Self-Supervised Representations for Multi-View Reinforcement Learning

Abstract

Learning from raw, pixel images is quite important for the real-world application of deep reinforcement learning (RL). Standard model-free RL algorithms aim to learn policies directly from high-dimensional observations and simultaneously solve the representation learning and policy learning problems in an end-to-end training process. However, learning directly from images is sample-inefficiency and sensitive to hyper-parameters in practice under the single supervision of model-free reward signals. In this work, we present **S**elf-**S**upervision **R**epresentations (S2R) for multi-view reinforcement learning, a sample-efficient representation learning method for learning policies from high-dimensional images when combined with the RL algorithm. S2R introduces a representation learning framework and defines an auxiliary objective based on the multi-view image observations and the Conditional Entropy Bottleneck (CEB) principle to learn efficient representations which can discard task-irrelevant information and preserve task-relevant information. We demonstrate the effectiveness of S2R in the visual DeepMind Control (DMControl) Suite and show its better performance on the default DMControl tasks and their variants by replacing the task's background with a random image or natural video.

1 INTRODUCTION

In recent years, deep reinforcement learning (RL) has shown the potential to learn high-quality policies directly from complex environments with high-dimensional observations, such as playing Atari video games (Mnih et al., 2015; Hessel et al., 2018) or operating in visual continuous control tasks (Lillicrap et al., 2016), etc. Note that we can decouple the RL learning process into two sub-processes: representation learning and policy learning. The former aims to abstract features that characterize high-dimensional observations, and the latter aims to find optimal policies that maximize expected cumulative returns. However, standard RL algorithms unify the above two sub-processes into an end-to-end training procedure, making the learning sample-inefficiency (Lake et al., 2017; Kaiser et al., 2019) when just being supervised by the model-free reward signals. This situation will be aggravated in the real world as collecting interacting data and training specific policies is expensive and time-consuming (Kalashnikov et al., 2018; Akkaya et al., 2019).

Therefore, for RL algorithms, decoupling representation learning and policy learning in one training procedure provides a feasible solution to alleviate the problem of sample inefficiency. Representation learning decomposes high-dimensional data into low-vectored representations that can faithfully characterize them (Lesort et al., 2018). Then, policy learning can benefit from these low-dimensional and informative representations, rather than the raw data, to make the task sample-efficiently solved. In this paper, we base our method on this idea, first relying on an auxiliary objective to explicitly obtain latent representations, then training the agent upon these latent representations.

We focus on multi-view RL, which extends RL to multi-view settings. While most RL algorithms solely consider one-view observation from the latent state space, multi-view settings release the restrictions that hinder the application of RL to real-life scenarios. Take the smart vehicle as an example, instead of only using one-view data, it fuses multi-view data perceived by the multiple sensors to make safe driving decisions. Actually, compared with the paradigm of learning in one-view settings, learning in multi-view settings is more complex due to the increase of sample complexities and difficulties of reasoning joint representations from different views. If solved, it can promote the generalization of multi-view RL across varying domains, including their applications in the real world. Thus, we propose S2R: **S**elf-**S**upervision **R**epresentations for multi-view reinforcement

Submitted to the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022). **To be used for reviewing only**.

learning. Our key contributions are summarized as follows.

- **Representation learning framework.** To support the representation learning in multi-view RL, we design a specific learning framework. It consists of four components: the encoder network, feature fusion module, view-specific predictor, and the multi-view predictor. The feature fusion module inputs the multi-view embeddings encoded by the encoder network to get joint latent representations further utilized by the multi-view predictor. Meanwhile, the view-specific predictor makes predictions via view-specific embedding.
- **Self-supervision objective.** To learn compressed representations, inspired by the Conditional Entropy Bottleneck (CEB) (Fischer, 2020), we define a new multi-view CEB (MCEB) auxiliary objective that maximizes the mutual information (i.e., task-relevant information) between representations (multi-view or view-specific) and self-supervision signals (transition dynamics and reward function), and compresses away any information (i.e., task-irrelevant information) that comes from multi-view observations but is not contained in the supervision signals.
- **Representation learning for RL.** To integrate the representation learning with the RL training, we incorporate the MCEB objective with the RL objective by optimizing the RL objective on top of the encoder network optimized by the MCEB objective. Empirically, we show that S2R performs better in terms of sample efficiency and asymptotic performance in visual continuous control tasks and their noisy variants by replacing the task's default background with a random image or complex natural video in the DeepMind Control (DMControl) Suite (Tassa et al., 2018).

2 RELATED WORK

Reconstruction-based representations. Auto-encoder, an unsupervised learning technique that uses neural networks for representation learning, is the early work that combines with RL in control tasks (Lange and Riedmiller, 2010; Lange et al., 2012; Yarats et al., 2021). These RL agents first trained an encoder via the reconstruction loss, then learned policies based on the representations encoded by the encoder. However, there is no guarantee that the encoder captures useful information for the control tasks in practice. Aiming at this problem, researchers proposed to train the encoder jointly with the RL dynamics to learn task-oriented and predictive representations (Watter et al., 2015; Wahlström et al., 2015; Hafner et al., 2019, 2020, 2021; Lee et al., 2020a). Although effective, these approaches try to encode all details into embeddings in the reconstruction process of visual images, resulting in their sensibility to visual changes and negative effection on performance due to the existence of task-irrelevant information (Zhang et al., 2018).

Contrastive-based representations. As a representation learning method, contrastive learning has been widely used in self-supervised settings and made significant progress in the research of image classification and detection (Caron et al., 2020; Xie et al., 2021). It uses data augmentation (Chen et al., 2020) or image patches (Henaff, 2020) to acquire data samples and learns rich representations via similarity functions (Belghazi et al., 2018; Poole et al., 2019) such that the distance between similar pairs is minimized, between dissimilar pairs is maximized. Many works (Kim et al., 2019; Srinivas et al., 2020; Mazoure et al., 2020) have introduced contrastive learning to RL settings to extract predictive features. However, under the effect of contrastive loss, these methods aim to capture all features in the images to maximize the lower bound of the mutual information, making the features containing task-irrelevant information.

Multi-view and other representations. To solely extract task-relevant features from high-dimensional data, researchers have tried various methods. Multi-view learning, also known as data fusion or data integration from multiple views data, is an emerging area in machine learning (Zhang et al., 2016). Though abundant in computer vision tasks (Federici et al., 2020; Wang et al., 2019; Wan et al., 2021), it gains less attention on RL decision-making tasks. Chen et al. (2017) proposed the double-task deep Q-Network with multiple views based on double-DQN and dueling-DQN. Li et al. (2019) defined a framework that generalized partially observable Markov decision processes (POMDPs) to multi-view settings. In addition, Zhang et al. (2021) introduced the bisimulation metric (Ferns et al., 2011) to learn latent representations that only encode task-relevant information of image observations. Laskin et al. (2020) proposed a plug-and-play module by incorporating data augmentations with the RL agent and achieved SOTA performance on default visual DMControl tasks. Lee et al. (2020b) learned compressed representations of the predictive information of RL dynamics through a CEB objective (principle) with the CatGen decoder (Fischer, 2020) in the single-view settings. By contrast, our work, S2R, which learns robust representations via an MCEB auxiliary objective based on the multi-view POMDPs, simultaneously takes advantage of multi-view learning and CEB principle to preserve task-relevant information and ignore task-irrelevant information. We empirically show the performance improvement of S2R against state-of-the-art methods on a variety of control benchmarks.

3 PRELIMINARIES

Multi-view Reinforcement Learning. In this paper, we consider the multi-view reinforcement learning, an extension of RL to multi-view settings, formulated as a multi-view partially observable Markov decision process (Multi-view POMDP) $\{S, A, P, O_1, \mathbb{O}_1, \cdots, O_N, \mathbb{O}_N, R, \gamma \in (0,1)\}$. Here, symbols S, A, $P(s^{t+1}|s^t, a^t) : S \times A \times S \mapsto [0,1]$,

$R(s^t, a^t) : S \times A \mapsto \mathbb{R}$ and γ have the same meaning as in POMDP and respectively denote the state space, action space, transition probability of state s^{t+1} when agent takes action a^t at state s^t, reward function that maps state s^t and action a^t into real number, and the discount factor. Besides, O_j and $\mathbb{O}_j(o^t|s^t) : S \times O \mapsto [0, 1]$ denote the observation space, and observation model that generates high-dimensional observations of each view $j \in [1, N]$.

In standard RL, agent aims to learn a policy that maximizes the expected return. To be consistent with this goal, in multi-view RL, we introduce a multi-view trajectory $\tau_{M-view} = [s^1, o_j^1, a^1, \cdots, s^T, o_j^T, a^T]$, defined as follows.

$$\begin{aligned}\tau_{M-view} = \mathbb{O}_j(o_j^T|s^T)\pi(a^T|s^T)P(s^T|s^{T-1}, a^{T-1}) \\ \cdots \mathbb{O}_j(o_j^1|s^1)\pi(a^1|s^1)P_0(s^1)\end{aligned} \tag{1}$$

Where j is the view index, $P_0(s^1)$ is the initial state distribution, T is the time horizon. Along with this trajectory, agent learns a policy to maximize the expected cumulative discounted reward $R(\tau_{M-view}) = \sum_t \gamma^t R(s^t, a^t)$.

Soft Actor-Critic. Soft Actor-Critic (SAC) (Haarnoja et al., 2018) is an off-policy actor-critic algorithm that learns a stochastic policy π_ϕ to maximize a maximum entropy return $\mathbb{E}_{s^t, a^t \sim \pi}[\sum_t r^t + \alpha \mathcal{H}(\pi(\cdot|o^t))]$ by optimizing three objectives. Given transition tuples $\tau^t = (o^t, a^t, r^t, o^{t+1})$ sampled from the replay buffer $\mathcal{B}$, the critic minimizes the below Bellman error.

$$L_{Q_{\varphi_i}} = \mathbb{E}_{\tau \sim \mathcal{B}}\left[\left(Q_{\varphi_i}(o^t, a^t) - (r^t + \gamma V(o^{t+1}))\right)^2\right] \tag{2}$$

Where $V(o^{t+1})$ is the target value of o^{t+1}, defined as:

$$V(o^{t+1}) = \mathbb{E}_{a' \sim \pi}\Big(\min_{i=1,2} \bar{Q}_{\bar{\varphi}_i}(o^{t+1}, a') - \alpha \log \pi_\phi(a'|o^{t+1})\Big) \tag{3}$$

Note that SAC maintains two critics (Q_{φ_1}, Q_{φ_2}), two target critics ($\bar{Q}_{\bar{\varphi}_1}$, $\bar{Q}_{\bar{\varphi}_2}$) and uses the exponential moving average (EMA) to update target network parameters. For the actor, actions are sampled using the reparameterization trick, i.e., $a_\phi(o^t, \xi) = \tan(\mu_\phi(o^t) + \sigma_\phi(o^t) \odot \xi)$ with a standard normalized noise vector $\xi \sim \mathcal{N}(0, I)$, it minimizes:

$$L_\phi = \mathbb{E}_{a \sim \pi}\Big[\alpha \log \pi_\phi(a|o^t) - \min_{i=1,2} Q_{\varphi_i}(o^t, a)\Big] \tag{4}$$

For the temperature, given the target entropy $\mathcal{H}$ of the policy distribution, it minimizes:

$$L_\alpha = \mathbb{E}_{a \sim \pi}[-\alpha \log \pi(a|o^t) - \alpha \mathcal{H}] \tag{5}$$

4 S2R FOR MULTI-VIEW RL

To address the learning challenges of multi-view RL mentioned in Sec. 1, we propose S2R, which mainly contains: the representation learning framework, the self-supervision objective, and the combination of S2R with RL. For readability, we simplify the time index of the transition tuple, replacing $\{o^t, a^t, r^t, o^{t+1}\}$ with $\{o, a, r, o'\}$.

4.1 S2R FRAMEWORK

To extract representations from pixel observations in multi-view RL, in Fig. 1, we design an S2R representation learning framework. It includes:

(1) Encoder network. It is responsible for encoding high-dimensional observations into low-dimensional embeddings (representations) in a common latent space.

(2) Feature fusion module. Its purpose is to integrate multi-view marginal embeddings into joint latent representations in the common latent space.

(3) View-specific predictor. By inputting the marginal embedding together with the action and predicting the transition dynamics and reward function, it can maximize task-relevant information and minimize task-irrelevant information in the marginal representation.

(4) Multi-view predictor. By inputting the joint latent representation together with the action and doing the same prediction, it can effectively extract useful information from the joint representation.

4.2 S2R OBJECTIVE

Two-view CEB. In 2020, CEB (Fischer, 2020) was proposed. Given the high-dimensional data X, it learns representation Z from X to predict label Y, defined as $\min_Z \beta I(X; Z|Y) - I(Y; Z)$, expecting that the information captured in Z is maximally relevant to Y. In CEB, $I(X; Z|Y)$ is the conditional mutual information, measuring the reduction of uncertainty of X due to learning Z when giving Y; $I(Y; Z)$ is the mutual information, measuring the reduction of uncertainty of Y due to learning Z (Cover, 1999). Based on CEB, we propose a new MCEB objective to optimize networks related to the S2R framework (Sec. 4.1). For simplicity, we start with a two-view case. Considering the sequential nature of RL, we define X_1, X_2 as the current observations, Z_1, Z_2, Z as the current latent representations, and Y_1, Y_2, Y as the next latent representations and rewards. Without loss of generality, we define the two-view CEB objective as:

$$\begin{aligned}\textbf{obj.} \min_{Z, Z_1, Z_2} \beta_1 I(X_1; Z_1|Y_1) + \beta_2 I(X_2; Z_2|Y_2) - I(Z; Y) \\ = \min_{s, s_1, s_2} \beta_1 I(o_1; s_1|s_1', r, a) + \beta_2 I(o_2; s_2|s_2', r, a) - \\ I(s; s', r|a) \ \textbf{s.t.}\ Z = f_\theta(Z_1, Z_2) \Rightarrow s = f_\theta(s_1, s_2)\end{aligned} \tag{6}$$

Where β_1, β_2 are regularization factors. To better understand this objective, we show an Information diagram (I-diagram) for X_1, X_2, Z_1, Z_2, Z, Y_1, Y_2 and Y in Fig. 2. Intuitively, we observe that: $I(X_1; Z_1) = I(Z_1; Y_1) + I(X_1; Z_1|Y_1)$, $I(X_2; Z_2) = I(Z_2; Y_2) + I(X_2; Z_2|Y_2)$. Thus, to get a minimal and sufficient Z, we must minimize redundant information ($I(X_1; Z_1|Y_1)$ and $I(X_2; Z_2|Y_2)$) and maximally preserve relevant information ($I(Z; Y)$, where $Z = f_\theta(Z_1, Z_2)$

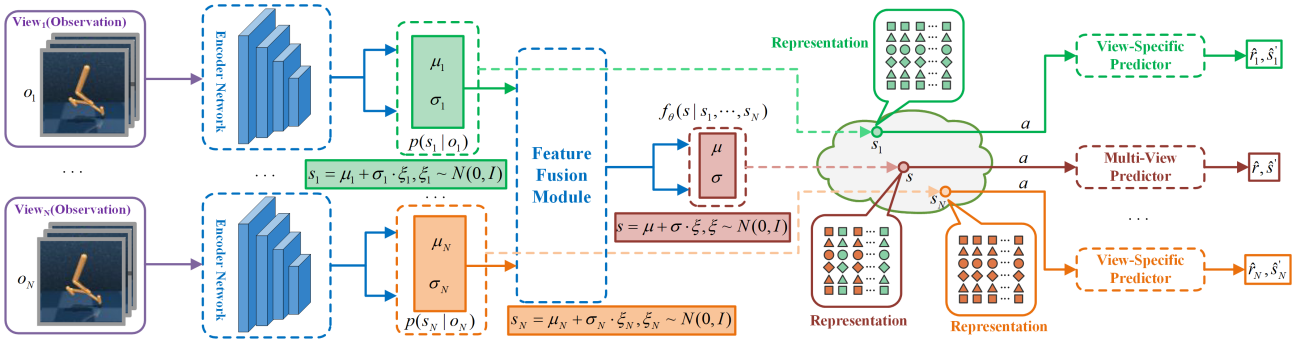

Figure 1: S2R framework. It contains the encoder network, feature fusion module, view-specific predictor, and multi-view predictor. Multi-view observations are fed into the encoder network to learn multi-view embeddings. These embeddings are successively fed into the feature fusion module and the multi-view predictor to predict the latent transition dynamics and reward function. Besides, they are also inputted into the view-specific predictor to get the corresponding predictions.

is the joint representation of marginal representations Z_1 and Z_2 fused by the S2R feature fusion module).

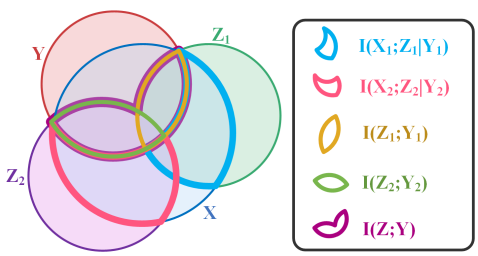

Figure 2: I-diagram of the two-view CEB.

Optimization of Two-view CEB. In Eq. (6), it is intractable to directly compute the (conditional) mutual information terms. Fortunately, the variational method provides a feasible solution by approximating intractable terms with variational bounds (Kingma and Welling, 2014). To get the variational upper bound of Eq. (6), we first rewrite it below.

$$\begin{aligned}&\min_{Z,Z_1,Z_2} \beta_1(I(X_1;Z_1)-I(Z_1;Y_1))+\beta_2(I(X_2;Z_2)- \\ &\qquad I(Z_2;Y_2))-I(Z;Y),\ \ Z=f_\theta(Z_1,Z_2) \\ &=\min_{s,s_1,s_2} \beta_1(I(o_1;s_1)-I(s_1;s_1'|a))+\beta_2(I(o_2;s_2)- \\ &\qquad I(s_2;s_2',r|a))-I(s;s',r|a),\ s=f_\theta(s_1,s_2) \end{aligned} \tag{7}$$

Then, we give the joint probability density function of variables o_1, o_2, s_1, s_2, s, s_1', s_2', s', r and a. According to the Bayes's rule, it can be expressed as:

$$\begin{aligned}&p(o_1,o_2,s_1,s_2,s,s_1',s_2',s',r,a)=p(s|o_1,o_2,s_1,s_2, \\ &s_1',s_2',s',r,a)\cdot p(s_1|o_1,o_2,s_2,s_1',s_2',s',r,a)\cdot \\ &p(s_2|o_1,o_2,s_1',s_2',s',r,a)\cdot p(o_1,o_2,s_1',s_2',s',r,a)\end{aligned} \tag{8}$$

Considering s_1 is extracted from o_1, s_2 is extracted from o_2, s is fused by s_1 and s_2, we thus infer that: s_1 is independent of variables other than o_1, s_2 is independent of variables other than o_2, and s is independent of variables other than s_1 and s_2. Therefore, we have:

$$\begin{aligned}&p(o_1,o_2,s_1,s_2,s,s_1',s_2',s',r,a)=p(s|s_1,s_2)\cdot \\ &p(s_1|o_1)\cdot p(s_2|o_2)\cdot p(o_1,o_2,s_1',s_2',s',r,a)\end{aligned} \tag{9}$$

Based on the standard definition of the mutual information, the non-negative property of the Kullback-Leibler divergence (KL-divergence), the above joint probability density function, and the Monte Carlo sampling (Shapiro, 2003), we derive the variational upper bound of Eq.(6) as follows.

$$\begin{aligned}&\beta_1 I(o_1;s_1|s_1',r,a)+\beta_2 I(o_2;s_2|s_2',r,a)-I(s;s',r|a)\leq \\ &\frac{1}{M}\sum^{M}\Big(\beta_1\int_{s_1}p(s_1|o_1)\log\frac{p(s_1|o_1)}{q_1(s_1)}d_{s_1}+\beta_2\int_{s_2}p(s_2|o_2) \\ &\log\frac{p(s_2|o_2)}{q_2(s_2)}d_{s_2}-\beta_1\int_{s_1}p(s_1|o_1)\log g_1(s_1',r|s_1,a)d_{s_1}- \\ &\beta_2\int_{s_2}p(s_2|o_2)\log g_2(s_2',r|s_2,a)d_{s_2}-\int_{s_1}\int_{s_2}\int_{s}p(s_1|o_1) \\ &p(s_2|o_2)p(s|s_1,s_2)\log g(s',r|s,a)d_{s_1}d_{s_2}d_s\Big)\end{aligned} \tag{10}$$

Where M is the size of data sampled by the Monte Carlo sampling, $g_1(s_1',r|s_1,a)$, $g_2(s_2',r|s_2,a)$ and $g(s',r|s,a)$ are distributions learned from neural networks to approximate real distributions $p(s_1',r|s_1,a)$, $p(s_2',r|s_2,a)$ and $p(s',r|s,a)$, variational distributions $q_1(s_1)\sim N(0,I), q_2(s_2)\sim N(0,I)$ are used to approximate real distributions $p(s_1)$ and $p(s_2)$. Detailed derivations of Eq. (10) are given in Appendix A.

Next, we assume $p(s_1|o_1)$, $p(s_2|o_2)$ and $p(s|s_1,s_2)$ are Gaussian distributions, where the relative means (μ_1,μ_2,μ) and variances $(\sigma_1,\sigma_2,\sigma)$ are learned from neural networks:

$$\begin{aligned}&p(s_1|o_1)=\mathcal{N}(\mu_1(o_1;\psi_1),\sigma_1(o_1;\psi_1)) \\ &p(s_2|o_2)=\mathcal{N}(\mu_2(o_2;\psi_2),\sigma_2(o_2;\psi_2)) \\ &p(s|s_1,s_2)=\mathcal{N}(\mu(s_1,s_2;\psi),\sigma(s_1,s_2;\psi))\end{aligned} \tag{11}$$

In Eq. (11), ψ_1, ψ_2 and ψ are parameters of the neural networks used for learning $p(s_1|o_1)$, $p(s_2|o_2)$ and $p(s|s_1,s_2)$,

respectively. To backpropagate the gradient through random variables s_1, s_2 and s, we use the reparameterization trick to transform them into deterministic functions:

$$\begin{aligned} s_1 &= \mu_1(o_1;\psi_1) + \sigma_1(o_1;\psi_1)\cdot\xi_1 \\ s_2 &= \mu_2(o_2;\psi_2) + \sigma_2(o_2;\psi_2)\cdot\xi_2 \\ s &= \mu(s_1,s_2;\psi) + \sigma(s_1,s_2;\psi)\cdot\xi \end{aligned} \tag{12}$$

Where $\xi_1, \xi_2, \xi \in \mathcal{N}(0, I)$. Therefore, Eq. (10) will be transformed into Eq. (13), the final optimization loss of Eq. (6).

$$\begin{aligned} \min_{s,s_1,s_2} \frac{1}{M}\sum^{M} \Big(&\beta_1 \mathbb{E}_{\xi_1}\big(\log\frac{p(s_1|o_1)}{q_1(s_1)} - \log g_1(s_1', r|s_1,a)\big) \\ &+ \beta_2 \mathbb{E}_{\xi_2}\big(\log\frac{p(s_2|o_2)}{q_2(s_2)} - \log g_2(s_2', r|s_2,a)\big) \\ &- \mathbb{E}_{\xi_1}\mathbb{E}_{\xi_2}\mathbb{E}_{\xi}\log g(s',r|s,a)\Big) \end{aligned} \tag{13}$$

From Two-view CEB to MCEB. For the cases with more than two views, we can easily generalize the two-view CEB objective to the MCEB objective by adding corresponding terms. Given N views $(X_1, \ldots, X_N)$, it is expressed as:

$$\begin{aligned} &\textbf{obj.} \quad \min_{Z,Z_1,\cdots,Z_N} \textstyle\sum_{j=1}^{N} \beta_j I(X_j;Z_j|Y_j) - I(Z;Y) = \\ &\min_{s,s_1,\cdots,s_N} \sum_{j=1}^{N} \big[\beta_j(I(o_j;s_j) - I(s_j;s_j',r|a))\big] - I(s;s',r|a) \\ &\textbf{s.t.} \quad Z = f_\theta(Z_1,\cdots,Z_N) \Rightarrow s = f_\theta(s_1,\cdots,s_N) \end{aligned} \tag{14}$$

Referring to the same derivation process of the two-view CEB objective, the final optimization loss of the MCEB objective (Eq. (14)) can be expressed as follows:

$$\begin{aligned} \min_{s,s_1,\cdots,s_N} \frac{1}{M}\sum^{M} \Big(&\sum_{j=1}^{N}\big[\beta_j \mathbb{E}_{\xi_j}\big(\log\frac{p(s_j|o_j)}{q_j(s_j)} - \log g_j(s_j', \\ &r|s_j,a)\big)\big] - \mathbb{E}_{\xi_1}\ldots\mathbb{E}_{\xi_N}\mathbb{E}_{\xi}\log g(s',r|s,a)\Big) \end{aligned} \tag{15}$$

4.3 INCORPORATE S2R INTO RL

To incorporate the S2R objective into RL, we simultaneously train the S2R model (encoder $p(o)$: $s \sim p(s|o)$ and other networks) and the RL agent and treat the S2R optimization loss as an auxiliary loss (Fig. 3).

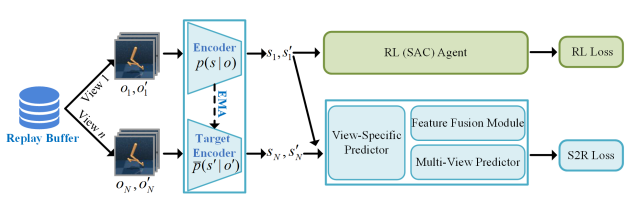

Figure 3: Joint training of S2R model and the RL agent.

To obtain the multi-view observations, we adopt the data augmentation to expand the data sampled from the replay buffer. Exactly, we randomly apply the augmentation across the sampled batches but keep it consistent across the consecutive frame stacks to retain the temporal information hidden in the observations. This practice allows the S2R model to infer task dynamics and is thus more suitable for the RL setting. In Algorithm 1, we give the detailed procedure of integrating S2R with SAC. In our actual implementation, we use two views' data. Observations from the first view are not only responsible for training the RL agent but also responsible for training the S2R model together with the observations from the second view.

Algorithm 1 S2R + SAC pseudo-code

Initial: parameters of the critic ($\varphi_i, i \in \{1,2\}$), actor(ϕ), S2R model ($\theta, \psi, \psi_j, j \in [1, N]$), replay buffer $\mathcal{B} \leftarrow \emptyset$, training step T, gradient step K, batch size M
for step $t = 1$ to T **do**
  for each collection step **do**
    Store interaction data: $\mathcal{B} \leftarrow \mathcal{B} \cup (o, a, r, o')$.
  end for
  for step $k = 1$ to K **do**
    Sample M batches $D = (o_j, a, r, o_j')$ from $\mathcal{B}$.
    Compute target value:
      $V = \min \bar{Q}_i(\bar{p}(o_1'), a') - \alpha \log \pi(a'|p(o_1'))$
    Update critic:
      $L_{\varphi_i} = [Q_i(p(o_1), a) - (r + \gamma V)]^2$
    Update actor:
      $L_\phi = \alpha \log \pi(a|p(o_1)) - \min Q_i(p(o_1), a)$
    Update temperature:
      $L_\alpha = -\alpha \log \pi(a|p(o_1)) - \alpha\mathcal{H}$
    Update S2R model ($p(o)$, etc.) by D, Eq. (13).
    Update target critic: $\bar{Q}_i = \tau_Q \cdot Q_i + (1-\tau_Q)\cdot\bar{Q}_i$
    Update target S2R encoder: $\bar{p} = \tau_p p + (1-\tau_p)\bar{p}$
  end for
end for

5 EXPERIMENTS

In this paper, we design a variety of experiments to answer the following questions:

- Can S2R have a better sample efficiency in RL visual control tasks (Table 1, Fig. 5 - 7)?
- Is S2R robust to complex settings with the random image distractor or natural video distractor (Fig. 7)?
- Can S2R perform better than existing reconstruction-based, non-reconstruction-based, or contrastive-based RL representation methods (Table 1, Fig. 5 - 7)?
- For S2R, How much information should be preserved for efficient representation? Is it sufficient to merely predict the reward function or transition dynamics in MCEB? Is the MCEB objective more suitable than its mutual information or CEB variants? (Fig. 7)

5.1 EXPERIMENT SETUP

DMControl Suite. To evaluate the performance of S2R, we combine it with the SAC algorithm and focus on the visual continuous control tasks in the DMControl Suite (Tassa et al., 2018). Our benchmark includes six different environments under three settings (default setting, image distractor setting, and natural video setting). **(1) Default Setting.** Agent receives pixel observations with default background. **(2) Image Distractor Setting.** Agent receives pixel observations with the random image as the background. **(3) Natural Video Setting.** Agent receives pixel observations with the natural video selected from the "arranging flowers" class of the Kinetics dataset (Kay et al., 2017) as the background. In Fig. 4, We show snapshots of pixel observations of all related control tasks in the above three settings.

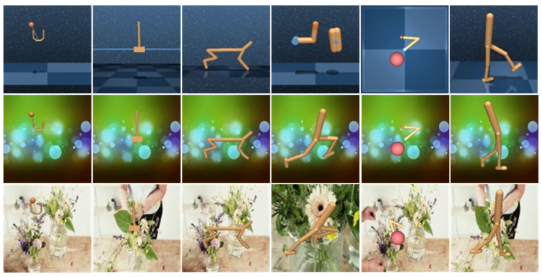

Figure 4: Top images are observations with the default background. Middle images are observations with a random image distractor as the background. The last images are observations with a natural video as the background. Tasks from left to right are ball-in-cup catch, cartpole swingup, cheetah run, finger spin (the first row) or walker run (the second and the third row), reacher easy, and walker walk.

Implementation. We base our S2R method on the implementation of RAD (Laskin et al., 2020) [1]. We use a desktop with an 8-core CPU, two Nvidia GeForce RTX 3090 for each benchmarking, and figures are averaged across three seeds unless specified otherwise. Throughout all experiments, we use almost all of the default parameters of the standard SAC algorithm, including the discount factor, updating frequency of target networks, etc. Full implementation details and hyper-parameters are listed in Appendix B.

5.2 BASELINE ALGORITHM

In this paper, we compare S2R + SAC with some state-of-the-art pixel-based RL methods. DBC (Zhang et al., 2021) learns effective representations for downstream control tasks through the bisimulation metric. RAD (Laskin et al., 2020) uses augmented data to train policy. CURL (Srinivas et al., 2020) combines contrastive learning objective with model-free RL agent. SLAC (Lee et al., 2020a) learns stochastic sequential models via a variational inference objective. PlaNet (Hafner et al., 2019) and Dreamer (Hafner et al., 2020) are two model-based algorithms, they both learn a world model and respectively choose actions via online planning and long-horizon imagination. SAC + AE (Yarats et al., 2021) combines auto-encoder with model-free RL algorithm via an auxiliary reconstruction loss. Pixel SAC is the SAC (Haarnoja et al., 2018) algorithm with image inputs, while State SAC operates on proprioceptive states (positions, velocities, etc.). All the above methods are built on top of the off-policy SAC algorithm. Besides, in DBC, we use the same action repeat as RAD to make a fair comparison.

[1] https://github.com/MishaLaskin/rad

5.3 MAIN RESULTS

Default Setting Results. To evaluate the sample efficiency of our method, we first give the median scores achieved by S2R + SAC along with the baselines at DMControl100k (low sample performance) and DMControl500k (optimal asymptotically performance) benchmarks [2] in Fig. 5 and show their relative scores on 6 control tasks in Table 1 and Fig. 6. In Fig. 5, S2R + SAC achieves 1.14x/1.04x higher median scores than State SAC, 1.59x/1.05x higher median scores than CURL, and 6.69x/5.12x higher median scores than Pixel SAC at 100k/500k environment steps, showing that S2R + SAC has a higher sample efficiency. In Table 1, S2R + SAC, which integrates multi-view CEB representation learning with model-free RL learning, is the state-of-the-art algorithm on all (6 out of 6) visual DMControl tasks on both DMControl100k and DMControl500k benchmarks. It achieves impressive results, exceeds the performance of best-performing RAD and CURL, matches the performance of State SAC operating from proprioceptive states, and significantly improves the performance of Pixel SAC on both DMControl100k and DMControl500k benchmarks. In Fig. 6, the learning curves of S2R + SAC and DBC again confirm the better sample efficiency of S2R + SAC.

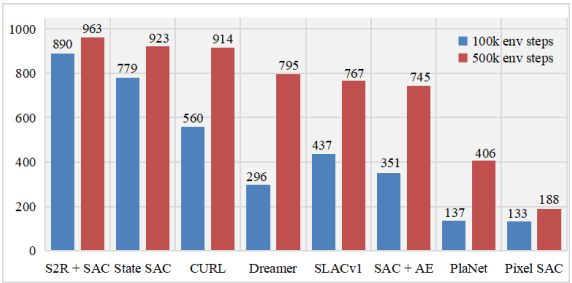

Figure 5: Performance of S2R + SAC relative to State SAC, CURL, Dreamer, SLACv1, SAC + AE, PlaNet, and Pixel SAC averaged across 10 seeds in the default setting. Results are the medians of 6 pixel-based control tasks in Table 1, and data other than S2R + SAC is reported in CURL.

[2] DMControl100k/DMControl500k refers to 100k/500k environment or simulator steps, which is equal to 50k/250k policy steps if the action repeat is set to 2.

Table 1: We report scores (mean and standard deviation) for S2R + SAC and baselines (report by RAD) on DMControl500k and DMControl100k. Results are statistics by averaging the scores of 10 seeds for the 6 control tasks. In both benchmarks, compared with existing baselines, S2R + SAC achieves state-of-the-art performance on all (6 out of 6) control tasks.

500K STEP SCORES	S2R + SAC	RAD	CURL	PlaNet	Dreamer	SAC + AE	SLACv1	PIXEL SAC	STATE SAC
FINGER, SPIN	**983** ± 5	947 ±101	926 ±45	561 ±284	796 ±183	884 ±128	673 ±92	192 ±166	923 ±211
CARTPOLE, SWING	**869** ± 10	863 ±9	845 ±45	475 ±71	762 ±27	735 ±63	-	419 ±40	848 ±15
REACHER, EASY	**981** ±5	955 ±71	929 ±44	210 ±44	793 ±164	627 ±58	-	145 ±30	923 ±24
CHEETAH, RUN	**811** ± 32	728 ±71	518 ±28	305 ±131	570 ±253	550 ±34	640 ±19	197 ±15	795 ±30
WALKER, WALK	**950** ±19	918 ±16	902 ±43	351 ±58	897 ±49	847 ±48	842 ±51	42 ±12	948 ±54
CUP, CATCH	**978** ±5	974 ±12	959 ±27	460 ±380	879 ±87	794 ±58	852 ±71	312 ±63	974 ±33
100K STEP SCORES	**S2R + SAC**	**RAD**	**CURL**	**PlaNet**	**Dreamer**	**SAC + AE**	**SLACv1**	**PIXEL SAC**	**STATE SAC**
FINGER, SPIN	**876** ±43	856 ±73	767 ±56	136 ±216	341 ±70	740 ±64	693 ±141	224 ±101	811 ±46
CARTPOLE, SWING	**868** ±9	828 ±27	582 ±146	297 ±39	326 ±27	311 ±11	-	200 ±72	835 ±22
REACHER, EASY	**961** ±40	826 ±219	538 ±233	20 ±50	314 ±155	274 ±14	-	136 ±15	746 ±25
CHEETAH, RUN	**603** ±26	447 ±88	299 ±48	138 ±88	235 ±137	267 ±24	319 ±56	130 ±12	616 ±18
WALKER, WALK	**897** ±42	504 ±191	403 ±24	224 ±48	277 ±12	394 ±22	361 ±73	127 ±24	891 ±82
CUP, CATCH	**968** ±6	840 ±179	769 ±43	0 ±0	246 ±174	391 ±82	512 ±110	97 ±27	746 ±91

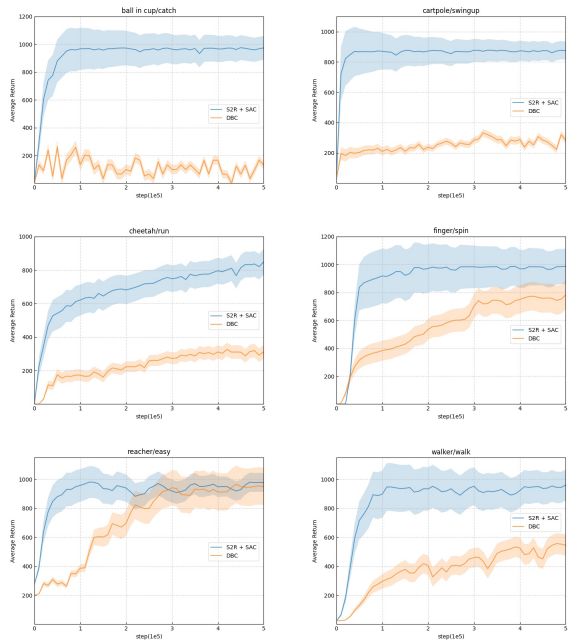

Figure 6: Learning curves of S2R + SAC in the default DMControl setting. We benchmark S2R + SAC with the bisimulation metric-based method (DBC), a supplement to the scores of Table 1. S2R + SAC outperforms DBC and achieves impressive performance on all 6 control tasks.

Image Distractor Setting Results. Then, we evaluate S2R performance in the image distractor setting by replacing the default background with a random image. In the top row of Fig. 7, we give the results of three tasks (ball-in-cup catch, cheetah run, and walker run) and additional results are shown in Appendix C. Results show that S2R + SAC performs comparable or substantially better than RAD, and consistently outperforms DBC, proving that S2R can discard task-irrelevant information in its learned representations.

Natural Video Setting Results. Next, we evaluate S2R + SAC, RAD, and DBC in a more complex distractor setting by introducing the natural video as the background. In the middle row of Fig. 7, we give the results of three tasks (ball-in-cup catch, cheetah run, and walker run) and additional results are shown in Appendix C. Based on the results, we conclude that comparing with RAD and DBC, S2R + SAC again attains better results and has a better sample efficiency. This attributes to our well designing of S2R, which makes the agent only focus on task-related features, insensitive to task-irrelevant visual changes, and thus providing good representations for the training of the actor and critic.

Ablation Studies. At last, we investigate how S2R is affected by the regularization factors in the MCEB objective, the data included in the MCEB objective (Y_1, Y_2 and Y), and different forms of the optimization objective, in the last row of Fig. 7 and additional results are shown in Ap-

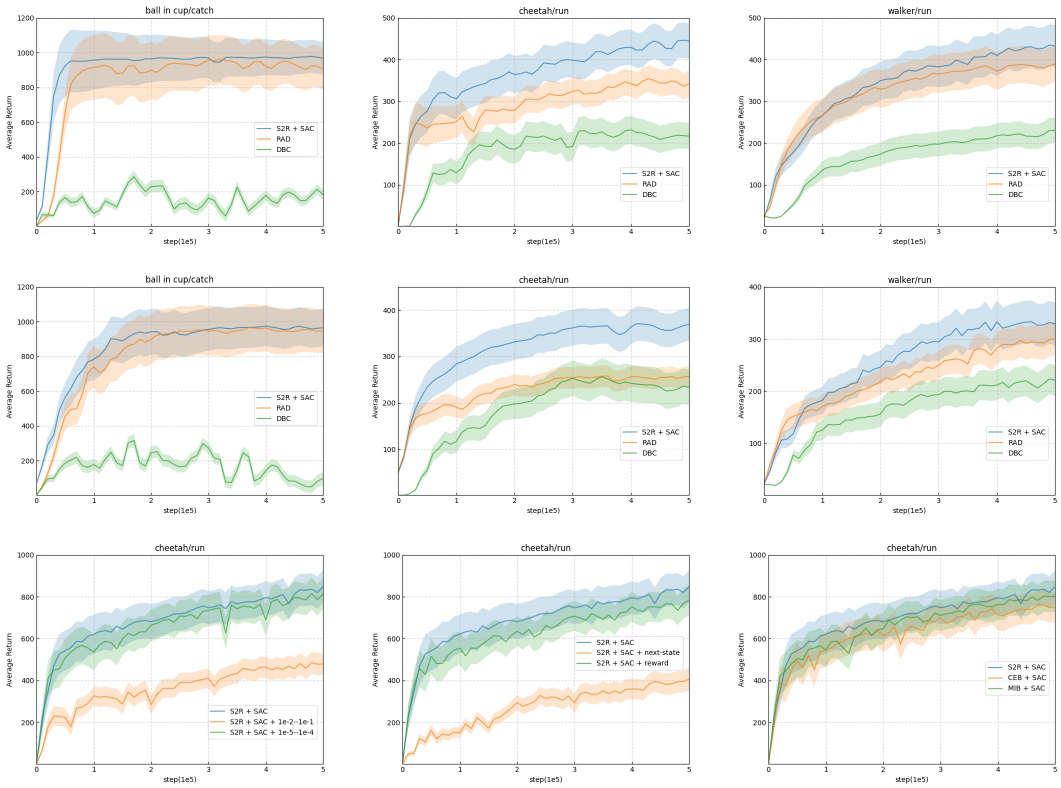

Figure 7: Performance of S2R + SAC in three background settings. **Top row**: Results in the image distractor setting. **Middle row**: Results in the natural video setting. We benchmark S2R + SAC with RAD and DBC in both settings, and results confirm the better performance of S2R + SAC. **Last row**: Results in the default setting for ablation studies. The first column compares regularization factors in the MCEB objective, the second column compares data included in Y_1, Y_2, and Y, and the third column compares different forms of the optimization objective. Additional results can be found in Appendix C.

pendix C. **(1) MCEB regularization factors.** In the MCEB objective, regularization factors are related to the trade-off between the sufficiency and robustness of the representation, and we use an exponential scheduler in all experiments. As seen from the first column, too-high values block information essential to the predictive data in the MCEB objective, while too-small values reduce the benefit of regularization. **(2) MCEB Predictive data.** To utilize the sequential nature of RL, the predictive data in the MCEB objective can contain the next latent state and reward or either of them. However, our experiment results in the second column show that simultaneously predicting latent transition dynamics and reward function is better than predicting either of them alone. **(3) Variants of the MCEB objective.** With a slight modification of the MCEB objective, its two variants that are similar to the reported works can be obtained. The first variant is equal to PI-SAC (Lee et al., 2020b), which optimizes the representation model by the CEB objective in the single-view RL setting. The second variant is equal to MIB (Multi-view Information Bottleneck) (Wang et al., 2019), which replaces the CEB term $I(X_j; Z_j|Y_j)$ with the IB term $I(X_j; Z_j)$ in MCEB. Compared to these two variants, our results in the third column show the better performance and higher sample efficiency of the MCEB objective.

6 CONCLUSIONS

In this paper, we present S2R, a multi-view-based self-supervision representation learning method to learn efficient and sufficient representations for the policy learning of the RL agent based on the conditional entropy bottleneck. S2R introduces a representation learning framework for multi-view RL and defines a novel MCEB auxiliary objective to extract useful features and ignore task-irrelevant information in high-dimensional pixel observations to train the actor and critic. We perform extensive experiments on the DM-Control suite. The experimental results show that S2R learns robust representations on various continuous control tasks and improves sample efficiency in the default and complex noisy settings. As a decoupling representation module, S2R is easy to integrate with both model-free and model-based RL algorithms. In the future, we aim to integrate S2R with other exploration mechanisms to solve the sparse-reward problems in complex visual RL settings.

References

Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International Conference on Machine Learning*, pages 531–540. PMLR, 2018.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems 33*, 2020.

Jun Chen, Tingzhu Bai, Xiangsheng Huang, Xian Guo, Jianing Yang, and Yuxing Yao. Double-task deep q-learning with multiple views. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 1050–1058, 2017.

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

Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

Marco Federici, Anjan Dutta, Patrick Forré, Nate Kushman, and Zeynep Akata. Learning robust representations via multi-view information bottleneck. In *8th International Conference on Learning Representations*, 2020.

Norm Ferns, Prakash Panangaden, and Doina Precup. Bisimulation metrics for continuous markov decision processes. *SIAM Journal on Computing*, 40(6):1662–1714, 2011.

Ian Fischer. The conditional entropy bottleneck. *Entropy*, 22(9):999, 2020.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870. PMLR, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, pages 2555–2565. PMLR, 2019.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *8th International Conference on Learning Representations*, 2020.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *9th International Conference on Learning Representations*, 2021.

Olivier Henaff. Data-efficient image recognition with contrastive predictive coding. In *International Conference on Machine Learning*, pages 4182–4192. PMLR, 2020.

Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, et al. Model-based reinforcement learning for atari. In *8th International Conference on Learning Representations*, 2019.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

Hyoungseok Kim, Jaekyeom Kim, Yeonwoo Jeong, Sergey Levine, and Hyun Oh Song. Emi: Exploration with mutual information. In *Proceedings of the 36th International Conference on Machine Learning*, pages 3360–3369. PMLR, 2019.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations*, 2014.

Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40, 2017.

Sascha Lange and Martin Riedmiller. Deep auto-encoder neural networks in reinforcement learning. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2010.

Sascha Lange, Martin Riedmiller, and Arne Voigtländer. Autonomous reinforcement learning on raw visual input data in a real world application. In *The 2012 international joint conference on neural networks (IJCNN)*, pages 1–8. IEEE, 2012.

Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. In *Advances in Neural Information Processing Systems 33*, 2020.

Alex X Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. In *Advances in Neural Information Processing Systems 33*, 2020a.

Kuang-Huei Lee, Ian Fischer, Anthony Liu, Yijie Guo, Honglak Lee, John Canny, and Sergio Guadarrama. Predictive information accelerates learning in rl. In *Advances in Neural Information Processing Systems 33*, 2020b.

Timothée Lesort, Natalia Díaz-Rodríguez, Jean-Franois Goudou, and David Filliat. State representation learning for control: An overview. *Neural Networks*, 108: 379–392, 2018.

Minne Li, Lisheng Wu, Haitham Bou Ammar, and Jun Wang. Multi-view reinforcement learning. In *Advances in Neural Information Processing Systems 32*, 2019.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations*, 2016.

Bogdan Mazoure, Remi Tachet des Combes, Thang Doan, Philip Bachman, and R Devon Hjelm. Deep reinforcement and infomax learning. In *Advances in Neural Information Processing Systems 33*, 2020.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In *International Conference on Machine Learning*, pages 5171–5180. PMLR, 2019.

Alexander Shapiro. Monte carlo sampling methods. *Handbooks in operations research and management science*, 10:353–425, 2003.

Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, pages 5639–5650. PMLR, 2020.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Niklas Wahlström, Thomas B Schön, and Marc Peter Deisenroth. From pixels to torques: Policy learning with deep dynamical models. *arXiv preprint arXiv:1502.02251*, 2015.

Zhibin Wan, Changqing Zhang, Pengfei Zhu, and Qinghua Hu. Multi-view information-bottleneck representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10085–10092, 2021.

Qi Wang, Claire Boudreau, Qixing Luo, Pang-Ning Tan, and Jiayu Zhou. Deep multi-view information bottleneck. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 37–45. SIAM, 2019.

Manuel Watter, Jost Tobias Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in Neural Information Processing Systems 28*, pages 2746–2754, 2015.

Zhenda Xie, Yutong Lin, Zheng Zhang, Yue Cao, Stephen Lin, and Han Hu. Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16684–16693, 2021.

Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*, pages 10674–10681, 2021.

Amy Zhang, Yuxin Wu, and Joelle Pineau. Natural environment benchmarks for reinforcement learning. *arXiv preprint arXiv:1811.06032*, 2018.

Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. In *9th International Conference on Learning Representations*, 2021.

Yanyan Zhang, Jianchun Zhang, Zhisong Pan, and Daoqiang Zhang. Multi-view dimensionality reduction via canonical random correlation analysis. *Frontiers of Computer Science*, 10(5):856–869, 2016.

A DERIVATION OF THE TWO-VIEW CEB LOSS

As mentioned in the main text, the two-view CEB objective is defined as:

$$\begin{aligned}\textbf{obj}: \quad & \min_{Z,Z_1,Z_2} \beta_1 I(X_1;Z_1|Y_1) + \beta_2 I(X_2;Z_2|Y_2) - I(Z;Y) \\ & = \min_{s,s_1,s_2} \beta_1 I(o_1;s_1|s_1',r,a) + \beta_2 I(o_2;s_2|s_2',r,a) - I(s;s',r|a) \\ \textbf{s.t.}: \quad & Z = f_\theta(Z_1,Z_2) \Rightarrow s = f_\theta(s_1,s_2)\end{aligned} \tag{16}$$

Considering $I(X_1;Z_1|Y_1) = I(X_1;Z_1) - I(Z_1;Y_1), I(X_2;Z_2|Y_2) = I(X_2;Z_2) - I(Z_2;Y_2)$, the original two-view CEB objective can be rewritten as:

$$\begin{aligned}\textbf{obj}: \quad & \min_{Z,Z_1,Z_2} \beta_1[I(X_1;Z_1) - I(Z_1;Y_1)] + \beta_2[I(X_2;Z_2) - I(Z_2;Y_2)] - I(Z;Y) \\ & = \min_{s,s_1,s_2} \beta_1(I(o_1;s_1) - I(s_1;s_1',r|a)) + \beta_2(I(o_2;s_2) - I(s_2;s_2',r|a)) - I(s;s',r|a) \\ \textbf{s.t.}: \quad & Z = f_\theta(Z_1,Z_2) \Rightarrow s = f_\theta(s_1,s_2)\end{aligned} \tag{17}$$

To begin with, we give the joint probability density function of variables o_1, o_2, s_1, s_2, s, s_1', s_2', s', r and a. Since s_1 is learned from o_1, s_2 is learned from o_2, s is fused by s_1 and s_2, thus, based on the Bayes' rule, this joint probability density function can be expressed as:

$$\begin{aligned}p(o_1,o_2,s_1,s_2,s,s_1',s_2',s',r,a) &= p(s|o_1,o_2,s_1,s_2,s_1',s_2',s',r,a) \cdot p(s_1|o_1,o_2,s_2,s_1',s_2',s',r,a) \cdot \\ &\quad p(s_2|o_1,o_2,s_1',s_2',s',r,a) \cdot p(o_1,o_2,s_1',s_2',s',r,a) \\ &= p(s|s_1,s_2) \cdot p(s_1|o_1) \cdot p(s_2|o_2) \cdot p(o_1,o_2,s_1',s_2',s',r,a)\end{aligned} \tag{18}$$

Then, we analysis the first term $I(o_1;s_1)$ in Eq. (17), according to the standard definition of mutual information, the mutual information between o_1 and s_1 is:

$$I(o_1;s_1) = \int_{o_1}\int_{s_1} p(o_1,s_1) \log \frac{p(s_1|o_1)}{p(s_1)} d_{o_1} d_{s_1} \tag{19}$$

Due to the intractable of $p(s_1)$, we use the variational distribution $q_1(s_1)$ to approximate it. Considering the non-negative property of the Kullback-Leibler divergence (KL-divergence), we can infer that:

$$KL(p(s_1)||q_1(s_1)) \ \geq \ 0 \Longrightarrow \int_{s_1} p(s_1) \log p(s_1) d_{s_1} \ \geq \ \int_{s_1} p(s_1) \log q_1(s_1) d_{s_1} \tag{20}$$

Substituting Eq. (20) into Eq. (19), we have:

$$\begin{aligned}I(o_1;s_1) &= \int_{o_1}\int_{s_1} p(o_1,s_1) \log \frac{p(s_1|o_1)}{p(s_1)} d_{o_1} d_{s_1} \ \leq \ \int_{o_1}\int_{s_1} p(o_1,s_1) \log \frac{p(s_1|o_1)}{q_1(s_1)} d_{o_1} d_{s_1} \\ &= \int_{o_1}\int_{o_2}\int_{s_1'}\int_{s_2'}\int_{s'}\int_{r}\int_{a}\int_{s_1} p(o_1,o_2,s_1',s_2',s',r,a,s_1) \log \frac{p(s_1|o_1)}{q_1(s_1)} d_{o_1} d_{o_2} d_{s_1'} d_{s_2'} d_{s'} d_r d_a d_{s_1}\end{aligned} \tag{21}$$

Considering variable s_1 depends only on variable o_1, we get the following variational bound for term $I(o_1;s_1)$:

$$I(o_1;s_1) \ \leq \ \int_{o_1}\int_{o_2}\int_{s_1'}\int_{s_2'}\int_{s'}\int_{r}\int_{a} p(o_1,o_2,s_1',s_2',s',r,a) d_{o_1} d_{o_2} d_{s_1'} d_{s_2'} d_{s'} d_r d_a \int_{s_1} p(s_1|o_1) \log \frac{p(s_1|o_1)}{q_1(s_1)} d_{s_1} \tag{22}$$

Similarly, for the third term $I(o_2;s_2)$ in Eq. (17), its variational bound is:

$$I(o_2;s_2) \ \leq \ \int_{o_1}\int_{o_2}\int_{s_1'}\int_{s_2'}\int_{s'}\int_{r}\int_{a} p(o_1,o_2,s_1',s_2',s',r,a) d_{o_1} d_{o_2} d_{s_1'} d_{s_2'} d_{s'} d_r d_a \int_{s_2} p(s_2|o_2) \log \frac{p(s_2|o_2)}{q_2(s_2)} d_{s_2} \tag{23}$$

Next, we focus on the second term $I(s_1;s_1',r|a)$ in Eq. (17). According to the definition, the conditional multual information of variables s_1, s_1' and r given a is:

$$I(s_1;s_1',r|a) = \int_{s_1}\int_{s_1'}\int_{r}\int_{a} p(s_1,s_1',r,a) \log \frac{p(s_1',r|s_1,a)}{p(s_1',r|a)} d_{s_1} d_{s_1'} d_r d_a \tag{24}$$

Since it is difficult to compute $p(s_1', r|s_1, a)$, we use distribution $g_1(s_1', r|s_1, a)$ learned form a neural network to approximate it. Since the KL-divergence between distributions $p(s_1', r|s_1, a)$ and $g_1(s_1', r|s_1, a)$ is always non-negative, we have:

$$\begin{aligned} KL(p(s_1', r|s_1, a)||g_1(s_1', r|s_1, a)) \ \geq 0 \implies & \int_{s_1}\int_{s_1'}\int_r\int_a p(s_1, s_1', r, a)\log p(s_1', r|s_1, a) d_{s_1} d_{s_1'} d_r d_a \ \geq \\ & \int_{s_1}\int_{s_1'}\int_r\int_a p(s_1, s_1', r, a)\log g_1(s_1', r|s_1, a) d_{s_1} d_{s_1'} d_r d_a \end{aligned} \tag{25}$$

Substituting Eq. (25) into Eq. (24), $I(s_1; s_1', r|a)$ is reshaped as:

$$\begin{aligned} I(s_1; s_1', r|a) \ \geq & \int_{s_1}\int_{s_1'}\int_r\int_a p(s_1, s_1', r, a)\log \frac{g_1(s_1', r|s_1, a)}{p(s_1', r|a)} d_{s_1} d_{s_1'} d_r d_a \\ = & \int_{s_1}\int_{s_1'}\int_r\int_a p(s_1, s_1', r, a)\log g_1(s_1', r|s_1, a) d_{s_1} d_{s_1'} d_r d_a - \int_{s_1'}\int_r\int_a p(s_1', r, a)\log p(s_1', r|a) d_{s_1'} d_r d_a \end{aligned} \tag{26}$$

Notice that the second term $\int_{s_1'}\int_r\int_a p(s_1', r, a)\log p(s_1', r|a) d_{s_1'} d_r d_a$ in Eq. (26) is independent of the optimization of the S2R model, so we can directly drop it. Then, Eq. (26) is equivalent to:

$$\begin{aligned} I(s_1; s_1', r|a) \ \geq & \int_{s_1}\int_{s_1'}\int_r\int_a p(s_1, s_1', r, a)\log g_1(s_1', r|s_1, a) d_{s_1} d_{s_1'} d_r d_a \\ = & \int_{o_1}\int_{o_2}\int_{s_1'}\int_{s_2'}\int_{s'}\int_r\int_a\int_{s_1} p(o_1, o_2, s_1', s_2', s', r, a, s_1)\log g_1(s_1', r|s_1, a) d_{o_1} d_{o_2} d_{s_1'} d_{s_2'} d_{s'} d_r d_a d_{s_1} \end{aligned} \tag{27}$$

Considering variable s_1 depends only on variable o_1, we get the following variational bound for term $I(s_1; s_1', r|a)$:

$$I(s_1; s_1', r|a) \ \geq \int_{o_1}\int_{o_2}\int_{s_1'}\int_{s_2'}\int_{s'}\int_r\int_a p(o_1, o_2, s_1', s_2', s', r, a) d_{o_1} d_{o_2} d_{s_1'} d_{s_2'} d_{s'} d_r d_a \int_{s_1} p(s_1|o_1)\log g_1(s_1', r|s_1, a) d_{s_1} \tag{28}$$

Similarly, for the fourth term $I(s_2; s_2', r|a)$ in Eq. (17), its variational bound is:

$$I(s_2; s_2', r|a) \ \geq \int_{o_1}\int_{o_2}\int_{s_1'}\int_{s_2'}\int_{s'}\int_r\int_a p(o_1, o_2, s_1', s_2', s', r, a) d_{o_1} d_{o_2} d_{s_1'} d_{s_2'} d_{s'} d_r d_a \int_{s_2} p(s_2|o_2)\log g_2(s_2', r|s_2, a) d_{s_2} \tag{29}$$

Finally, we derive the variational bound for the last term $I(s; s', r|a)$ in Eq. (17). According to the definition, the conditional multual information of variables s, s' and r given a is:

$$I(s; s', r|a) = \int_s\int_{s'}\int_r\int_a p(s, s', r, a)\log \frac{p(s', r|s, a)}{p(s', r|a)} d_s d_{s'} d_r d_a \tag{30}$$

Since $p(s', r|s, a)$ is intractable, we use distribution $g(s', r|s, a)$ learned from a nueral network to approximate it. Considering the non-negative of the KL-divergence between distributions $p(s', r|s, a)$ and $g(s', r|s, a)$, we have:

$$\begin{aligned} KL(p(s', r|s, a)||g(s', r|s, a)) \ \geq 0 \implies & \int_s\int_{s'}\int_r\int_a p(s, s', r, a)\log p(s', r|s, a) d_s d_{s'} d_r d_a \ \geq \\ & \int_s\int_{s'}\int_r\int_a p(s, s', r, a)\log g(s', r|s, a) d_s d_{s'} d_r d_a \end{aligned} \tag{31}$$

Therefore, $I(s; s', r|a)$ is bounded by:

$$\begin{aligned} I(s; s', r|a) \ \geq & \int_s\int_{s'}\int_r\int_a p(s, s', r, a)\log \frac{g(s', r|s, a)}{p(s', r|a)} d_s d_{s'} d_r d_a \\ = & \int_s\int_{s'}\int_r\int_a p(s, s', r, a)\log g(s', r|s, a) d_s d_{s'} d_r d_a - \int_{s'}\int_r\int_a p(s', r, a)\log p(s', r|a) d_{s'} d_r d_a \end{aligned} \tag{32}$$

In Eq. (32), the second term $\int_{s'}\int_r\int_a p(s', r, a)\log p(s', r|a) d_{s'} d_r d_a$ can be ignored, then, we have:

$$\begin{aligned} I(s; s', r|a) \ \geq & \int_s\int_{s'}\int_r\int_a p(s, s', r, a)\log g(s', r|s, a) d_s d_{s'} d_r d_a \\ = & \int_{o_1}\int_{o_2}\int_{s_1'}\int_{s_2'}\int_{s'}\int_r\int_a\int_{s_1}\int_{s_2}\int_s p(o_1, o_2, s_1', s_2', s', r, a, s_1, s_2, s) \\ & \log g(s', r|s, a) d_{o_1} d_{o_2} d_{s_1'} d_{s_2'} d_{s'} d_r d_a d_{s_1} d_{s_2} d_s \end{aligned} \tag{33}$$

By using the joint probability density function in Eq. (18), the variational bound for term $I(s; s', r|a)$ is:

$$\begin{aligned} I(s; s', r|a) \ \geq & \int_{o_1}\int_{o_2}\int_{s'_1}\int_{s'_2}\int_{s'}\int_{r}\int_{a} p(o_1, o_2, s'_1, s'_2, s', r, a) d_{o_1} d_{o_2} d_{s'_1} d_{s'_2} d_{s'} d_r d_a \cdot \\ & \int_{s_1}\int_{s_2}\int_{s} p(s_1|o_1) p(s_2|o_2) p(s|s_1, s_2) \log g(s', r|s, a) d_{s_1} d_{s_2} d_s \end{aligned} \tag{34}$$

With the variational bounds listed in Eq. (22), Eq. (23), Eq. (28), Eq. (29) and Eq. (34), the final variational upper bound of the two-view MCEB objective in Eq. (16) is summarized as:

$$\begin{aligned} \beta_1 I(o_1; s_1|s'_1, r, a) + \beta_2 I(o_2; s_2|s'_2, r, a) - I(s; s', r|a) \ \leq & \int_{o_1}\int_{o_2}\int_{s'_1}\int_{s'_2}\int_{s'}\int_{r}\int_{a} p(o_1, o_2, s'_1, s'_2, s', r, a) d_{o_1} d_{o_2} d_{s'_1} d_{s'_2} \\ d_{s'} d_r d_a \Big(\beta_1 \int_{s_1} p(s_1|o_1) \log \frac{p(s_1|o_1)}{q_1(s_1)} d_{s_1} + \beta_2 \int_{s_2} p(s_2|o_2) \log \frac{p(s_2|o_2)}{q_2(s_2)} d_{s_2} - \beta_1 \int_{s_1} p(s_1|o_1) \log g_1(s'_1, r|s_1, a) d_{s_1} \\ - \beta_2 \int_{s_2} p(s_2|o_2) \log g_2(s'_2, r|s_2, a) d_{s_2} - \int_{s_1}\int_{s_2}\int_{s} p(s_1|o_1) p(s_2|o_2) p(s|s_1, s_2) \log g(s', r|s, a) d_{s_1} d_{s_2} d_s \Big) \end{aligned} \tag{35}$$

In the actual implementation, we use the Monte Carlo sampling to sample empirical data to approximate o_1, o_2, s'_1, s'_2, s', r and a, then, Eq. (35) is simplified as:

$$\begin{aligned} & \beta_1 I(o_1; s_1|s'_1, r, a) + \beta_2 I(o_2; s_2|s'_2, r, a) - I(s; s', r|a) \ \leq \\ & \frac{1}{M} \sum^{M} \Big(\beta_1 \int_{s_1} p(s_1|o_1) \log \frac{p(s_1|o_1)}{q_1(s_1)} d_{s_1} + \beta_2 \int_{s_2} p(s_2|o_2) \log \frac{p(s_2|o_2)}{q_2(s_2)} d_{s_2} - \beta_1 \int_{s_1} p(s_1|o_1) \log g_1(s'_1, r|s_1, a) d_{s_1} \\ & - \beta_2 \int_{s_2} p(s_2|o_2) \log g_2(s'_2, r|s_2, a) d_{s_2} - \int_{s_1}\int_{s_2}\int_{s} p(s_1|o_1) p(s_2|o_2) p(s|s_1, s_2) \log g(s', r|s, a) d_{s_1} d_{s_2} d_s \Big) \end{aligned} \tag{36}$$

Where M is the size of the sampled data.

B IMPLEMENTATION DETAILS

In our implementation, both **actor and critic** are parameterized by a 3-layer fully connected network of 256 units with the ReLU activations. For the **encoder**, it is a convolutional network that contains four convolutional layers followed by the ReLU activations. The kernel size of the convolutional layer is 3×3, and we use stride 2 for the first layer and stride 1 for the rest layers. The output of the convolutional network is fed into a fully connected layer to project into a 50-dimension feature vector and further passed a Layer Normalization. For other networks (the feature fusion module, view-specific predictor, and the multi-view predictor) in the **S2R model**, we use the same architecture for them and implement them as three dense layers of 256 units with the ReLU activations. In Table 2, we show a full list of hyper-parameters used for our experiments.

C ADDITIONAL DMCONTROL RESULTS

For the image distractor setting and natural video setting, Figure 8 and Figure 9 show the performance of S2R + SAC, RAD, and DBC on 6 DMControl tasks. In both settings, S2R + SAC performs comparable or substantially better than RAD, and consistently outperforms DBC, showing its ability to learn efficient and robust representations. As expected, the MCEB objective urges the S2R + SAC agent to pay attention to the robot control task itself, ignore the task-independent details in the environment background, and be more robust to the visual noise in the environment. For ablation studies, we choose the cheetah run and walker walk tasks and compare the performance of S2R with its variants in Figure 10, including the regularization factors in the MCEB objective, the predictive data (Y_1, Y_2, and Y) included in the MCEB objective, and different forms of the optimization objective. Results confirm the correctness of the MCEB β value in Table 2, the rationality of learning representations based on the latent transition dynamics and reward functions, and the efficiency of integrating multi-view data with CEB in the MCEB objective. These all are significant factors for the design and success of S2R.

Table 2: Full list of hyper-parameters used for the DMControl suite.

Hyperparameter	**Value**
Augmentation	Crop
Observation	100×100
Cropped observation	84×84
Replay buffer capacity	10^5
Initial steps	1000
Total trainning steps	500000
Stacked Frames	3
Action Repeat	2 finger spin; walker walk 4 cheetah run; ball-in-cup catch; reacher easy; walker run 8 cartpole swingup
Evaluation episodes	10
Optimizer	Adam
Learning rate (encoder/policy/Q Function)	$2e-4$ cheetah run; $1e-3$ otherwise
Learning rate (α)	$1e-4$
Batch size	512
Q Function EMA τ	0.01
Encoder EMA τ	0.05
Critic target update freq	2
MCEB β (default setting)	$1e-4 \rightarrow 1e-3$ cheetah run; $1e-3 \rightarrow 1e-2$ otherwise
MCEB β (Random image setting)	$1e-4 \rightarrow 1e-2$
MCEB β (Natural video setting)	$1e-4 \rightarrow 1e-2$
Convolutional layers	4
Number of filters	32
Non-linearity	ReLU
Encoder feature dimension	50
Discount factor γ	0.99
Initial temperature	0.1

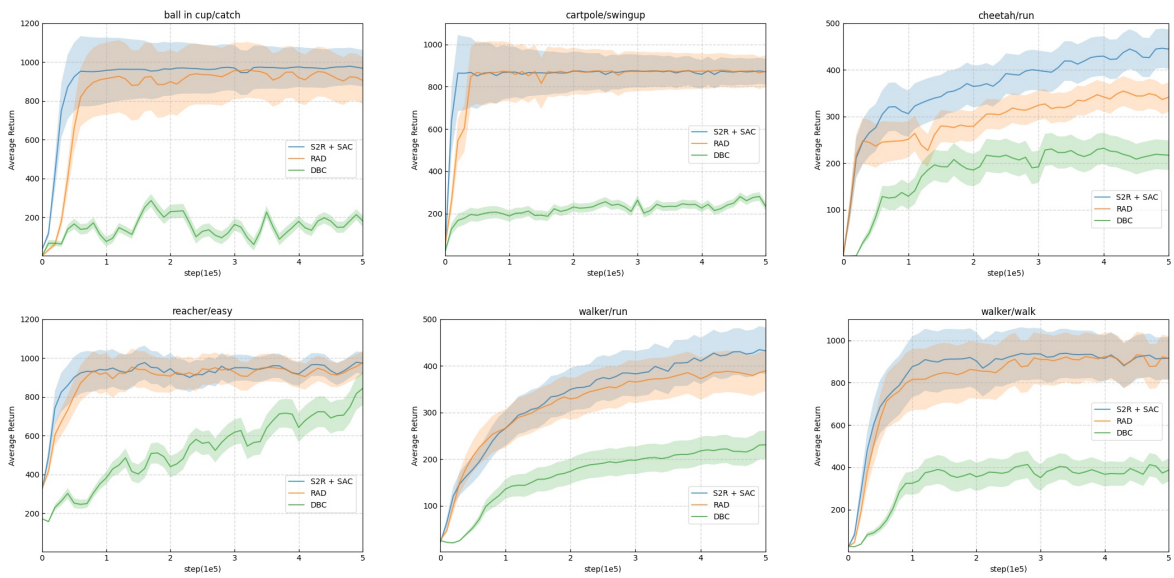

Figure 8: Performance of S2R + SAC over three seeds with mean and one standard error in the image distractor DMControl setting. We benchmark S2R + SAC with bisimulation metric-based method (DBC) and best-performing method RAD. S2R + SAC performs comparable or better than RAD and significantly improves the performance of DBC, on all 6 pixel-based control tasks.

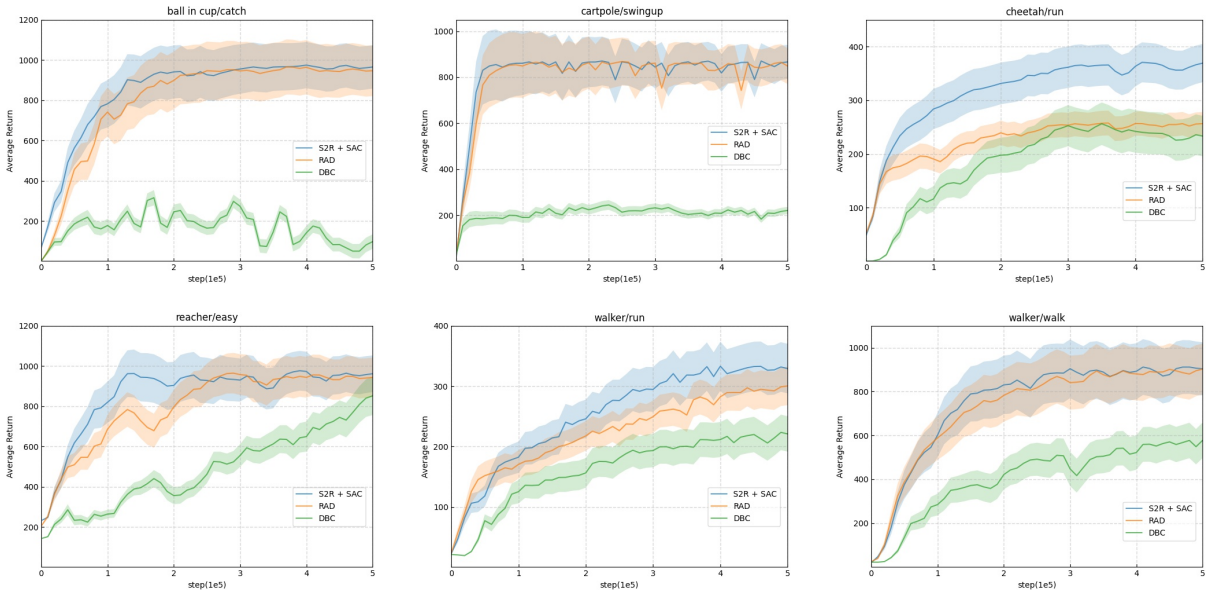

Figure 9: Performance of S2R + SAC over three seeds with mean and one standard error in the natural video DMControl setting. We benchmark S2R + SAC with DBC and RAD. S2R + SAC again performs comparable or better than RAD and DBC on all 6 pixel-based control tasks.

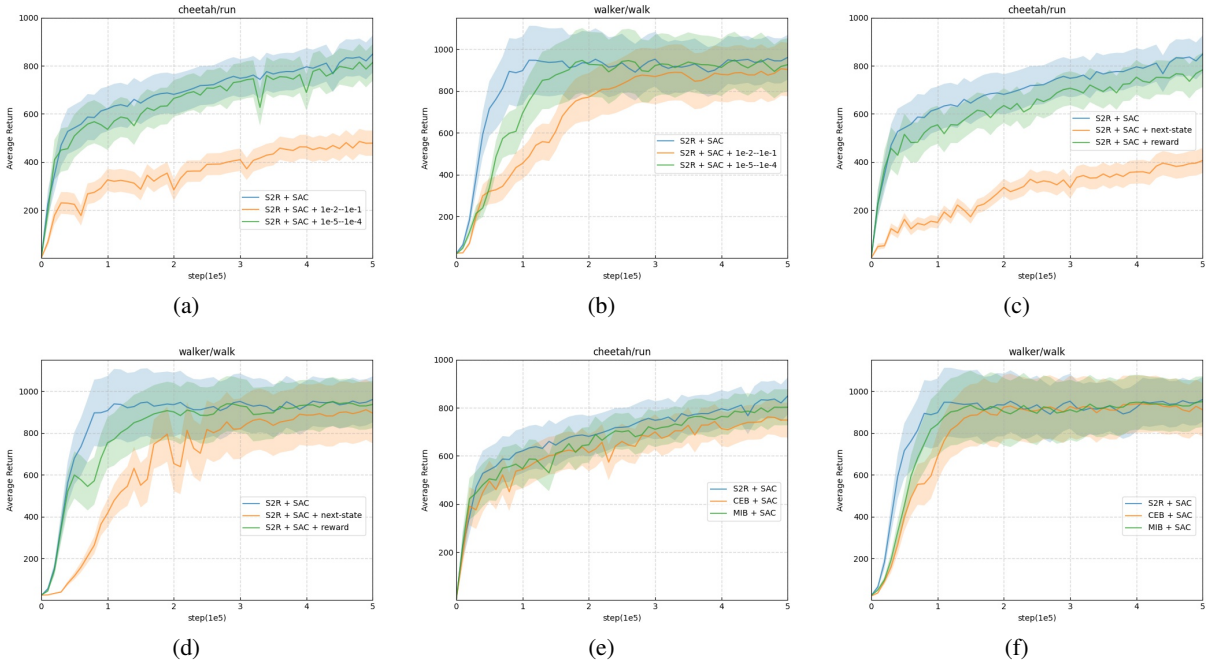

Figure 10: Performance of S2R + SAC over three seeds with mean and one standard error in the default DMControl setting for ablation studies. We compare the performance of S2R with its variants, i.e., the regularization factors in the MCEB objective in (a) and (b), predictive data (Y_1, Y_2, and Y) included in the MCEB objective in (c) and (d), and different forms of the optimization objective in (e) and (f). Results show that choosing suitable values for the regularization factors, simultaneously predicting latent transition dynamics and reward functions, together with the MCEB objective, is significant for the success of S2R.