Self-Supervised Representations for Multi-View Reinforcement Learning (Supplementary Material)

Huanhuan Yang[1] **Dianxi Shi** [*2,3,1] **Guojun Xie**[4] **Yingxuan Peng**[1] **Yi Zhang**[2] **Yantai Yang**[3] **Shaowu Yang**[1]

[1]College of Computer, National University of Defense Technology, Changsha, China
[2]Artificial Intelligence Research Center, Defense Innovation Institute, Beijing, China
[3]Tianjin Artificial Intelligence Innovation Center, Tianjin, China
[4]College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China

A DERIVATION OF THE TWO-VIEW CEB LOSS

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

$$\begin{aligned}\mathbf{obj}: \quad & \min_{Z,Z_1,Z_2} \beta_1 I(X_1;Z_1|Y_1) + \beta_2 I(X_2;Z_2|Y_2) - I(Z;Y) \\ & = \min_{z,z_1,z_2} \beta_1 I(s_1;z_1|z_1',r,a) + \beta_2 I(s_2;z_2|z_2',r,a) - I(z;z',r|a) \\ \mathbf{s.t.}: \quad & Z = f_\theta(Z_1,Z_2) \Rightarrow z = f_\theta(z_1,z_2)\end{aligned} \tag{15}$$

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:

$$\begin{aligned}\mathbf{obj}: \quad & \min_{Z,Z_1,Z_2} \beta_1[I(X_1;Z_1) - I(Z_1;Y_1)] + \beta_2[I(X_2;Z_2) - I(Z_2;Y_2)] - I(Z;Y) \\ & = \min_{z,z_1,z_2} \beta_1(I(s_1;z_1) - I(z_1;z_1',r|a)) + \beta_2(I(s_2;z_2) - I(z_2;z_2',r|a)) - I(z;z',r|a) \\ \mathbf{s.t.}: \quad & Z = f_\theta(Z_1,Z_2) \Rightarrow z = f_\theta(z_1,z_2)\end{aligned} \tag{16}$$

To begin with, we give the joint probability density function of variables s_1, s_2, z_1, z_2, z, z_1', z_2', z', r and a. Since z_1 is learned from s_1, z_2 is learned from s_2, z is fused by z_1 and z_2, thus, based on the Bayes' rule, this joint probability density function can be expressed as:

$$\begin{aligned}p(s_1,s_2,z_1,z_2,z,z_1',z_2',z',r,a) = & \, p(z|s_1,s_2,z_1,z_2,z_1',z_2',z',r,a) \cdot p(z_1|s_1,s_2,z_2,z_1',z_2',z',r,a) \cdot \\ & p(z_2|s_1,s_2,z_1',z_2',z',r,a) \cdot p(s_1,s_2,z_1',z_2',z',r,a) \\ = & \, p(z|z_1,z_2) \cdot p(z_1|s_1) \;\cdot\; p(z_2|s_2) \;\cdot\; p(s_1,s_2,z_1',z_2',z',r,a)\end{aligned} \tag{17}$$

Then, we analysis the first term $I(s_1;z_1)$ in Eq. (16), according to the standard definition of the mutual information, the mutual information between s_1 and z_1 is:

$$I(s_1;z_1) = \int d_{s_1} d_{z_1} \; p(s_1,z_1) \log \frac{p(z_1|s_1)}{p(z_1)} \tag{18}$$

Due to the intractable of $p(z_1)$, we use the variational distribution $q_1(z_1)$ to approximate it. Considering the non-negative property of the Kullback-Leibler divergence (KL-divergence), we can infer that:

$$KL(p(z_1)||q_1(z_1)) \;\geq\; 0 \Longrightarrow \int d_{z_1} \; p(z_1) \log p(z_1) \;\geq\; \int d_{z_1} \; p(z_1) \log q_1(z_1) \tag{19}$$

[*]Corresponding author (dxshi@nudt.edu.cn).

Accepted for the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022).

Substituting Eq. (19) into Eq. (18), we have:

$$\begin{aligned} I(s_1; z_1) &\leq \int d_{s_1} d_{z_1}\ p(s_1, z_1) \log \frac{p(z_1|s_1)}{q_1(z_1)} \\ &= \int d_{s_1} d_{s_2} d_{z_1'} d_{z_2'} d_{z'} d_r d_a d_{z_1}\ p(s_1, s_2, z_1', z_2', z', r, a, z_1) \log \frac{p(z_1|s_1)}{q_1(z_1)} \end{aligned} \tag{20}$$

Considering variable z_1 only depends on variable s_1, we get the following variational bound for term $I(s_1; z_1)$:

$$I(s_1; z_1)\ \leq\ \int d_{s_1} d_{s_2} d_{z_1'} d_{z_2'} d_{z'} d_r d_a\ p(s_1, s_2, z_1', z_2', z', r, a) \int d_{z_1}\ p(z_1|s_1) \log \frac{p(z_1|s_1)}{q_1(z_1)} \tag{21}$$

Similarly, for the third term $I(s_2; z_2)$ in Eq. (16), its variational bound is:

$$I(s_2; z_2)\ \leq\ \int d_{s_1} d_{s_2} d_{z_1'} d_{z_2'} d_{z'} d_r d_a\ p(s_1, s_2, z_1', z_2', z', r, a) \int d_{z_2}\ p(z_2|s_2) \log \frac{p(z_2|s_2)}{q_2(z_2)} \tag{22}$$

Next, we focus on the second term $I(z_1; z_1', r|a)$ in Eq. (16). According to the definition, the conditional multual information of variables z_1, z_1' and r given a is:

$$I(z_1; z_1', r|a) = \int d_{z_1} d_{z_1'} d_r d_a\ p(z_1, z_1', r, a) \log \frac{p(z_1', r|z_1, a)}{p(z_1', r|a)} \tag{23}$$

Since it is difficult to compute $p(z_1', r|z_1, a)$, we use distribution $g_{\omega_1}(z_1', r|z_1, a)$ learned form a neural network to approximate it. Since the KL-divergence between distributions $p(z_1', r|z_1, a)$ and $g_{\omega_1}(z_1', r|z_1, a)$ is always non-negative, we have:

$$\begin{aligned} KL(p(z_1', r|z_1, a)||g_{\omega_1}(z_1', r|z_1, a))\ \geq 0\ \Longrightarrow \int d_{z_1} d_{z_1'} d_r d_a\ p(z_1, z_1', r, a) \log p(z_1', r|z_1, a)\ \geq \\ \int d_{z_1} d_{z_1'} d_r d_a\ p(z_1, z_1', r, a) \log g_{\omega_1}(z_1', r|z_1, a) \end{aligned} \tag{24}$$

Substituting Eq. (24) into Eq. (23), $I(z_1; z_1', r|a)$ is reshaped as:

$$\begin{aligned} I(z_1; z_1', r|a)\ &\geq \int d_{z_1} d_{z_1'} d_r d_a\ p(z_1, z_1', r, a) \log \frac{g_{\omega_1}(z_1', r|z_1, a)}{p(z_1', r|a)} \\ &= \int d_{z_1} d_{z_1'} d_r d_a\ p(z_1, z_1', r, a) \log g_{\omega_1}(z_1', r|z_1, a) - \underbrace{\int d_{z_1'} d_r d_a\ p(z_1', r, a) \log p(z_1', r|a)}_{\text{dropped}} \end{aligned} \tag{25}$$

Notice that the $\int d_{z_1'} d_r d_a\ p(z_1', r, a) \log p(z_1', r|a)$ term in Eq. (25) is independent of the optimization of the S2R model, so we can directly drop it. Then, Eq. (25) is equivalent to:

$$\begin{aligned} I(z_1; z_1', r|a)\ &\geq \int d_{z_1} d_{z_1'} d_r d_a\ p(z_1, z_1', r, a) \log g_{\omega_1}(z_1', r|z_1, a) \\ &= \int d_{s_1} d_{s_2} d_{z_1'} d_{z_2'} d_{z'} d_r d_a d_{z_1}\ p(s_1, s_2, z_1', z_2', z', r, a, z_1) \log g_{\omega_1}(z_1', r|z_1, a) \end{aligned} \tag{26}$$

Considering variable z_1 only depends on variable s_1, we get the following variational bound for term $I(z_1; z_1', r|a)$:

$$I(z_1; z_1', r|a)\ \geq\ \int d_{s_1} d_{s_2} d_{z_1'} d_{z_2'} d_{z'} d_r d_a\ p(s_1, s_2, z_1', z_2', z', r, a) \int d_{z_1}\ p(z_1|s_1) \log g_{\omega_1}(z_1', r|z_1, a) \tag{27}$$

Similarly, for the fourth term $I(z_2; z_2', r|a)$ in Eq. (16), its variational bound is:

$$I(z_2; z_2', r|a)\ \geq\ \int d_{s_1} d_{s_2} d_{z_1'} d_{z_2'} d_{z'} d_r d_a\ p(s_1, s_2, z_1', z_2', z', r, a) \int d_{z_2}\ p(z_2|s_2) \log g_{\omega_2}(z_2', r|z_2, a) \tag{28}$$

Finally, we derive the variational bound for the last term $I(z; z', r|a)$ in Eq. (16). According to the definition, the conditional multual information of variables z, z' and r given a is:

$$I(z; z', r|a) = \int d_z d_{z'} d_r d_a\ p(z, z', r, a) \log \frac{p(z', r|z, a)}{p(z', r|a)} \tag{29}$$

Since $p(z', r|z, a)$ is intractable, we use distribution $g_{\omega_{12}}(z', r|z, a)$ learned from a nueral network to approximate it. Considering the non-negative of the KL-divergence between distributions $p(z', r|z, a)$ and $g_{\omega_{12}}(z', r|z, a)$, we have:

$$KL(p(z', r|z, a)||g_{\omega_{12}}(z', r|z, a)) \geq 0 \implies \int d_z d_{z'} d_r d_a \; p(z, z', r, a) \log p(z', r|z, a) \geq \int d_z d_{z'} d_r d_a \; p(z, z', r, a) \log g_{\omega_{12}}(z', r|z, a) \tag{30}$$

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

$$\begin{aligned} I(z; z', r|a) &\geq \int d_z d_{z'} d_r d_a \; p(z, z', r, a) \log \frac{g_{\omega_{12}}(z', r|z, a)}{p(z', r|a)} \\ &= \int d_z d_{z'} d_r d_a \; p(z, z', r, a) \log g_{\omega_{12}}(z', r|z, a) - \underbrace{\int d_{z'} d_r d_a \; p(z', r, a) \log p(z', r|a)}_{\text{dropped}} \end{aligned} \tag{31}$$

In Eq. (31), the $\int d_{z'} d_r d_a \; p(z', r, a) \log p(z', r|a)$ term can be ignored, then, we have:

$$\begin{aligned} I(z; z', r|a) &\geq \int d_z d_{z'} d_r d_a \; p(z, z', r, a) \log g_{\omega_{12}}(z', r|z, a) \\ &= \int d_{s_1} d_{s_2} d_{z'_1} d_{z'_2} d_{z'} d_r d_a d_{z_1} d_{z_2} d_z \; p(s_1, s_2, z'_1, z'_2, z', r, a, z_1, z_2, z) \log g_{\omega_{12}}(z', r|z, a) \end{aligned} \tag{32}$$

By using the joint probability density function in Eq. (17), the variational bound for term $I(z; z', r|a)$ is:

$$\begin{aligned} I(z; z', r|a) \geq \int & d_{s_1} d_{s_2} d_{z'_1} d_{z'_2} d_{z'} d_r d_a \; p(s_1, s_2, z'_1, z'_2, z', r, a) \cdot \\ & \int d_{z_1} d_{z_2} d_z \; p(z_1|s_1) p(z_2|s_2) p(z|z_1, z_2) \log g_{\omega_{12}}(z', r|z, a) \end{aligned} \tag{33}$$

With the variational bounds listed in Eq. (21), Eq. (22), Eq. (27), Eq. (28) and Eq. (33), the final variational upper bound of the two-view CEB objective in Eq. (15) is summarized as:

$$\begin{aligned} &\beta_1 I(s_1; z_1|z'_1, r, a) + \beta_2 I(s_2; z_2|z'_2, r, a) - I(z; z', r|a) \leq \\ &\int d_{s_1} d_{s_2} d_{z'_1} d_{z'_2} d_{z'} d_r d_a \; p(s_1, s_2, z'_1, z'_2, z', r, a) \Big(\beta_1 \int d_{z_1} \; p(z_1|s_1) \left[\log \frac{p(z_1|s_1)}{q_1(z_1)} - \log g_{\omega_1}(z'_1, r|z_1, a) \right] + \beta_2 \\ &\int d_{z_2} \; p(z_2|s_2) \left[\log \frac{p(z_2|s_2)}{q_2(z_2)} - \log g_{\omega_2}(z'_2, r|z_2, a) \right] - \int d_{z_1} d_{z_2} d_z \; p(z_1|s_1) p(z_2|s_2) p(z|z_1, z_2) \log g_{\omega_{12}}(z', r|z, a) \Big) \end{aligned} \tag{34}$$

In the actual implementation, we use the Monte Carlo sampling to sample empirical data to approximate s_1, s_2, z'_1, z'_2, z', r and a, then, Eq. (34) is simplified as:

$$\begin{aligned} &\beta_1 I(s_1; z_1|z'_1, r, a) + \beta_2 I(s_2; z_2|z'_2, r, a) - I(z; z', r|a) \leq \\ &\frac{1}{M} \sum^{M} \Big(\beta_1 \left[D_{KL} \left(p(z_1|s_1) || q_1(z_1) \right) - \mathbb{E}_{z_1 \sim p(z_1|s_1)} \log g_{\omega_1}(z'_1, r|z_1, a) \right] + \\ &\beta_2 \left[D_{KL} \left(p(z_2|s_2) || q_2(z_2) \right) - \mathbb{E}_{z_2 \sim p(z_2|s_2)} \log g_{\omega_2}(z'_2, r|z_2, a) \right] - \\ &\mathbb{E}_{z_1 \sim p(z_1|s_1)} \mathbb{E}_{z_2 \sim p(z_2|s_2)} \mathbb{E}_{z \sim p(z|z_1, z_2)} \left[\log g_{\omega_{12}}(z', r|z, a) \right] \Big) \end{aligned} \tag{35}$$

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 and target encoder**, both of them consist of four convolutional layers followed by the

ReLU activations. The kernel size of the convolutional layer is 3×3. We use stride 2 for the first layer and stride 1 for the rest layers. The output of the last convolutional layer is fed into a fully-connected layer to project into a 50-dimension feature vector and further passed a Layer Normalization. For **feature fusion module and MLPs**, we use the same architecture for them and implement them as three dense layers of 256 units with the ReLU activations. For **view-specific predictor and multi-view predictor**, we implement them as one network, i.e., 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.

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

Hyperparameter	**Value**
Augmentation	Crop
Image states	100×100
Cropped image states	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 (M)	512
View number (N)	2
Q Function EMA $\tau_{\varphi_i}, i = 1, 2$	0.01
Encoder EMA τ_ρ	0.05
Critic/encoder target update freq	2
MCEB β_j (default setting), $j \in [1, N]$	$1e-4 \rightarrow 1e-3$ cheetah run; $1e-3 \rightarrow 1e-2$ otherwise
MCEB β_j (random image setting), $j \in [1, N]$	$1e-4 \rightarrow 1e-2$
MCEB β_j (natural video setting), $j \in [1, N]$	$1e-4 \rightarrow 1e-2$
Convolutional layers	4
Number of filters	32
Non-linearity	ReLU
Encoder feature dimension	50
Discount factor γ	0.99
Initial temperature	0.1

C ADDITIONAL DMCONTROL RESULTS

For the image distractor setting and natural video setting, Fig. 9 and Fig. 10 show the performance of S2R + SAC, RAD, and DBC on 6 DMControl tasks. In both settings, S2R + SAC performs comparably or better than RAD, and substantially 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 task-independent details in the environment background, and thus 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 Fig. 11, including the regularization factors, predictive data (Y_1, Y_2, and Y), optimization objectives, and the number of views included in the MCEB objective. Results confirm the correctness of the value of MCEB β_j given in Table 2, the rationality of learning representations based on the latent transition function and reward function, 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. Besides, the MCEB objective in S2R can take advantage of the multi-view data to learn robust representations.

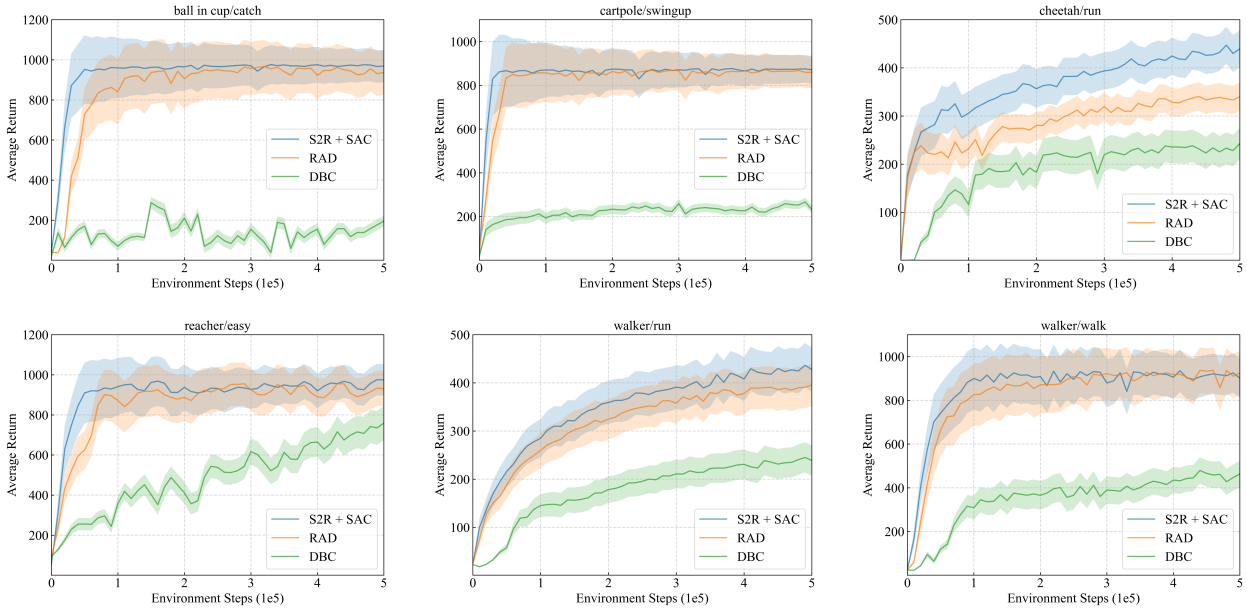


Figure 9: Performance of S2R + SAC over five seeds with mean and one standard error in the image distractor DMControl setting. We benchmark it with RAD and DBC. S2R + SAC performs comparably or better than RAD and significantly improves the performance of DBC, on all 6 pixel-based control tasks.

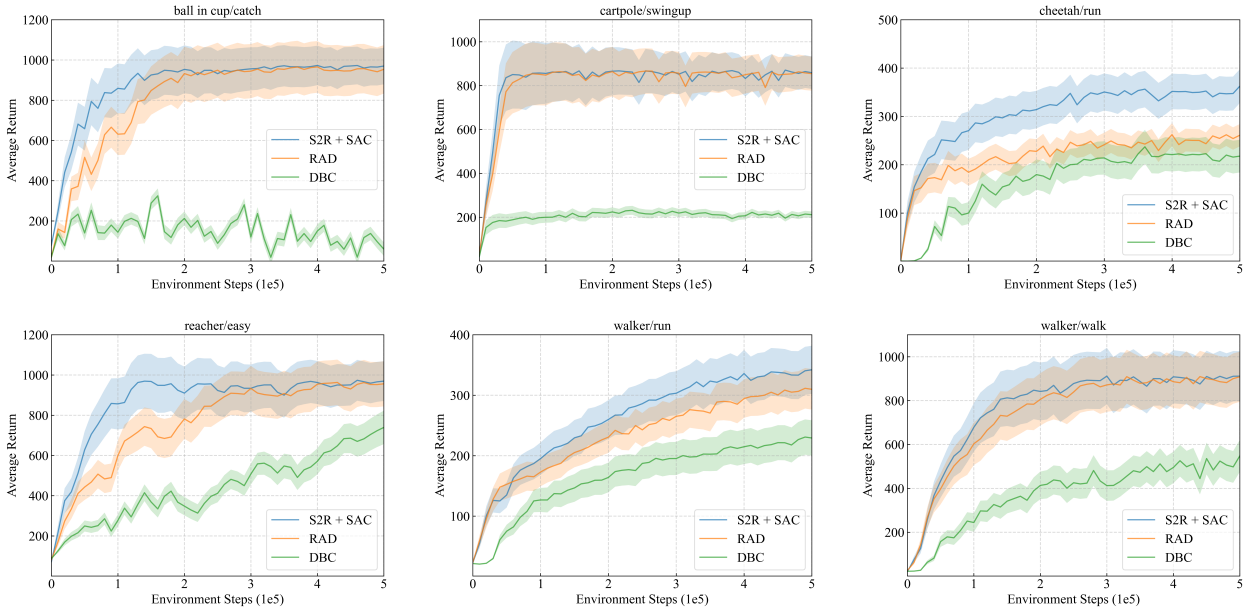
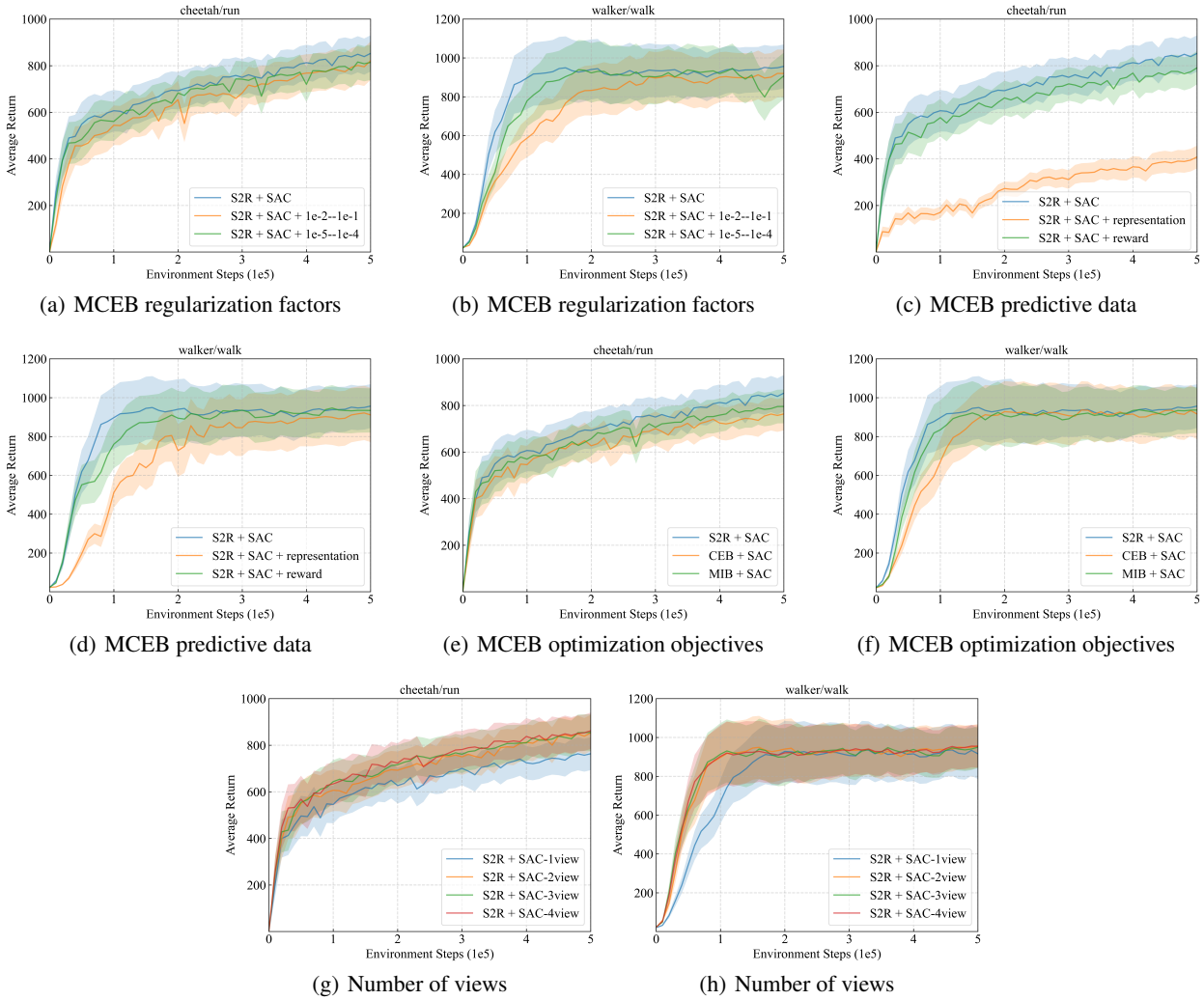


Figure 10: Performance of S2R + SAC over five seeds with mean and one standard error in the natural video DMControl setting. We benchmark it with RAD and DBC. S2R + SAC again performs comparably or better than RAD and DBC, on all 6 pixel-based control tasks.



(a) MCEB regularization factors (b) MCEB regularization factors (c) MCEB predictive data

(d) MCEB predictive data (e) MCEB optimization objectives (f) MCEB optimization objectives

(g) Number of views (h) Number of views

Figure 11: Performance of S2R + SAC over five 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., regularization factors in (a) and (b), predictive data (Y_1, Y_2, and Y) in (c) and (d), optimization objectives in (e) and (f), and the number of views N in (g) and (h). Results show that choosing suitable values for the regularization factors, and simultaneously predicting the latent transition function and reward function, together with the MCEB objective, is significant for the success of S2R. Besides, the increase of N generally improves (comparably or better than the two-view case) the performance of the S2R method.