1 Paper Details

1.1 Detailed Parameter Settings

Table 1: Hyperparameters for continuous control tasks.

	Hyperparameters	**Value**
Agent	Training steps	chosen from {1M, 2M}
	Buffer size	1×10^6
	Batch size	256
	Evaluation interval	5000
	Update interval	1
	Random seed	10, 100, 1000, 10000
SAC	γ	0.99
	Init α	1.0
	Actor learning rate	0.0003
	Critic learning rate	0.0003
	α learning rate	0.0003
	Hidden network sizes	[256, 256]
ReMERN	Error learning rate	0.0003
	Error hidden network sizes	[256, 256, 256]
	Init temperature τ	10.0
LFIW	Buffer size $\|\mathcal{D}\|$	1×10^6
	Buffer size $\|\mathcal{D}_{\mathrm{f}}\|$	1×10^5
	κ_ψ hidden network sizes	[128, 128]
	Temperature T	7.5
SCIM (ours)	si-buffer size $\|\mathcal{D}_{\mathrm{si}}\|$	2×10^5
	Init $R^{\max}$	-1000

The detailed parameter settings are listed in Tab. 1. The algorithms ReMERT, ReMERN, LFIW, and SCIM (ours) all contain the full parameters of *Agent* and *SAC*. ReMERN introduces an error network to calculate the cumulative Bellman error with a learning rate of 0.0003 and a hidden network of [256, 256, 256]. In addition, ReMERN maintains a moving average of the temperatures initialized as 10.0 to perform the weighting.

The replay buffer size $|\mathcal{D}_{\mathrm{f}}|$ of LFIW affects the number of experiences we treat as "on-policiness". According to LFIW's previous experience, the performance is relatively stable for $|\mathcal{D}_{\mathrm{f}}| = 1 \times 10^5$. The hidden network sizes of κ_ψ are [128, 128], and the temperature hyperparameter T for self-normalization to the importance weights is 7.5. The normalization is:

$$\tilde{\kappa}_\psi(s,a) := \frac{\kappa_\psi(s,a)^{1/T}}{\mathbb{E}_{\mathcal{D}_{\mathrm{s}}}\left[\kappa_\psi(s,a)^{1/T}\right]}.$$

ReMERT, ReMERN, and SCIM maintain uniform parameters with LFIW in calculating the likelihood-free importance weight. The difference is that SCIM sets the size of si-buffer to $|\mathcal{D}_{\mathrm{ho}}| = 2 \times 10^5$. It intends to prevent overfitting and catastrophic forgetting due to a lack of diversity.

For Atari games, the parameter settings are the same as A2C+SIL (https://github.com/junhyukoh/self-imitation-learning).

1.2 Implementation Details

Baselines. The source codes refer to ReMERT (https://github.com/AIDefender/MyDiscor). Our method SCIM also alters based on this and adds only a dozen lines of code. Some implementation details about SCIM are as follows.

Compute entropy $\mathcal{H}(\pi(\cdot|s))$. For the algorithms in discrete action spaces, the entropy is calculated by $\mathcal{H}(y) = -\sum_j y_j \log y_j$. However, this study only emphasizes continuous control tasks. Thus, we use the differential entropy of Gaussian distribution:

$$\mathcal{H}\left[\mathcal{N}\left(\mu, \sigma^2\right)\right] = \frac{1}{2}\log 2\pi e \sigma^2.$$

Note that the result of the differential entropy may have negative values. We normalize them so that they can be used as sample weights:

$$\mathcal{H}(\pi(\cdot|s)) = \frac{\mathcal{H}(\pi(\cdot|s)) - \min \mathcal{H}(\pi(\cdot|s))}{\max \mathcal{H}(\pi(\cdot|s)) - \min \mathcal{H}(\pi(\cdot|s))}.$$

Compute confidence weight $\omega(s,a) := \frac{\mathcal{T}(s,a)}{\mathcal{S}(s,a)}$. To ensure that the $\mathcal{T}(s,a)$ and $\mathcal{S}(s,a)$ have the same magnitude, we compute the confidence weight using the following formula instead:

$$\omega(s,a) := \frac{\exp(\lambda^{t(s,a)+\mathcal{S}(s,a)})}{\mathrm{e}-1},$$

where $\mathcal{T}(s,a) = \sqrt{\ln t(s,a)}$, $t(s,a)$ is the step of sample (s,a) in every episode. $\lambda = 0.996$ and e is the natural logarithm.