998 [996, 999]
cartpole-balance_sparse	1000 [1000, 1000]	1000 [1000, 1000]	1000 [1000, 1000]	1000 [1000, 1000]
cartpole-swingup	882 [881, 883]	882 [881, 883]	881 [880, 882]	878 [870, 882]
cartpole-swingup_sparse	322 [0, 644]	334 [0, 671]	827 [814, 837]	324 [0, 655]
cheetah-run	918 [915, 921]	917 [915, 919]	920 [918, 921]	918 [917, 920]
dog-run	392 [337, 448]	338 [295, 380]	296 [272, 315]	294 [279, 311]
dog-stand	905 [857, 953]	910 [887, 933]	944 [933, 955]	926 [891, 954]
dog-trot	580 [503, 685]	611 [474, 750]	645 [552, 738]	672 [594, 755]
dog-walk	792 [710, 851]	793 [745, 846]	807 [761, 856]	808 [754, 864]
finger-spin	957 [929, 979]	950 [904, 981]	979 [974, 982]	982 [980, 985]
finger-turn_easy	830 [745, 917]	790 [491, 965]	933 [890, 971]	975 [968, 982]
finger-turn_hard	781 [619, 894]	763 [635, 891]	959 [936, 973]	973 [969, 977]
fish-swim	787 [772, 802]	756 [680, 806]	803 [793, 813]	792 [768, 816]
hopper-hop	159 [115, 202]	228 [177, 262]	284 [268, 300]	176 [109, 232]
hopper-stand	847 [714, 952]	794 [607, 946]	949 [944, 953]	814 [553, 946]
humanoid-run	135 [125, 146]	146 [135, 156]	186 [170, 204]	159 [153, 165]
humanoid-stand	712 [583, 827]	573 [276, 799]	860 [797, 909]	809 [683, 921]
humanoid-walk	534 [505, 566]	520 [443, 608]	622 [606, 639]	651 [586, 761]
pendulum-swingup	557 [243, 871]	399 [80, 719]	822 [808, 838]	348 [20, 675]
quadruped-run	940 [922, 951]	936 [916, 952]	935 [909, 942]	943 [938, 948]
quadruped-walk	945 [933, 955]	962 [956, 969]	960 [954, 966]	968 [958, 976]
reacher-easy	986 [985, 987]	985 [982, 988]	983 [982, 985]	984 [981, 987]
reacher-hard	981 [979, 983]	961 [922, 981]	978 [974, 983]	959 [914, 983]
walker-run	798 [788, 808]	794 [778, 808]	794 [786, 803]	795 [782, 807]
walker-stand	986 [978, 993]	988 [984, 990]	987 [985, 989]	988 [986, 991]
walker-walk	975 [972, 979]	976 [974, 979]	976 [975, 977]	975 [970, 980]
Aggregate Results				
Mean	744	726	805	762
Median	839	794	927	898
IQM	834	816	898	874

Table 21: Complete ablation results of auxiliary tasks on **DMC (visual)**. The [bracketed values] represent a 95% bootstrap confidence interval.

Tasks	**NASDAQ**			
	$(\lambda_{\text{Rec}} = 0, \lambda_{\text{n-step}} = 0)$	$(\lambda_{\text{Rec}} = 0, \lambda_{\text{n-step}} > 0)$	$(\lambda_{\text{Rec}} > 0, \lambda_{\text{n-step}} = 0)$	$(\lambda_{\text{Rec}} > 0, \lambda_{\text{n-step}} > 0)$
acrobot-swingup	312 [227, 377]	296 [233, 354]	331 [250, 412]	264 [171, 356]
ball_in_cup-catch	700 [351, 973]	955 [922, 977]	971 [965, 977]	975 [973, 978]
cartpole-balance	997 [995, 999]	996 [992, 999]	996 [991, 999]	997 [996, 998]
cartpole-balance_sparse	998 [993, 1000]	997 [992, 1000]	997 [990, 1000]	1000 [1000, 1000]
cartpole-swingup	873 [864, 880]	876 [869, 880]	878 [875, 880]	873 [867, 878]
cartpole-swingup_sparse	0 [0, 0]	0 [0, 0]	841 [837, 845]	835 [827, 843]
cheetah-run	705 [671, 738]	761 [741, 781]	812 [766, 858]	853 [821, 885]
dog-run	25 [23, 26]	31 [29, 34]	54 [49, 61]	63 [48, 79]
dog-stand	133 [127, 142]	160 [150, 169]	195 [177, 213]	244 [216, 269]
dog-trot	35 [32, 38]	42 [40, 44]	72 [65, 79]	63 [59, 68]
dog-walk	48 [45, 51]	56 [51, 62]	88 [77, 97]	83 [75, 93]
finger-spin	522 [131, 913]	908 [779, 980]	983 [981, 985]	986 [985, 987]
finger-turn_easy	640 [523, 930]	587 [362, 786]	902 [817, 968]	915 [873, 956]
finger-turn_hard	330 [193, 467]	713 [543, 817]	881 [802, 950]	914 [871, 958]
fish-swim	75 [63, 91]	64 [55, 73]	84 [61, 111]	67 [64, 70]
hopper-hop	62 [0, 186]	27 [5, 54]	276 [258, 298]	283 [256, 308]
hopper-stand	144 [5, 413]	825 [700, 916]	923 [915, 931]	902 [850, 934]
humanoid-run	1 [1, 1]	1 [1, 1]	1 [1, 1]	1 [1, 1]
humanoid-stand	6 [6, 7]	6 [6, 7]	8 [7, 9]	6 [5, 7]
humanoid-walk	2 [2, 2]	2 [2, 2]	3 [3, 4]	2 [2, 3]
pendulum-swingup	581 [283, 850]	684 [385, 843]	592 [349, 807]	836 [816, 858]
quadruped-run	413 [394, 431]	397 [344, 449]	463 [459, 467]	471 [458, 495]
quadruped-walk	662 [557, 729]	730 [632, 820]	772 [706, 839]	809 [748, 855]
reacher-easy	974 [971, 976]	978 [975, 980]	938 [867, 976]	977 [975, 979]
reacher-hard	178 [68, 290]	888 [777, 966]	931 [852, 973]	935 [893, 977]
walker-run	501 [464, 528]	538 [513, 564]	554 [423, 653]	676 [651, 704]
walker-stand	973 [969, 977]	979 [977, 982]	986 [983, 989]	987 [985, 989]
walker-walk	912 [836, 955]	959 [953, 964]	964 [962, 966]	963 [958, 968]
Aggregate Results				
Mean	422	516	589	606
Median	372	635	792	836
IQM	375	541	668	701

Table 22: Complete ablation results of auxiliary tasks on **Atari100k**. The [bracketed values] represent a 95% bootstrap confidence interval. The aggregate mean, median, and interquartile mean (IQM) are computed over the human-normalized score (see Appendix B.1).

Tasks	**NASDAQ**			
	$(\lambda_{\text{Rec}} = 0, \lambda_{\text{n-step}} = 0)$	$(\lambda_{\text{Rec}} = 0, \lambda_{\text{n-step}} > 0)$	$(\lambda_{\text{Rec}} > 0, \lambda_{\text{n-step}} = 0)$	$(\lambda_{\text{Rec}} > 0, \lambda_{\text{n-step}} > 0)$
Alien	900 [824, 970]	1009 [865, 1225]	915 [775, 1032]	1183 [951, 1480]
Amidar	149 [135, 161]	172 [154, 191]	154 [137, 175]	158 [113, 203]
Assault	431 [298, 565]	599 [565, 632]	637 [604, 672]	676 [640, 715]
Asterix	899 [737, 1060]	1155 [1078, 1233]	1235 [1095, 1375]	1320 [1112, 1514]
BankHeist	20 [10, 28]	34 [22, 45]	30 [21, 42]	35 [22, 50]
BattleZone	5544 [2180, 12128]	11274 [7810, 15188]	3058 [2670, 3540]	4880 [2484, 9086]
Boxing	24 [21, 30]	57 [50, 64]	65 [60, 70]	75 [68, 80]
Breakout	1 [1, 1]	10 [4, 15]	18 [15, 22]	17 [14, 20]
ChopperCommand	1679 [1404, 1926]	1822 [1464, 2179]	1682 [1340, 1964]	1781 [1606, 1953]
CrazyClimber	23282 [21142, 25339]	45939 [40455, 52334]	57314 [35783, 73445]	65146 [53208, 76493]
DemonAttack	160 [153, 165]	1110 [951, 1295]	1821 [1580, 2039]	2013 [1319, 2721]
Freeway	25 [12, 31]	12 [0, 24]	0 [0, 0]	13 [0, 25]
Frostbite	1612 [1478, 1766]	1610 [975, 2042]	2211 [1241, 2856]	1939 [997, 2761]
Gopher	406 [339, 498]	609 [551, 670]	673 [592, 763]	806 [676, 936]
Hero	3570 [2733, 5117]	4317 [3593, 5089]	3897 [2790, 6852]	5804 [4613, 6996]
Jamesbond	117 [31, 289]	311 [173, 420]	349 [299, 400]	380 [316, 427]
Kangaroo	1978 [147, 5211]	2663 [582, 6120]	94 [7, 222]	476 [259, 648]
Krull	3111 [2898, 3288]	4769 [4344, 5158]	6596 [6093, 7121]	7515 [7061, 7914]
KungFuMaster	8569 [6936, 10431]	10781 [8827, 13202]	11372 [9778, 12967]	15509 [12738, 17771]
MsPacman	1435 [1130, 1776]	1357 [1144, 1529]	1166 [999, 1324]	1363 [1229, 1498]
Pong	-2 [-9, 7]	7 [-1, 17]	10 [0, 18]	8 [0, 17]
PrivateEye	40 [0, 80]	-30 [-256, 100]	20 [0, 60]	77 [37, 100]
Qbert	2074 [1165, 2986]	2418 [1704, 3180]	697 [637, 750]	839 [588, 1192]
RoadRunner	8662 [1869, 15455]	14065 [9364, 18572]	11611 [9485, 15026]	13286 [9238, 17276]
Seaquest	442 [366, 527]	655 [540, 767]	798 [699, 900]	1022 [834, 1219]
UpNDown	4580 [4224, 4961]	8085 [5485, 10684]	28976 [18459, 38070]	11913 [5180, 24252]
Aggregate Results				
Mean	0.38	0.71	0.86	0.93
Median	0.16	0.32	0.16	0.34
IQM	0.23	0.38	0.36	0.42

C.7 Results on the diagnostic set of tasks

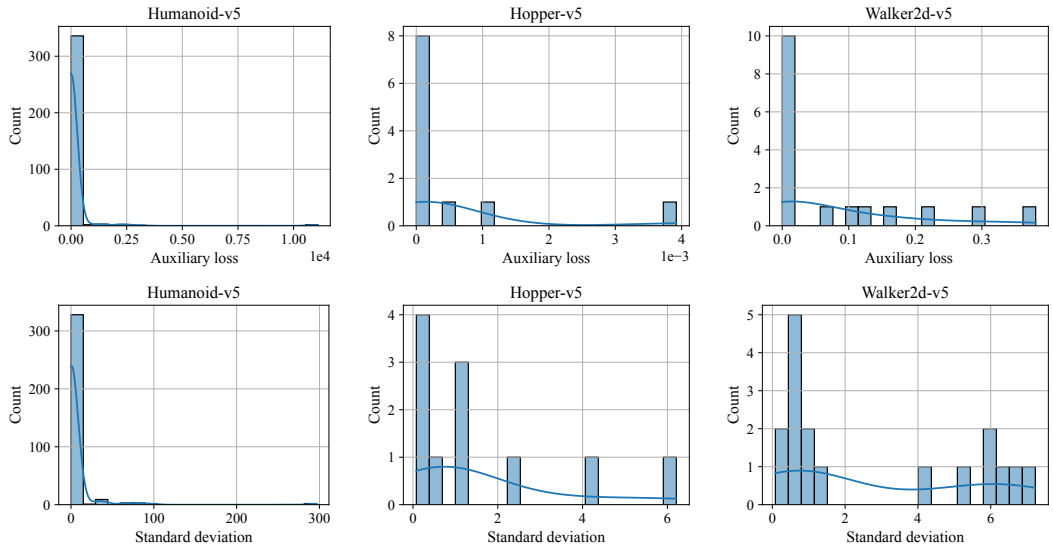


Figure 9: Histograms and kernel density-estimated PDFs of per-dimension statistics, obtained from a single run of **OFENet+TD3** on three tasks. *Top*: distribution of auxiliary losses for each observation dimension, where the value for each dimension is computed by averaging the recorded auxiliary losses over the final 20k time steps. *Bottom*: distribution of standard deviation over dimensions, computed from the final 20k samples.

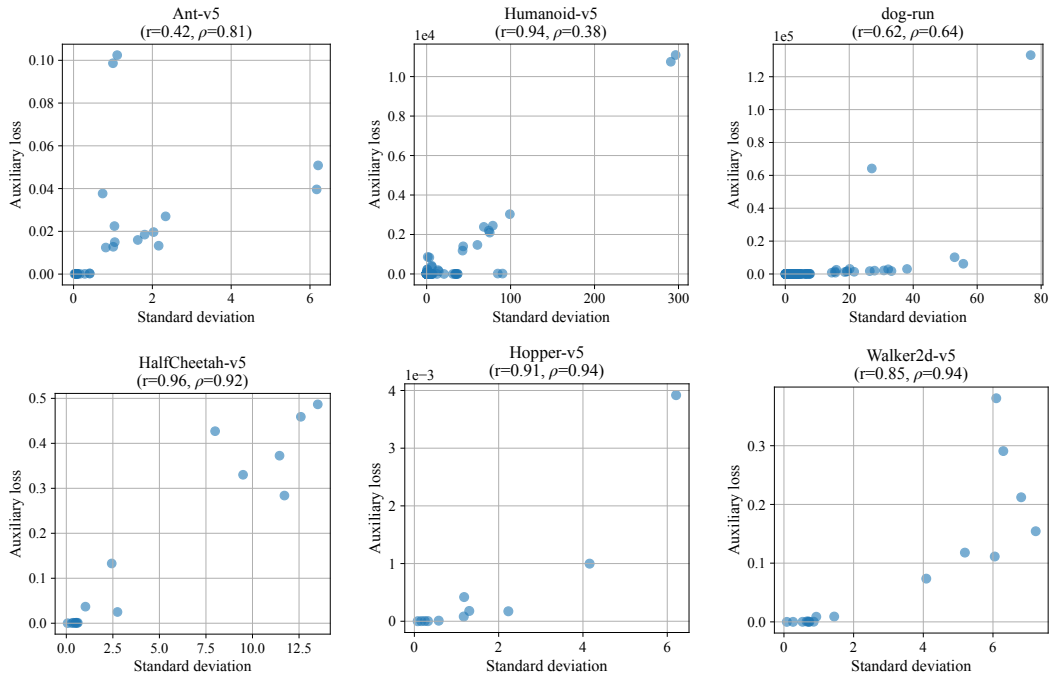


Figure 10: Scatter plots of auxiliary loss versus standard deviation for each observation dimension on the diagnostic set of tasks. Data is obtained from a single run of **OFENet+TD3**. Each point corresponds to one observation dimension. Subplot titles report the Spearman rank (ρ) and Pearson correlation coefficient (r), quantifying the monotonic and linear relationships, respectively.

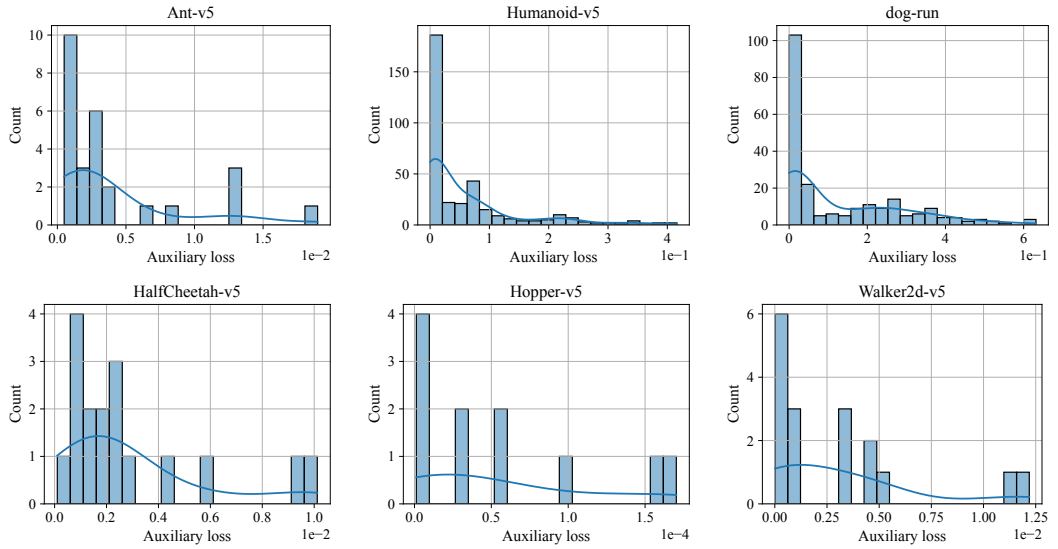


Figure 11: Histograms and kernel density estimated PDFs of per-dimension auxiliary loss across the diagnostic set of tasks. Results are obtained from a single run of **OFENet+TD3 with SARON**. The value for each dimension is computed by averaging the recorded auxiliary losses over the final 20k time steps.

D Additional analysis

D.1 Beyond aggregate metrics—a fine-grained analysis

We divide the tasks in the Gym and DMC benchmarks into different difficulty levels. Specifically, following the categorization in [22], we divide the DMC tasks into three difficulty levels (easy, medium, and hard) in Table 23. Since [22] does not discuss the difficulty of the fish-swim and dog tasks, we categorize them as hard based on the poor performance of RL baselines on these tasks under the visual setting. For the Gym benchmark, we associate each Gym task with the most similar DMC task in the table above. Under this categorization, Humanoid is classified as hard, while all other Gym tasks are classified as medium.

For Atari, we additionally report the SPR-normalized score, which is computed in a manner similar to the human-normalized score. Intuitively, this metric assigns less credit to improvements on games where SPR already performs strongly and more credit to improvements on games where SPR performs poorly. It therefore provides an additional view emphasizing tasks that are difficult from the perspective of a strong self-predictive RL baseline.

Based on the task-categorization protocol and evaluation metric described above, we organize and analyze our experimental results at a finer granularity. We summarize the aggregated results of NASDAQ with SARON and the other baselines on Gym, DMC, and Atari. The numbers in parentheses are the number of tasks included in each difficulty level. Bold indicates the best result, and italics indicate the second-best result.

Overall, the advantages of NASDAQ with SARON over the other baselines are mainly concentrated on the medium and hard task subsets. On the easy tasks, it either achieves the best performance or remains close to the best-performing method. These results indicate that NASDAQ with SARON maintains strong performance on simple tasks while being effective on complex control tasks. NASDAQ also outperforms MR.Q in terms of the SPR-normalized score. This is, to some extent, consistent with the results above, with the medium and hard Atari tasks defined according to SPR's performance.

In addition, the advantage of NASDAQ with SARON over NASDAQ with simple ON is also mainly concentrated on the medium and hard task subsets. Considering the differences between SARON and simple ON, we hypothesize that RL agents experience more substantial distributional shifts in their

Table 23: Task difficulty categorization.

DMC Task	Gym Task	Difficulty
cartpole-balance		Easy
cartpole-balance_sparse		Easy
cartpole-swingup		Easy
ball_in_cup-catch		Easy
finger-spin		Easy
hopper-stand		Easy
pendulum-swingup		Easy
walker-stand		Easy
walker-walk		Easy
acrobot-swingup		Medium
cartpole-swingup_sparse		Medium
cheetah-run	HalfCheetah	Medium
finger-turn_easy		Medium
finger-turn_hard		Medium
hopper-hop	Hopper	Medium
quadruped-run	Ant	Medium
quadruped-walk		Medium
reacher-easy		Medium
reacher-hard		Medium
walker-run	Walker2d	Medium
fish-swim		Hard
3 humanoid tasks	Humanoid	Hard
4 dog tasks		Hard

observations while progressing from being unable to solve medium and hard tasks to successfully solving them, compared with the shifts experienced on easy tasks.

These findings suggest that NASDAQ with SARON is particularly attractive for medium and hard tasks, providing evidence that this combination effectively addresses the bottleneck motivating this work, namely that observation-predictive RL tends to struggle on challenging tasks with low-dimensional observations.

Table 24: Mean performance across difficulty subsets on Gym.

Task	OFENet+TD3	TD7	TD-MPC2	DreamerV3	MR.Q	NASDAQ with SARON
Medium (4)	1.19	1.06	0.43	0.61	1.24	**1.36**
Hard (1)	2.13	1.54	0.10	0.88	2.54	**3.50**

Table 25: Mean performance across difficulty subsets on DMC (proprioceptive).

Task	OFENet+TD3	TD7	TD-MPC2	DreamerV3	MR.Q	NASDAQ with SARON
Easy (9)	860	800	**957**	859	951	953
Medium (11)	642	713	**865**	575	812	816
Hard (8)	142	440	490	109	604	**652**

Table 26: Mean performance across difficulty subsets on DMC (visual).

Task	DrQ-v2	TD-MPC2	DreamerV3	MR.Q	NASDAQ
Easy (9)	899	845	879	**933**	930
Medium (11)	573	562	460	721	**742**
Hard (8)	23	28	24	66	**67**

Table 27: SPR-normalized scores on Atari100k.

Metric	Rainbow (enhanced)	MR.Q	NASDAQ	DreamerV3
Mean	0.92	1.13	1.26	**3.93**
IQM	0.93	0.99	1.13	**1.56**

D.2 Ablation of SARON components

Table 28 reports the results of our ablation study of SARON across both low-dimensional benchmarks. All variants are evaluated within the NASDAQ framework.

DMC-easy, DMC-medium, and DMC-hard denote the task subsets at the easy, medium, and hard difficulty levels, respectively. All Gym tasks except Humanoid are classified as medium, while Humanoid is classified as hard. Since the hard subset of Gym contains only one task, we report aggregate results over all Gym tasks rather than separate difficulty-level results. Bold numbers indicate the best results.

Table 28: Aggregate performance of different observation normalization variants within NASDAQ. Gym scores are normalized by Deep-TD3 before aggregation, whereas DMC scores are raw returns.

Variant	Gym		DMC-all		DMC-easy	DMC-medium	DMC-hard
	Mean	IQM	Mean	IQM	Mean	Mean	Mean
EMA + bias correction + return-based IQR filtering (SARON)	**1.79**	**1.50**	**805**	**898**	953	816	**652**
EMA + bias correction	1.73	1.45	799	**898**	944	**818**	647
EMA only	1.72	1.46	756	890	932	777	594
w/o ON	0.99	0.98	627	745	955	750	118
Simple SB3 normalization	1.75	**1.50**	773	886	**956**	749	629

Introducing EMA-based observation normalization (+ EMA) over the variant without observation normalization (w/o ON) leads to significant improvements, although it still performs slightly worse overall than simple SB3 normalization on both benchmarks.

Adding bias correction to EMA-based observation normalization (+ EMA + bias correction) further improves the performance on the DMC benchmark, where the variant achieves higher aggregate results than simple SB3 normalization.

Further adding return-based IQR filtering, resulting in the complete SARON, improves the aggregate performance on DMC and enables SARON to achieve higher aggregate performance than simple SB3 normalization on Gym.

In summary, EMA alone is insufficient for EMA-based observation normalization to achieve the best or near-best performance on these two benchmarks. Bias correction and return-based IQR filtering provide effective and complementary improvements. Although bias correction does not yield a clear improvement on the Gym benchmark, its benefit is reflected on DMC. Conversely, while the improvement from return-based IQR filtering on DMC is primarily observed in the easy-task subset, it also improves the average performance on Gym. Using these components together supports SARON's overall performance across these benchmarks, as well as its potential performance on a broader range of benchmarks.

D.3 Sensitivity to the effective window size

We evaluated several candidate values for SARON's effective window size on a small subset of tasks and selected the hyperparameter value based on these results. Table 29 summarizes the per-task raw scores of NASDAQ with SARON on this task subset for effective window sizes ranging from 50k to 500k. The dog-run and humanoid-run tasks are from DMC (proprioceptive), while Ant and Humanoid are from Gym. Values in brackets denote 95% bootstrap confidence intervals, and bold values indicate the best performance. The results show that NASDAQ with SARON is robust to large variations in the effective window size. In addition, the preferred effective window size varies across tasks, ranging from 50k to 500k. We therefore selected the intermediate value of 200k as the default setting.

Table 29: Sensitivity of NASDAQ with SARON to the effective window size.

Task	50k	100k	200k	300k	500k
dog-run	**325** [267, 388]	287 [251, 332]	296 [272, 315]	309 [273, 345]	248 [131, 317]
humanoid-run	166 [154, 179]	172 [156, 187]	**186** [170, 204]	154 [137, 166]	152 [139, 161]
Ant	7747 [7039, 8225]	7806 [7453, 8086]	7871 [7664, 8103]	8138 [8077, 8202]	**8167** [7904, 8469]
Humanoid	9993 [9698, 10288]	9944 [9696, 10169]	9938 [9577, 10251]	9851 [9514, 10136]	**10071** [9622, 10451]

E Discussion of aleatoric and epistemic uncertainties

The diagnostic experiments in Table 1 are conducted on MuJoCo-based Gym and DMC proprioceptive tasks, whose simulator dynamics are deterministic given the full state and action. However, the agent's low-dimensional observations may not be fully Markovian, so the next observation o_{t+1} may remain ambiguous given only o_t and a_t. Consequently, the prediction loss may reflect both epistemic uncertainty from imperfect learning and aleatoric uncertainty arising from partial observability.

We discuss below how aleatoric and epistemic uncertainties relate to our empirical findings. For observation dimension d, let $O_{t+1,d}$ be the d-th dimension of the next-observation random variable and Z_t the state-action representation at time t. We define the conditional mean as $m_d(z) := \mathbb{E}[O_{t+1,d} \mid Z_t = z]$ and denote the decoder prediction by $\hat{m}_d(z)$. Assuming homoscedastic residuals within each observation dimension, we write

$$O_{t+1,d} = m_d(Z_t) + \varepsilon_{t,d}, \qquad \mathbb{E}[\varepsilon_{t,d} \mid Z_t] = 0, \quad \mathrm{Var}(\varepsilon_{t,d} \mid Z_t) = \sigma^2_{\varepsilon,d},$$

where $\sigma^2_{\varepsilon,d}$ may differ across observation dimensions but does not depend on z. The marginal variance of dimension d satisfies

$$\mathrm{Var}(O_{t+1,d}) = \mathrm{Var}\left(m_d(Z_t)\right) + \sigma^2_{\varepsilon,d}.$$

Here, $\mathrm{Var}(m_d(Z_t))$ captures the systematic variation in the conditional mean induced by changes in the state-action representation, rather than the residual randomness that remains after conditioning on Z_t. In contrast, $\sigma^2_{\varepsilon,d}$ represents the irreducible variation due to aleatoric uncertainty. The expected prediction loss (MSE) can be decomposed as

$$\mathrm{MSE}_d := \mathrm{E}\left[\left(O_{t+1,d} - \hat{m}_d(Z_t)\right)^2\right] = \sigma^2_{\varepsilon,d} + \mathrm{E}\left[\left(m_d(Z_t) - \hat{m}_d(Z_t)\right)^2\right].$$

Thus, the MSE contains both irreducible aleatoric error and reducible prediction error. When the decoder accurately estimates the conditional mean, $\mathrm{MSE}_d \approx \sigma^2_{\varepsilon,d}$. Therefore, the aleatoric variance $\sigma^2_{\varepsilon,d}$ contributes to both the marginal variance of an observation dimension and its prediction loss. If the variance of the conditional mean, $\mathrm{Var}(m_d(Z_t))$, is broadly comparable across dimensions, dimensions with larger aleatoric variances will tend to have both larger marginal standard deviations and larger MSEs. This provides one possible explanation consistent with the positive correlation between a dimension's value range, measured by its standard deviation, and its auxiliary loss (MSE) observed in our experiments.

If $\mathrm{Var}(m_d(Z_t))$ varies substantially across dimensions, however, the marginal standard deviation is no longer primarily determined by aleatoric variance, weakening both its monotonic associations with the MSE. This may partly explain the lower Spearman and Pearson correlations observed in some environments plotted in Figure 10 of the paper.

F Broader impacts

Our proposed methods, SARON and NASDAQ, enable observation-predictive RL to match or exceed state-of-the-art model-based and self-predictive RL in performance while requiring significantly less training wall-time. This could help reduce energy consumption and carbon emissions when training agents. In addition, our experimental design promotes fairer comparisons among methods that share underlying principles (e.g., fine-grained alignment of design choices and parameter control described in Section 6). We hope this inspires the community to adopt more rigorous and fair experimental practices.

As with any work that improves efficiency, our methods lower the barrier to using RL, which may lead to an increased risk of misuse. We acknowledge that improvements in efficiency should be accompanied by advances in social oversight and governance mechanisms.

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: The abstract and introduction clearly state our contributions and scope: (1) identifying a key bottleneck of observation-predictive RL (see Section 4); (2) proposing a novel observation normalization method for the bottleneck (see Section 5.1); (3) introducing a unified framework applicable to diverse domains for observation-predictive RL (see Section 5.2).

   Guidelines:

   - The answer [N/A] 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 [N/A] 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: The paper discusses limitations in Section 7.

   Guidelines:

   - The answer [N/A] 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: [N/A]

Justification: This paper does not include theoretical results.

Guidelines:

- The answer [N/A] 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: The experimental details for reproducibility can be found in Appendix B, including benchmarks, baselines, evaluation protocol, network architectures, implementation details, hyperparameters, and other necessary information.

Guidelines:

- The answer [N/A] 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: The benchmarks used in this paper are publicly available. The code, together with sufficient instructions, is uploaded as a zip file and will be publicly released upon acceptance.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: See Appendix B for experimental details.

   Guidelines:

   - The answer [N/A] 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: For our own runs, we report the mean with 95% bootstrap confidence intervals over 5 runs for final performance metrics and learning curves of each method. For results taken from prior papers, statistics are reported when available from the original sources.

   Guidelines:

   - The answer [N/A] 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: All experiments were conducted on a machine equipped with an NVIDIA RTX 4090 GPU. See Figure 3 for training wall-time across methods.

   Guidelines:

   - The answer [N/A] 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 have carefully reviewed the Code of Ethics and affirm that our work fully complies with its principles.

   Guidelines:

   - The answer [N/A] 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 and negative societal impacts in Appendix F.

    Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This question is not applicable because the paper does not release high-risk data or models.

    Guidelines:

    - The answer [N/A] 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: [Yes]

    Justification: We have faithfully cited the corresponding paper and open-sourced code according to their licenses.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.