

---

# Skill-aware Mutual Information Optimisation for Generalisation in Reinforcement Learning

---

Anonymous Author(s)

Affiliation

Address

email

## Abstract

1 Meta-Reinforcement Learning (Meta-RL) agents can struggle to operate across  
2 tasks with varying environmental features that require different optimal *skills* (i.e.,  
3 different modes of behaviours). Using context encoders based on contrastive  
4 learning to enhance the generalisability of Meta-RL agents is now widely studied  
5 but faces challenges such as the requirement for a large sample size, also referred  
6 to as the log- $K$  curse. To improve RL generalisation to different tasks, we first  
7 introduce Skill-aware Mutual Information (SaMI), an optimisation objective that  
8 aids in distinguishing context embeddings according to skills, thereby equipping RL  
9 agents with the ability to identify and execute different skills across tasks. We then  
10 propose Skill-aware Noise Contrastive Estimation (SaNCE), a  $K$ -sample estimator  
11 used to optimise the SaMI objective. We provide a framework for equipping an RL  
12 agent with SaNCE in practice and conduct experimental validation on modified  
13 MuJoCo and Panda-gym benchmarks. We empirically find that RL agents that learn  
14 by maximising SaMI achieve substantially improved zero-shot generalisation to  
15 unseen tasks. Additionally, the context encoder equipped with SaNCE demonstrates  
16 greater robustness to reductions in the number of available samples, thus possessing  
17 the potential to overcome the log- $K$  curse.

## 18 1 Introduction

19 Meta-Reinforcement Learning  
20 (Meta-RL) agents often learn  
21 policies that do not generalise  
22 across tasks in which the  
environmental features and optimal  
skills are different [des Combes  
et al., 2018, Garcin et al., 2024].  
Consider a set of cube-moving  
tasks where an agent must  
move a cube to a goal position  
on a table (Figure 1). These  
tasks become challenging if  
environmental features, such  
as table friction, vary between  
tasks. When facing an unknown  
environment, the agent needs to  
explore effectively, understand the environment, and adjust its behaviour accordingly within an  
episode. For instance, if the agent tries to push a cube across a table covered by a tablecloth and finds  
it “unpushable,” it should infer that the table friction is relatively high and adapt by lifting the cube to

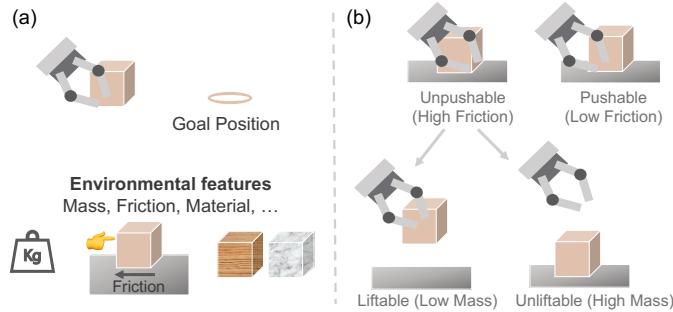


Figure 1: (a) In a cube-moving environment, tasks are defined according to different environmental features. (b) Different tasks have different transition dynamics caused by underlying environmental features, hence optimal skills are different across tasks.

23 avoid friction, rather than continuing to push. Recent advances in Meta-RL [Lee et al., 2020, Agarwal  
24 et al., 2021, Mu et al., 2022, Dunion et al., 2023b,a] have shown promising potential to perceive  
25 and understand environmental features by inferring context embeddings from a small number of  
26 interactions in the environment. The context embedding is expected to capture the distribution of  
27 tasks and efficiently infer new tasks. Meta-RL methods then train a policy conditioned on the context  
28 embedding to generalise to multiple tasks.

29 The context encoder is the key component for capturing the context embedding from recent experi-  
30 ences [Clavera et al., 2019a, Lee et al., 2020], which affects generalisation performance significantly.  
31 Some Meta-RL algorithms [Fu et al., 2021, Wang et al., 2021, Li et al., 2021, Sang et al., 2022]  
32 are equipped with context encoders based on contrastive learning, which uses the InfoNCE lower  
33 bound [Oord et al., 2019] to optimise the mutual information (MI) between context embeddings and  
34 trajectories. InfoNCE is a  $K$ -sample MI estimator and is utilised to learn distinct context embeddings  
35 for each task. Despite significant advancements, integrating contrastive learning with Meta-RL poses  
36 several unresolved challenges, of which two are particularly relevant to this research: **(i) existing**  
37 **context encoders based on contrastive learning do not distinguish tasks that require different**  
38 **skills**; many prior algorithms only pull embeddings of the same tasks together and push those of  
39 different tasks apart. However, for example, a series of cube-moving tasks with high friction may only  
40 require a Pick&Place skill (picking the cube off the table and placing it at the goal position), making  
41 further differentiation unnecessary. **(ii)  $K$ -sample MI estimators are sensitive to the sample size  $K$**   
42 **(i.e., the  $\log\text{-}K$  curse)** [Poole et al., 2019]; a substantial quantity of negative samples is required to  
43 approximate the true MI, which is challenging due to RL’s low sample efficiency [Franke et al., 2021]  
44 and often impractical for achieving accurate MI estimation [Arora et al., 2019, Nozawa and Sato,  
45 2021]. The effectiveness of  $K$ -sample MI estimators breaks down with a small sample size and leads  
46 to a significant performance drop in downstream RL tasks [Mnih and Teh, 2012, Guo et al., 2022].

47 To enhance RL generalisation across different tasks, we propose that the context embeddings should  
48 optimise downstream tasks and indicate whether the current skill remains optimal or requires further  
49 exploration, thereby addressing issue (i). This approach also reduces the necessary sample size and  
50 helps to overcome issue (ii) by concentrating solely on extracting task-relevant MI. Specifically,  
51 we propose a three-step process tailored to RL: (1) We introduce ***Skill-aware Mutual Information***  
52 (***SaMI***), a smaller ground-truth MI that discriminates context embeddings according to skills by  
53 maximising the MI between context embedding, skills and trajectories. Additionally, we provide  
54 a theoretical proof of why introducing skills can make the ground-truth MI smaller, thus making  
55 it easier to optimise; (2) We propose a more data-efficient  $K$ -sample estimator, ***Skill-aware Noise***  
56 ***Contrastive Estimation*** (***SaNCE***), used to optimise SaMI and reduce the negative sample space based  
57 on skills to overcome the  $\log\text{-}K$  curse; (3) We provide a framework to show how a simple objective,  
58 SaMI, can enable Meta-RL agents to autonomously discover skills, and propose a practical skill  
59 definition and skill-aware trajectory sampling method for SaNCE.

60 We demonstrate empirically in MuJoCo [Todorov et al., 2012] and Panda-gym [Gallouédec et al.,  
61 2021] that SaMI improves the zero-shot generalisation performance in sets of previously unseen tasks  
62 with moderate and extreme difficulty. In the MuJoCo and Panda-gym benchmark, SaMI enhances  
63 two Meta-RL algorithms [Yu et al., 2020, Fu et al., 2021] by achieving higher returns/success rates,  
64 especially in moderate and extreme testing tasks. This indicates SaMI’s advantage in encoding  
65 information that enables an agent to execute effective skills in downstream control tasks. SaNCE-  
66 based RL algorithms utilise smaller sample spaces while achieving improved downstream control  
67 performance, indicating their potential to overcome the  $\log\text{-}K$  curse. Visualisation of the learned  
68 context embeddings shows distinct clusters corresponding to different skills, indicating that SaMI is a  
69 step towards more robust and versatile RL agents.

## 70 2 Related works

71 **Meta-RL.** Meta-RL methods train an agent conditioned on context embeddings to improve generali-  
72 sation to unseen tasks. As the key component of Meta-RL, the quality of the context embedding can  
73 significantly affect the agent’s performance. Existing algorithms can be categorised into three types  
74 based on different context embeddings. In the first category, the context embedding is learned by  
75 minimising the downstream RL loss [Rakelly et al., 2019, Yu et al., 2020]. PEARL [Rakelly et al.,  
76 2019] learns probabilistic context embeddings by recovering the value function. Multi-task SAC  
77 + TE (TESAC) [Yu et al., 2020] uses the Task Embedding (TE) [Hausman et al., 2018] as input to

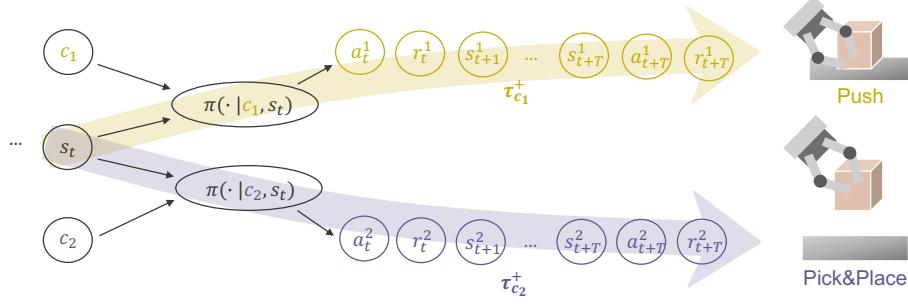


Figure 2: A policy  $\pi$  conditioned on a fixed context embedding  $c$  is defined as a skill  $\pi(\cdot|c)$  (shortened as  $\pi_c$ ). The policy  $\pi$  conditioned on a fixed  $c$  alters the state of the environment in a consistent way, thereby exhibiting a mode of skill. The skill  $\pi(\cdot|c_1)$  moves the cube on the table in trajectory  $\tau_{c_1}^+$  and is referred to as the Push skill; correspondingly, the Pick&Place skill  $\pi(\cdot|c_2)$  takes the cube off the table and places it in the goal position in the trajectory  $\tau_{c_2}^+$ .

78 policies, parameterising the learned policies via a shared embedding space and maximising average  
 79 returns. However, the update signals from the RL loss are stochastic and weak, and may not capture  
 80 the similarity relations among tasks [Fu et al., 2021]. The second category involves learning context  
 81 embeddings through dynamics prediction [Lee et al., 2020, Zhou et al., 2019], which can make the  
 82 context embeddings noisy, as they may model irrelevant dependencies and overlook task-specific  
 83 information [Fu et al., 2021]. The third category employs contrastive learning [Fu et al., 2021, Wang  
 84 et al., 2021, Li et al., 2021, Sang et al., 2022], achieving significant improvements in context learning.  
 85 However, these methods overlook the similarity of skills between different tasks, thus failing to  
 86 achieve effective zero-shot generalisation by executing different skills. Our improvements build upon  
 87 this third category by distinguishing context embeddings according to different optimal skills.

88 **Contrastive learning.** Contrastive learning has been applied to RL due to its significant momentum in  
 89 representation learning in recent years, attributed to its superior effectiveness [Tishby and Zaslavsky,  
 90 2015, Hjelm et al., 2019, Dunion and Albrecht, 2024], ease of implementation [Oord et al., 2019],  
 91 and strong theoretical connection to mutual information (MI) estimation [Poole et al., 2019]. MI  
 92 is often estimated through noise-contrastive estimation (NCE) [Gutmann and Hyvärinen, 2010],  
 93 InfoNCE [Oord et al., 2019], and variational objectives [Hjelm et al., 2019]. InfoNCE has garnered  
 94 recent interest due to its lower variance [Song and Ermon, 2020] and superior performance in  
 95 downstream tasks. However, InfoNCE may underestimate true MI because it is constrained by the  
 96 number of samples  $K$ . To address this issue, CCM [Fu et al., 2021] leverages InfoNCE to learn  
 97 the context embedding with a large number of samples. DOMINO [Mu et al., 2022] reduces the  
 98 total MI by introducing an independence assumption; however, this results in biased information.  
 99 Correspondingly, we focus on proposing an unbiased alternative MI objective and a more data-  
 100 efficient  $K$ -sample estimator tailored for downstream RL tasks, which, to our knowledge, have not  
 101 been addressed in previous research.

### 102 3 Preliminaries

103 **Reinforcement learning.** In Meta-RL, we assume an *environment* is a distribution  $\xi(e)$  of *tasks*  $e$   
 104 (e.g. uniform in our experiments). Each task  $e \sim \xi(e)$  has a similar structure that corresponds to a  
 105 Markov Decision Process (MDP) [Puterman, 2014], defined by  $\mathcal{M}_e = (\mathcal{S}, \mathcal{A}, R, P_e, \gamma)$ , with a state  
 106 space  $\mathcal{S}$ , an action space  $\mathcal{A}$ , a reward function  $R(s_t, a_t)$  where  $s_t \in \mathcal{S}$  and  $a_t \in \mathcal{A}$ , state transition  
 107 dynamics  $P_e(s_{t+1}|s_t, a_t)$ , and a discount factor  $\gamma \in [0, 1]$ . In order to address the problem of zero-  
 108 shot generalisation, we consider the transition dynamics  $P_e(s_{t+1}|s_t, a_t)$  vary across tasks  $e \sim \xi(e)$   
 109 according to multiple *environmental features*  $e = \{e^0, e^1, \dots, e^N\}$  that are not included in states  $s$  and  
 110 can be continuous random variables, such as mass and friction, or discrete random variables, such as  
 111 the cube’s material. For instance, in a cube-moving environment (Figure 1), an agent has different  
 112 tasks that are defined by different environmental features (e.g., mass and friction). The Meta-RL  
 113 agent’s goal is to learn a generalisable policy  $\pi$  that is robust to such dynamic changes. Specifically,  
 114 given a set of training tasks  $e$  sampled from  $\xi_{\text{train}}(e)$ , we aim to learn a policy that can maximise the

115 discounted returns,  $\arg \max_{\pi} \mathbb{E}_{e \sim \xi_{\text{train}}(e)} [\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | a_t \sim \pi(a_t | s_t), s_{t+1} \sim P_e(s_{t+1} | s_t, a_t)]$ ,  
 116 and can produce accurate control for unseen test tasks  $e$  sampled from  $\xi_{\text{test}}(e)$ .

117 **Contrastive learning.** In a Meta-RL setting, the context encoder  $\psi(c | \tau_{0:t})$  first takes the trajectory  
 118  $\tau_{c,0:t} = \{s_0, a_0, r_0, \dots, s_t\}$  from the current episode as input and compresses it into a context  
 119 embedding  $c$ . Then, the policy  $\pi$ , conditioned on context embedding  $c$ , consumes the current state  $s_t$ ,  
 120 outputs the action  $a_t$ . The policy  $\pi$  conditioned on a fixed  $c$  alters the state of the environment in a  
 121 consistent way, thereby exhibiting a mode of skill. As a key component, the embedding  $c$  generated  
 122 by the context encoder  $\psi$  for a task directly determines how the policy behaves. MI is a good measure  
 123 of compression [Goldfeld et al., 2019], hence we focus on a context encoder that optimises the  
 124 InfoNCE objective  $I_{\text{InfoNCE}}(x; y)$ , which is a  $K$ -sample estimator and lower bound of the MI  $I(x; y)$   
 125 [Oord et al., 2019]. Given a query  $x$  and a set  $Y = \{y_1, \dots, y_K\}$  of  $K$  random samples containing one  
 126 positive sample  $y_1$  and  $K - 1$  negative samples from the distribution  $p(y)$ ,  $I_{\text{InfoNCE}}(x; y)$  is obtained  
 127 by comparing pairs sampled from the joint distribution  $x, y_1 \sim p(x, y)$  to pairs  $x, y_k$  built using a set  
 128 of negative examples  $y_{2:K}$ :

$$I_{\text{InfoNCE}}(x; y | \psi, K) = \mathbb{E} \left[ \log \frac{f_{\psi}(x, y_1)}{\frac{1}{K} \sum_{k=1}^K f_{\psi}(x, y_k)} \right]. \quad (1)$$

129 InfoNCE constructs a formal lower bound to the MI, i.e.,  $I_{\text{InfoNCE}}(x; y | \psi, K) \leq I(x; y)$  [Guo et al.,  
 130 2022, Chen et al., 2021]. Given two inputs  $x$  and  $y$ , their *embedding similarity* is  $f_{\psi}(x, y) =$   
 131  $e^{\psi(x)^T \cdot \psi(y_1) / \beta}$ , where  $\psi$  is the context encoder that projects  $x$  and  $y$  into the context embedding  
 132 space, the dot product is used to calculate the similarity score between  $\psi(x), \psi(y)$  pairs [Wu et al.,  
 133 2018, He et al., 2020], and  $\beta$  is a temperature hyperparameter that controls the sensitivity of the  
 134 product. Some previous Meta-RL methods [Lee et al., 2020, Mu et al., 2022] learn a context  
 135 embedding  $c$  by maximising  $I_{\text{InfoNCE}}(c; \tau_c | \psi, K)$  between the context  $c$  embedded from a trajectory  
 136 in the current task, and the historical trajectories  $\tau_c$  under the same environmental features setting.

## 137 4 Skill-aware mutual information optimisation for Meta-RL

### 138 4.1 The *log-K* curse of $K$ -sample MI estimators

139 In this section, we provide a theoretical analysis of the challenge inherent in learning a  $K$ -sample  
 140 estimator for MI, commonly referred to as the *log-K* curse. Based on this theoretical analysis, we  
 141 give insights to overcome this challenge. Given that we focus on the generalisation of RL, we only  
 142 consider cases with a finite sample size of  $K$ . If a context encoder  $\psi$  in Equation 1 has sufficient  
 143 training epochs, then  $I_{\text{InfoNCE}}(x; y | \psi, K) \approx \log K$  [Mnih and Teh, 2012, Guo et al., 2022]. Hence,  
 144 the MI we can optimise is bottlenecked by the number of available samples, formally expressed as:

**Lemma 1** *Learning a context encoder  $\psi$  with a  $K$ -sample estimator and finite sample size  $K$ , we always have  $I_{\text{InfoNCE}}(x; y | \psi, K) \leq \log K \leq I(x; y)$ , when  $x \neq y$ . (see proof in Appendix A)*

We do not consider the case when  $x \perp\!\!\!\perp y$ , i.e.,  $\log K \geq I(x; y) = 0 (\forall K \geq 1)$ , because a meta-RL agent learns a context encoder by maximising MI between trajectories  $\tau_c$  and context embeddings  $c$ , which are not independent according to the MDP graph in Figure 2. Good compression is crucial for generalisation, and compressing valuable information from a limited number of  $K$  samples requires MI as an effective measure of compression [Goldfeld et al., 2019]. We derive three key insights when learning a context encoder with finite sample size: (1) focus on a ground-truth MI that is smaller than  $I(x; y)$ ; (2) develop a  $K$ -sample estimator tighter than the  $I_{\text{InfoNCE}}$ ; (3) increasing sample quantity  $K$ , however, this is usually impractical. A meta-RL agent learns a context encoder by maximising MI between trajectories  $\tau_c$  and context embeddings  $c$ . Driven

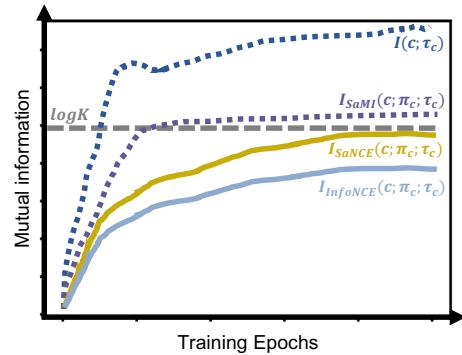


Figure 3:  $I_{\text{InfoNCE}}(c; \pi_c; \tau_c)$ , with a finite sample size of  $K$ , is a loose lower bound of  $I(c; \tau_c)$  and leads to lower performance embeddings.  $I_{\text{SaMI}}(c; \pi_c; \tau_c)$  is a lower ground-truth MI, and  $I_{\text{SaNCE}}(c; \pi_c; \tau_c)$  is a tighter lower bound.

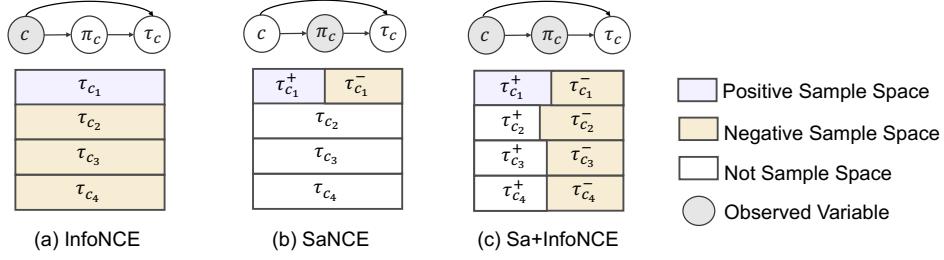


Figure 4: A comparison of sample spaces for task  $e_1$ . Positive samples  $\tau_{c_1}$  or  $\tau_{c_1}^+$  are always from current task  $e_1$ . For SaNCE, in a task  $e_k$  with embedding  $c_k$ , the positive skill  $\pi_{c_k}^+$  conditions on  $c_k$  and generates positive trajectories  $\tau_{c_k}^+$ , and the negative skill  $\pi_{c_k}^-$  generates negative trajectories  $\tau_{c_k}^-$ . The top graphs show the relationship between  $c$ ,  $\pi_c$  and  $\tau_c$ .

148 by insight (1), we introduce Skill-aware Mutual Information (SaMI) in Section 4.2, designed to  
 149 enhance the zero-shot generalisation of downstream RL tasks. Corresponding to insight (2), we  
 150 propose Skill-aware Noise Contrastive Estimation (SaNCE) to maximise SaMI with finite samples in  
 151 Section 4.3. Finally, Section 4.4 demonstrates how to equip a Meta-RL agent with SaNCE in practice.

## 152 4.2 Skill-aware mutual information: a smaller ground-truth MI

153 A useful tool in learning a versatile agent is to understand when to explore novel skills or switch  
 154 between existing skills in multi-task settings. To start with, we define skills [Eysenbach et al., 2018]:

155 **Definition 1 (Skills)** A policy  $\pi$  conditioned on a fixed context embedding  $c$  is defined as a skill  
 156  $\pi(\cdot|c)$ , abbreviated as  $\pi_c$ . If a skill  $\pi_c$  is conditioned on a state  $s_t$ , we can sample actions  $a_t \sim$   
 157  $\pi(\cdot|c, s_t)$ . After sampling actions from  $\pi_c$  at consecutive timesteps, we obtain a trajectory  $\tau_{c,t:t_T} =$   
 158  $\{s_t, a_t, r_t, s_{t+1}, \dots, s_{t+T}, a_{t+T}, r_{t+T}\}$  which demonstrates a consistent mode of behaviour.

159 After interacting with the environment, an agent should infer the task (i.e., environmental features  
 160  $e = e^0, e^1, \dots, e^N$ ) and adapt within an episode. The context encoder  $\psi$  should learn by maximising  
 161 the MI between context embedding  $c$ , skills  $\pi_c$ , and trajectories  $\tau_c$ . We achieve this learning process  
 162 by maximising the MI  $I_{\text{SaMI}}(c; \pi_c; \tau_c)$ , in which we introduce a variable, skill  $\pi_c$ , into  $I(c; \tau_c)$ .  
 163 Formally, we propose a novel MI optimisation objective for a context encoder, **Skill-aware Mutual**  
 164 **Information (SaMI)**, which is defined as:

$$I_{\text{SaMI}}(c; \pi_c; \tau_c) = \mathbb{E}_{p(c, \pi_c, \tau_c)} \left\{ \log \frac{p(c, \pi_c, \tau_c)}{p(c)p(\pi_c)p(\tau_c)} \right\}. \quad (2)$$

165 Although we cannot evaluate  $p(c, \pi_c, \tau_c)$  directly, we approximate it by Monte-Carlo sampling, using  
 166  $K$  samples from  $p(c, \pi_c, \tau_c)$ . A context encoder  $\psi$  trained with the objective of maximising MI  
 167  $I_{\text{SaMI}}(c; \pi_c; \tau_c)$  will converge more quickly because  $I_{\text{SaMI}}(c; \pi_c; \tau_c) \leq I(c; \tau_c)$  (see proof in Appendix  
 168 B).  $I_{\text{SaMI}}$  is an objective that can discriminate between skills, therefore it also enables RL agents to  
 169 autonomously discover diverse skills.

## 170 4.3 Skill-aware noise contrastive estimation: a tighter $K$ -sample estimator

171 Despite InfoNCE’s success as a  $K$ -sample estimator for approximating MI [Laskin et al., 2020,  
 172 Eysenbach et al., 2022], its learning efficiency plunges due to limited numerical precision, which is  
 173 called the log- $K$  curse, i.e.,  $I_{\text{InfoNCE}} \leq \log K \leq I_{\text{SaMI}}$  [Chen et al., 2021] (see proof in Appendix B).  
 174 When  $K \rightarrow +\infty$ , we can expect  $I_{\text{InfoNCE}} \approx \log K \approx I_{\text{SaMI}}$  [Guo et al., 2022]. However, increasing  
 175  $K$  is too expensive, especially in complex environments with enormous negative sample space. In  
 176 response, we propose a novel  $K$ -sample estimator with a reduced required sample space size  $K^*$   
 177 ( $K^* \ll +\infty$ ). First, we define  $K^*$ :

178 **Definition 2 ( $K^*$ )**  $K^* = |c| \cdot |\pi_c| \cdot M$  is defined as the number of trajectories in the replay buffer  
 179 (i.e., the sample space), in which  $|c|$  represents the number of different context embeddings  $c$ ,  $|\pi_c|$   
 180 represents the number of different skills  $\pi_c$ , and  $M$  is a natural number.

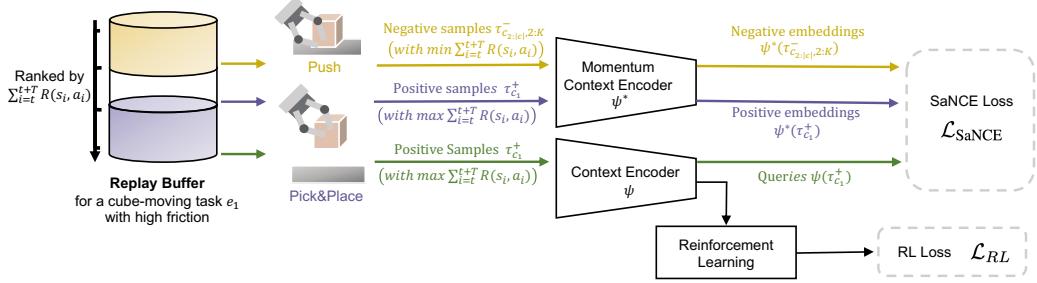


Figure 5: A practical framework for using SaNCE in the meta-training phase. During meta-training, we sample trajectories from the task-specific replay buffer for off-policy training. **Queries** are generated by a context encoder  $\psi$ , which is updated with gradients from both the SaNCE loss  $\mathcal{L}_{\text{SaNCE}}$  and the RL loss  $\mathcal{L}_{\text{RL}}$ . **negative/positive** embeddings are encoded by a momentum context encoder  $\psi^*$ , which is driven by a momentum update with the encoder  $\psi$ . During meta-testing, the meta-trained context encoder  $\psi$  embeds the current trajectory, and the RL policy takes the embedding as input together with the state for adaptation within an episode.

181 Note that  $K^*$  is the maximum batch size that can be sampled in contrastive learning. Therefore,  
182 to ensure that  $I_{\text{InfoNCE}}$  is a tight bound of  $I_{\text{SaMI}}$ , we require that  $I_{\text{InfoNCE}} \approx \log K \approx I_{\text{SaMI}}$  when  
183  $K \rightarrow K^*$ . Under the definition of  $K^*$ , the replay buffer can be divided according to the different  
184 context embeddings  $c$  and skills  $\pi_c$  (i.e., observing context embeddings  $c$  and skills  $\pi_c$ ). In real-world  
185 robotic control tasks, the sample space size significantly increases due to multiple environmental  
186 features  $e = \{e^0, e^1, \dots, e^N\}$ . Taking the sample space of InfoNCE as an example (Figure 4(a)), in  
187 the current task  $e_1$  with context embedding  $c_1$ , positive samples are trajectories  $\tau_{c_1}$  generated after  
188 executing the skill  $\pi_{c_1}$  in task  $e_1$ , and negative samples are trajectories  $\{\tau_{c_2}, \dots\}$  from other tasks  
189  $\{e_2, \dots\}$ . The permutations and combinations of  $N$  environmental features lead to an exponential  
190 growth in task number  $|c|$ , which in turn results in an increase of sample space  $K_{\text{InfoNCE}}^* = |c| \cdot |\pi| \cdot M$ .  
191 We introduce a tight  $K$ -sample estimator, **Skill-aware Noise Contrastive Estimation (SaNCE)**,  
192 which is used to approximate  $I_{\text{SaMI}}(c; \pi_c; \tau_c)$  with a reduced  $K_{\text{SaNCE}}^*$ . For SaNCE, both positive  
193 samples  $\tau_{c_1}^+$  and negative samples  $\tau_{c_1}^-$  are sampled from the current tasks  $e_1$ , but are generated by  
194 executing positive skills  $\pi_{c_1}^+$  and negative skills  $\pi_{c_1}^-$ , respectively. Here, a *positive skill* is intuitively  
195 defined by whether it is optimal for the current task  $e$ , with a more formal definition provided in  
196 Section 4.4. For instance, in a cube-moving task under a large friction setting, the agent executes  
197 a skill  $\pi_c^+$  after several iterations of learning, and obtains corresponding trajectories  $\tau_c^+$  where the  
198 cubes leave the table surface. This indicates that the skill  $\pi_c^+$  is Pick&Place and other skills  $\pi_c^-$  may  
199 include Push or Flip (flipping the cube to the goal position), with corresponding trajectories  $\tau_c^-$  where  
200 the cube remains stationary or rolls on the table. Formally, we can optimise the  $K$ -sample lower  
201 bound  $I_{\text{SaNCE}}$  to approximate  $I_{\text{SaMI}}$ :

$$\begin{aligned} I_{\text{SaNCE}}(c; \pi_c; \tau_c | \psi, K) &= \mathbb{E}_{p(c_1, \pi_{c_1}, \tau_{c_1}^+) p(\tau_{c_1, 2:K}^-)} \left[ \log \left( \frac{K \cdot f_\psi(c_1, \pi_{c_1}, \tau_{c_1}^+)}{f_\psi(c_1, \pi_{c_1}, \tau_{c_1}^+) + \sum_{k=2}^K f_\psi(c_1, \pi_{c_1}, \tau_{c_1, k}^-)} \right) \right] \\ &\leq I_{\text{SaMI}}(c; \pi_c; \tau_c) \end{aligned} \quad (3)$$

202 where  $f_\psi(c_1, \pi_{c_1}, \tau_{c_1}) = e^{\psi(\tau_{c_1})^\top \cdot \psi^*(\tau_{c_1}) / \beta}$ . The **query**  $c_1 = \psi(\tau_{c_1})$  is generated by the context  
203 encoder  $\psi$ . For training stability, we use a momentum encoder  $\psi^*$  to produce the **positive** and  
204 **negative** embeddings. SaNCE significantly reduces the required sample space size  $K_{\text{SaNCE}}^*$  by  
205 sampling trajectories  $\tau_c$  based on different skills  $\pi_c$  (Figure 4(b)) in task  $e_1$ , so that  $K_{\text{SaNCE}}^* =$   
206  $|c| \cdot |\pi_c| \cdot M = |\pi_{c_1}| \cdot M \leq K_{\text{InfoNCE}}^* (|c| = |c_1| = 1)$ . Therefore,  $I_{\text{SaNCE}}$  satisfies Lemma 2:

207 **Lemma 2** *With a context encoder  $\psi$  and finite sample size  $K$ , we have  $I_{\text{InfoNCE}}(c; \pi_c; \tau_c | \psi, K) \leq$   
208  $I_{\text{SaNCE}}(c; \pi_c; \tau_c | \psi, K) \leq \log K \leq I_{\text{SaMI}}(c; \pi_c; \tau_c) \leq I(c; \tau_c)$ . (see proof in Appendix B)*

209 SaNCE is a plug-and-play module for any other NCE. For example, the negative sample space can  
 210 be more diverse by combining SaNCE and InfoNCE with  $K_{\text{Sa+InfoNCE}}^* = \left( \sum_{i=1}^{|c|} |\pi_{c_i}^-| + |\pi_{c_i}^+| \right) \cdot M$   
 211 (Figure 4(c)). A detailed analysis of the sample size of  $I_{\text{Sa+InfoNCE}}$  can be found in Appendix C.

#### 212 4.4 Skill-aware trajectory sampling strategy

213 Methods that focus on skill diversity often rely heavily on accurately defining and identifying skills  
 214 [Eysenbach et al., 2018], with some requiring a prior skill distribution that is often inaccessible  
 215 [Shi et al., 2022]. This variability in skill definitions across tasks can affect the robustness and  
 216 generalisability of these methods. For these reasons, our approach does not use such skill definitions  
 217 and priors for specific environments or tasks. In this section, we give the distinctiveness of skills and  
 218 propose a practical trajectory sampling method.

219 In this study, we believe that distinctiveness of skills is inherently difficult to achieve — a slight  
 220 difference in states can make two skills distinguishable, and not necessarily in a semantically  
 221 meaningful way. Instead, we should focus on whether skills acquired by an agent can complete a  
 222 task. For example, in high-friction tasks, the agent must acquire the Pick&Place skill to avoid large  
 223 frictional forces, while in high-mass tasks, the agent must learn the Push skill since it cannot lift the  
 224 cube. In that way, without skills definition in a semantically meaningful way, we only need to train an  
 225 agent on a set of tasks, and it will autonomously discover diverse skills to work across multiple tasks.

226 We consider the task-specific definition of positive/negative samples. In a given task  $e$ , *positive skills*  
 227  $\pi_c^+$  are defined as the optimal skills achieving the highest return  $\sum_{i=t}^{t+T} R(s_i, a_i)$ , while *negative*  
 228 *skills*  $\pi_c^-$  are those with lower returns. Thus, we can simply sample the trajectory with the ranked  
 229 highest return as the positive sample  $\tau_c^+$ , and the one with the lowest return as the negative sample  
 230  $\tau_c^-$ . Positive samples are generated by the optimal skill for the current task, while lower return  
 231 samples are classified as negative. This polarised definition helps the model select the optimal skill  
 232 from among many skills with varying returns and avoids the issue of hard negative examples during  
 233 training Robinson et al. [2021]. The SaNCE loss is then minimised to bring the context embeddings  
 234 of the highest return trajectories closer in the embedding space while distancing those of negative  
 235 trajectories. Note that, at the end of the training, the top-ranked trajectories in the ranked replay buffer  
 236 correspond to positive samples  $\tau_c^+$  with high returns, and the lower-ranked ones are negative samples  
 237  $\tau_c^-$  with low returns. However, before the agent is able to achieve high returns, all trajectories are  
 238 with low returns. Therefore, our SaNCE loss is a soft version of the  $K$ -sample SaNCE estimator:

$$\mathcal{L}_{\text{SaNCE}} = - \max \left( \|\psi(\tau_c^+), \psi(\tau_c^-)\|_{L2}, 1 \right) \cdot I_{\text{SaNCE}} \quad (4)$$

239 where  $\|\cdot\|_{L2}$  represents the Euclidean distance [Tabak, 2014]. Figure 5 provides a practical framework  
 240 of SaNCE, with a cube-moving example task  $e_1$  under high friction. In task  $e_1$ , the positive skill  $\pi_{c_1}^+$   
 241 is the *Pick&Place* skill, which is used to generate *queries*  $\psi(\tau_{c_1}^+)$  and *positive embeddings*  $\psi^*(\tau_{c_1}^+)$ ;  
 242 after executing *Push* skill we get *negative samples*  $\tau_{c_1}^-$  and *negative embeddings*  $\psi^*(\tau_{c_1}^-)$ .

## 243 5 Experiments

244 We use a three-step process to demonstrate the benefits of SaMI in each environment and answer  
 245 three questions: (1) Does optimising SaMI lead to increased returns during training and zero-shot  
 246 generalisation (see Table 1 and 2)?; (2) Does SaMI help the RL agents to be versatile and embody  
 247 multiple skills (see Figure 6)?; (3) Can SaNCE overcome the log- $K$  curse in sample-limited scenarios  
 248 (see Table 1 and 2, and Section 5.4)?

### 249 5.1 Experimental setup

250 **Modified benchmarks with multiple environmental features.**<sup>1</sup> We demonstrate the efficacy of  
 251 our method using two benchmarks, Panda-gym [Gallouédec et al., 2021] and MuJoCo [Todorov  
 252 et al., 2012] (details in Sections 5.2 and 5.3). The benchmarks are modified to be influenced by  
 253 multiple environmental features simultaneously. Environmental features are sampled at the start  
 254 of each episode during both the meta-training and meta-testing phases. During meta-training, we

<sup>1</sup>Our modified benchmarks are open-sourced at [Anonymous Link](#)

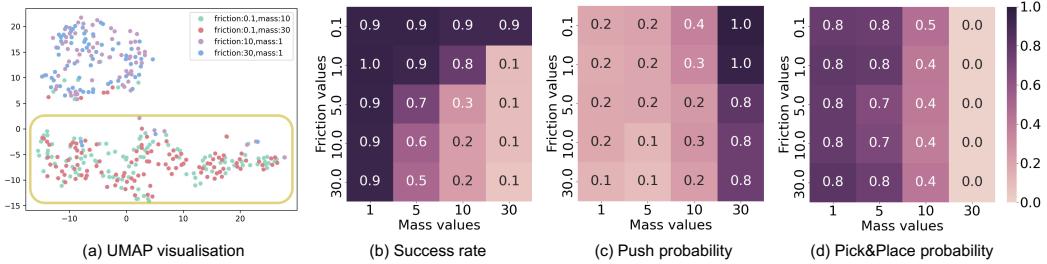


Figure 6: (a) UMAP visualisation of context embeddings for the SaCCM in the Panda-gym environment. The points in the yellow box correspond to the Push skill executed in high-mass tasks. Heatmap of (b) success rate, and probability of learned (c) Push skill and (d) Pick&Place skill of SaCCM. Under large mass scenarios, the probability of executing Push skill is higher than Pick&Place skill.

uniform-randomly select a combination of environmental features from a training task set. At test time, we evaluate each algorithm in unseen tasks with different environmental features outside the training range. Generalisation performance is measured in two different regimes: moderate and extreme. The moderate regime draws environmental features from a closer range to the training range compared to the extreme. For all our experiments, we report the mean and standard deviation of the models trained over five seeds in both training and test tasks. Further experimental details are available in Appendix D.

**Baselines.** In our experiments, we primarily compare InfoNCE with SaNCE to demonstrate the performance improvements brought by SaNCE. Therefore, we compare our method with three prevailing and competitive baselines. First, we consider CCM [Fu et al., 2021], which is equipped with InfoNCE. Additionally, we consider TESAC [Yu et al., 2020], which employs a value function loss, allowing us to evaluate the impact on the context encoder without contrastive loss. Given that CCM and TESAC are using RNN encoder, we also consider PEARL [Rakelly et al., 2019], which utilises an MLP context encoder and a similar loss as TESAC. In Appendix G, we also provide a comparison with DOMINO [Mu et al., 2022] and CaDM [Lee et al., 2020] using exactly the same environmental setting in MuJoCo benchmark.

**Our methods.**<sup>2</sup> We employ Soft Actor-Critic (SAC) [Haarnoja et al., 2018] as the base RL algorithm and trained agents for 1.6 million timesteps in each environment (please refer to Appendix D.3 for more implementation details). SaNCE is a simple objective based on mutual information that can be used to train any context encoder. Two RL algorithms are equipped with SaNCE: (1) SaTESAC is TESAC with SaNCE, which uses SaNCE for contrastive learning, with a  $|c|$  times smaller sample space than that of other algorithms, as shown in Figure 4(b); (2) SaCCM is CCM with SaNCE, where the contrastive learning combines InfoNCE and SaNCE, as shown in Figure 4(c).

## 5.2 Panda-gym

**Task description.** Our modified Panda-gym benchmark contains a robot arm control task using the Franka Emika Panda [Gallouédec et al., 2021], where the robot needs to move a cube to a target position. Unlike previous works, we simultaneously modify multiple environmental features (cube mass and table friction) that characterise the transition dynamics, and the robot can flexibly execute different skills (Push and Pick&Place) for different tasks. This environment demands high skill diversity from the agent. For example, in high-friction tasks, the agent must use the Pick&Place skill, while in high-mass tasks, it must use the Push skill.

Table 1: Comparison of success rate with baselines in Panda-gym (over 5 seeds). **Bold text** signifies the highest average return. \* next to the number means that the algorithm with SaMI has statistically significant improvement over the same algorithm without SaMI. All significance claims based on paired t-tests with significance threshold of  $p < 0.05$ .

	Training	Test (moderate)	Test (extreme)
PEARL	$0.42 \pm 0.19$	$0.10 \pm 0.06$	$0.11 \pm 0.05$
TESAC	$0.50 \pm 0.22$	$0.31 \pm 0.20$	$0.22 \pm 0.21$
CCM	$0.80 \pm 0.19$	$0.49 \pm 0.23$	$0.29 \pm 0.28$
SaTESAC	<b><math>0.92 \pm 0.04^*</math></b>	<b><math>0.56 \pm 0.24^*</math></b>	<b><math>0.37 \pm 0.34^*</math></b>
SaCCM	<b><math>0.93 \pm 0.05^*</math></b>	<b><math>0.57 \pm 0.26^*</math></b>	<b><math>0.36 \pm 0.35^*</math></b>

<sup>2</sup>Our code, video demos and experimental data are available at <https://anonymous.4open.science/r/SaMI>

Table 2: Comparison of average return  $\pm$  standard deviation with baselines in modified MuJoCo benchmark (over 5 seeds). **Bold number** signifies the highest return. \* next to the number means that the algorithm with SaMI has statistically significant improvement over the same algorithm without SaMI. All significance claims based on t-tests with significance threshold of  $p < 0.05$ .

Crippled Ant			Crippled Half-cheetah		
Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
PEARL	1682 $\pm$ 73	996 $\pm$ 21	888 $\pm$ 31	1998 $\pm$ 973	698 $\pm$ 548
TESAC	2139 $\pm$ 90	1952 $\pm$ 40	1048 $\pm$ 124	3967 $\pm$ 955	874 $\pm$ 901
CCM	2361 $\pm$ 114	2047 $\pm$ 83	1527 $\pm$ 301	3481 $\pm$ 488	821 $\pm$ 575
SaTESAC	2638 $\pm$ 406	2379 $\pm$ 528	<b>2131<math>\pm</math>132*</b>	4328 $\pm$ 1092	<b>1143<math>\pm</math>664*</b>
SaCCM	2355 $\pm$ 170	2310 $\pm$ 314	2007 $\pm$ 68*	<b>4478<math>\pm</math>1131*</b>	1007 $\pm$ 568
Ant			Half-cheetah		
Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
PEARL	5153 $\pm$ 581	3873 $\pm$ 235	3802 $\pm$ 409	5802 $\pm$ 773	2190 $\pm$ 970
TESAC	6789 $\pm$ 451	4705 $\pm$ 279	4108 $\pm$ 369	6298 $\pm$ 2310	3173 $\pm$ 1210
CCM	6901 $\pm$ 567	5179 $\pm$ 902	4700 $\pm$ 696	6955 $\pm$ 788	3963 $\pm$ 622
SaTESAC	7314 $\pm$ 545	5513 $\pm$ 648*	4940 $\pm$ 531*	7430 $\pm$ 1026	4058 $\pm$ 890
SaCCM	7478 $\pm$ 539	5717 $\pm$ 488	5215 $\pm$ 377	7154 $\pm$ 965	3849 $\pm$ 689
SlimHumanoid			HumanoidStandup		
Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
PEARL	6947 $\pm$ 3541	3697 $\pm$ 2674	2018 $\pm$ 907	95456 $\pm$ 13445	63242 $\pm$ 13546
TESAC	8437 $\pm$ 1798	6989 $\pm$ 1301	3760 $\pm$ 308	158384 $\pm$ 14455	153944 $\pm$ 15046
CCM	7696 $\pm$ 1907	5784 $\pm$ 531	2887 $\pm$ 1058	146480 $\pm$ 33745	154601 $\pm$ 16291
SaTESAC	<b>10216<math>\pm</math>1620</b>	<b>7886<math>\pm</math>2203</b>	6123 $\pm$ 1403*	178142 $\pm$ 10081*	168337 $\pm$ 12123
SaCCM	9312 $\pm$ 705	7430 $\pm$ 1587	<b>6473<math>\pm</math>2001*</b>	<b>187930<math>\pm</math>19338*</b>	<b>181033<math>\pm</math>14628</b>
Hopper			Crippled Hopper		
Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
PEARL	934 $\pm$ 242	874 $\pm$ 366	799 $\pm$ 298	3091 $\pm$ 298	2387 $\pm$ 656
TESAC	1492 $\pm$ 59	1499 $\pm$ 35	1459 $\pm$ 72	3575 $\pm$ 192	3298 $\pm$ 551
CCM	1484 $\pm$ 54	1446 $\pm$ 64	1452 $\pm$ 58	3455 $\pm$ 301	3409 $\pm$ 239
SaTESAC	1502 $\pm$ 20	1453 $\pm$ 39	1447 $\pm$ 14	3391 $\pm$ 84	3262 $\pm$ 166
SaCCM	1462 $\pm$ 45	1462 $\pm$ 14	1451 $\pm$ 67	3449 $\pm$ 103	3390 $\pm$ 211
Walker			Crippled Walker		
Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
PEARL	7524 $\pm$ 2455	3355 $\pm$ 2555	1984 $\pm$ 356	7899 $\pm$ 2532	4377 $\pm$ 2563
TESAC	7747 $\pm$ 1772	4355 $\pm$ 1530	2581 $\pm$ 407	9908 $\pm$ 1561	5929 $\pm$ 1971
CCM	8136 $\pm$ 557	5476 $\pm$ 803	2519 $\pm$ 682	10317 $\pm$ 1137	6233 $\pm$ 1869
SaTESAC	8675 $\pm$ 752	5840 $\pm$ 676	<b>3632<math>\pm</math>404*</b>	10389 $\pm$ 1031	<b>8387<math>\pm</math>1291*</b>
SaCCM	8361 $\pm$ 586	5779 $\pm$ 691	3481 $\pm$ 332*	10496 $\pm$ 951	8235 $\pm$ 1212

279 **Results and skill analysis.** As shown in Table 1, SaTESAC and SaCCM achieve superior generali-  
280 sation performance compared to PEARL, TESAC, and CCM, with a smaller sample space. Faced  
281 with an unknown task, the agent achieved the three steps of "explore effectively, infer, adapt," and  
282 acquired multiple skills (Push, Pick&Place) to work across various tasks (please refer to the video  
283 demos<sup>2</sup>). We used UMAP [McInnes et al., 2020] and t-SNE [Van der Maaten and Hinton, 2008]  
284 to visualise the context embedding. We plotted the context embedding at the final time step in 100  
285 tests for each task, which can compress how a skill alters the state of the environment in a consistent  
286 way. Additionally, we determined the skills by detecting contact points between the end effector and  
287 the cube, and between the cube and the table (see Appendix D for more details), and then employed  
288 heatmaps [Waskom, 2021] to visualise executed skills. As shown in Figure 6, when the cube mass is  
289 large (30 Kg and 10 Kg), the agent learned the Push skill (corresponding to the clustered points in the  
290 yellow bounding box in Figure 6(a)). With smaller masses, the agent learned the Pick&Place skill.  
291 However, as shown in Figure 15 in Appendix F.1, CCM did not exhibit clear skill grouping. Based on  
292 the t-test results in Table 1, SaMI significantly improved the success rate in the training, moderate test,  
293 and extreme test sets at a significance level of 0.05. Overall, SaMI helps to compress high-quality  
294 skill-related information from the trajectories and helps to acquire diverse skills autonomously. More  
295 visualisation results can be found in Appendix F.

296 **5.3 MuJoCo**

297 **Task description.** We extended the modified MuJoCo benchmark introduced in DOMINO [Mu  
298 et al., 2022] and CaDM [Lee et al., 2020]. It contains ten typical robotic control environments  
299 based on the MuJoCo physics engine [Todorov et al., 2012]. Hopper, Walker, Half-cheetah, Ant,  
300 HumanoidStandup, and SlimHumanoid are influenced by continuous environmental features (i.e.,  
301 mass, damping) that affect transition dynamics. Crippled Ant, Crippled Hopper, Crippled Walker, and  
302 Crippled Half-cheetah are more challenging due to the addition of discrete environmental features  
303 (i.e., randomly crippled leg joints), requiring agents to master different skills (e.g., switching from  
304 running to crawling after a leg is crippled).

**Results and skill analysis.** Table 2 shows the average return of our method and baselines on both training and test tasks. SaTESAC and SaCCM gained higher returns in both training and testing in most of the tasks, in which the Ant, Half-Cheetah and Hopper tasks are exceptions. Ant and Hopper robots only need to learn a single skill to generalise across different tasks. For example, the Hopper robot learned to hop forward on the floor to adapt to different mass values. When environments (Crippled Ant, Crippled Hopper, Crippled Half-Cheetah, SlimHumanoid, HumanoidStandup, and Crippled Walker) become complex and require diverse skills for different tasks, SaNCE brings significant improvements. For instance, when the Ant robot has 3 or 4 legs available, it learns to roll to work across varying mass and damping. However, during zero-shot generalisation, when only 2 legs are available, the ant robot can no longer roll and it adapts by walking using its 2 healthy legs. This aligns with the result of the t-test [Rice and Rice, 2007] in Table 2, which SaMI brings significant improvement on the extreme test set (more details refer to Appendix H). Therefore, i) SaMI helps the RL agents to be versatile and embody multiple skills; ii) SaMI leads to increased returns during training and zero-shot generalisation, especially in environments that require different skills. Please refer to our video demos<sup>2</sup> for different skills in all environments, and visualisation results in Appendix F.2.

#### 5.4 Analysis of the log- $K$ curse in sample-limited scenarios

In this section, we further analyse whether SaNCE can overcome the log- $K$  curse. During the training phases, we sample the environmental features at the beginning of each episode. Therefore, throughout the training process, the context encoder needs to learn the context embedding distribution of multiple tasks. Because InfoNCE requires sampling negative samples across all tasks, and SaNCE samples negative samples from the current task, SaNCE’s negative sample space is  $|c|$  times smaller than that of InfoNCE. For example, where both mass and damping have five values, SlimHumanoid environment has a maximum of 25 different tasks, making the sampling space of InfoNCE potentially 25 times larger than that of SaNCE. According to Table 1 and 2, RL algorithms equipped with SaNCE can achieve better or comparable performance with a much smaller number of negative samples  $K$  than InfoNCE. This indicates that SaNCE indeed plays a significant role in addressing the log- $K$  curse, and the SaMI objective helps the contrastive context encoder extract information that is crucial for downstream RL tasks.

Furthermore, the number of negative samples  $K$  also relates to two hyperparameters: **buffer size**, which directly determines the size of the negative sample space; and **contrastive batch size**, which determines the number of samples used for training contrastive context encoder at each update step. Therefore, we conducted further analysis on these two hyperparameters. As shown in Figure 7, we find that reductions in buffer size and contrastive batch size do not significantly decrease the average return for SaCCM and SaTESAC, which exhibit state-of-the-art performance with small buffers and contrastive batch sizes. Note that the results in Table 2 correspond to a buffer size of 100,000 and

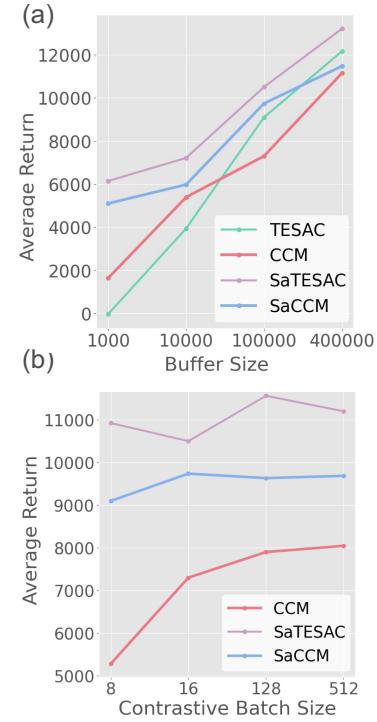


Figure 7: Changes of average return in the SlimHumanoid environment. (a) The impact of buffer size on RL control performance (i.e., TESAC, CCM, SaTESAC, SaCCM). (b) The impact of contrastive batch size on the contrastive context encoder (i.e., CCM, SaTESAC, SaCCM).

320 a contrastive batch size of 12. Experimental results for all environments (Appendix E.2) further  
321 demonstrate that SaNCE is not highly sensitive to changes in  $K$ , and indeed shows great potential in  
322 overcoming the log- $K$  curse.

## 323 **6 Conclusion and future work**

324 In this paper, we propose a Skill-aware Mutual Information (SaMI) to learn context embeddings for  
325 zero-shot generalisation in downstream RL tasks, and a Skill-aware Noise Contrastive Estimation  
326 (SaNCE) to optimise SaMI and overcome the log- $K$  curse. RL algorithms equipped with SaMI have  
327 achieved state-of-the-art performance in the MuJoCo and Panda-gym benchmarks. Through skill  
328 analysis and video demos in Panda-gym and MuJoCo, we confirm that the context encoder, learned  
329 by maximising SaMI, can compress high-quality skill-related information from trajectories, thereby  
330 assisting the RL agent acquire diverse skills autonomously and zero-shot generalising across various  
331 tasks. Notably, the optimisation process of SaNCE utilises a far smaller negative sample space than  
332 baselines. Coupled with further experimental analysis of buffer size and contrastive batch size on  
333 MuJoCo, we demonstrate that SaNCE helps overcome the log- $K$  curse. Our results indicate the  
334 importance of aligning the optimisation objectives of representation learning and downstream optimal  
335 control to enhance task performance and improve data efficiency.

336 Given that environmental features are often interdependent, such as a cube’s material correlating  
337 with friction and mass, SaMI does not introduce independence assumptions like DOMINO [Mu  
338 et al., 2022]. Therefore, future work will focus on verifying and enhancing SaMI’s potential in more  
339 complex tasks where environmental features are correlated. This will contribute to our ultimate  
340 goal: developing a generalist and versatile agent capable of working across multiple tasks and even  
341 real-world tasks in the near future.

## 342 **References**

- 343 Rishabh Agarwal, Marlos C Machado, Pablo Samuel Castro, and Marc G Bellemare. Contrastive  
344 behavioral similarity embeddings for generalization in reinforcement learning. *arXiv preprint*  
345 *arXiv:2101.05265*, 2021.
- 346 Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saun-  
347 shi. A theoretical analysis of contrastive unsupervised representation learning. *arXiv preprint*  
348 *arXiv:1902.09229*, 2019.
- 349 Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing  
350 mutual information across views. *Advances in neural information processing systems*, 32, 2019.
- 351 Junya Chen, Zhe Gan, Xuan Li, Qing Guo, Liqun Chen, Shuyang Gao, Tagyoung Chung, Yi Xu,  
352 Belinda Zeng, Wenlian Lu, et al. Simpler, faster, stronger: Breaking the log-k curse on contrastive  
353 learners with flatnce. *arXiv preprint arXiv:2107.01152*, 2021.
- 354 Ignasi Clavera, Anusha Nagabandi, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and  
355 Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement  
356 learning. In *International Conference on Learning Representations*, 2019a.
- 357 Ignasi Clavera, Anusha Nagabandi, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and  
358 Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement  
359 learning. In *International Conference on Learning Representations*, 2019b.
- 360 Remi Tachet des Combes, Philip Bachman, and Harm van Seijen. Learning invariances for policy  
361 generalization, 2018. URL <https://openreview.net/forum?id=BJHRaK1PG>.
- 362 Mhairi Dunion and Stefano V Albrecht. Multi-view disentanglement for reinforcement learning with  
363 multiple cameras. In *Reinforcement Learning Conference*, 2024.
- 364 Mhairi Dunion, Trevor McInroe, Kevin Sebastian Luck, Josiah P. Hanna, and Stefano V Albrecht.  
365 Conditional mutual information for disentangled representations in reinforcement learning. In  
366 *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a.

- 367 Mhairi Dunion, Trevor McInroe, Kevin Sebastian Luck, Josiah P. Hanna, and Stefano V Albrecht.  
368 Temporal disentanglement of representations for improved generalisation in reinforcement learning.  
369 In *The Eleventh International Conference on Learning Representations*, 2023b.
- 370 Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need:  
371 Learning skills without a reward function, 2018. URL <https://arxiv.org/abs/1802.06070>.
- 372 Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Russ R Salakhutdinov. Contrastive learning  
373 as goal-conditioned reinforcement learning. *Advances in Neural Information Processing Systems*,  
374 35:35603–35620, 2022.
- 375 Jörg K.H. Franke, Gregor Koehler, André Biedenkapp, and Frank Hutter. Sample-efficient automated  
376 deep reinforcement learning. In *International Conference on Learning Representations*, 2021.
- 377 Haotian Fu, Hongyao Tang, Jianye Hao, Chen Chen, Xidong Feng, Dong Li, and Wulong Liu.  
378 Towards effective context for meta-reinforcement learning: an approach based on contrastive  
379 learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7457–7465,  
380 2021.
- 381 Quentin Gallouédec, Nicolas Cazin, Emmanuel Dellandréa, and Liming Chen. panda-gym: Open-  
382 Source Goal-Conditioned Environments for Robotic Learning. *4th Robot Learning Workshop:  
383 Self-Supervised and Lifelong Learning at NeurIPS*, 2021.
- 384 Samuel Garcin, James Doran, Shangmin Guo, Christopher G. Lucas, and Stefano V. Albrecht.  
385 DRED: Zero-shot transfer in reinforcement learning via data-regularised environment design. In  
386 *International Conference on Machine Learning (ICML)*, 2024.
- 387 Ziv Goldfeld, Ewout van den Berg, Kristjan Greenewald, Igor Melnyk, Nam Nguyen, Brian Kingsbury,  
388 and Yury Polyanskiy. Estimating information flow in deep neural networks, 2019. URL <https://arxiv.org/abs/1810.05728>.
- 390 Qing Guo, Junya Chen, Dong Wang, Yuewei Yang, Xinwei Deng, Jing Huang, Larry Carin, Fan  
391 Li, and Chenyang Tao. Tight mutual information estimation with contrastive fenchel-legendre  
392 optimization. *Advances in Neural Information Processing Systems*, 35:28319–28334, 2022.
- 393 Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle  
394 for unnormalized statistical models. In *Proceedings of the thirteenth international conference on  
395 artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings,  
396 2010.
- 397 Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy  
398 maximum entropy deep reinforcement learning with a stochastic actor. In *International conference  
399 on machine learning*, pages 1861–1870. PMLR, 2018.
- 400 Karol Hausman, Jost Tobias Springenberg, Ziyu Wang, Nicolas Heess, and Martin Riedmiller.  
401 Learning an embedding space for transferable robot skills. In *International Conference on  
402 Learning Representations*, 2018.
- 403 Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for  
404 unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on  
405 computer vision and pattern recognition*, pages 9729–9738, 2020.
- 406 R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam  
407 Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation  
408 and maximization. In *International Conference on Learning Representations*, 2019.
- 409 Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David  
410 Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks, 2016.
- 411 Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations  
412 for reinforcement learning. In *International conference on machine learning*, pages 5639–5650.  
413 PMLR, 2020.

- 414 Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware dynamics  
415 model for generalization in model-based reinforcement learning. In *International Conference on*  
416 *Machine Learning*, pages 5757–5766. PMLR, 2020.
- 417 Lanqing Li, Yuanhao Huang, Mingzhe Chen, Siteng Luo, Dijun Luo, and Junzhou Huang. Provably  
418 improved context-based offline meta-rl with attention and contrastive learning. *arXiv preprint*  
419 *arXiv:2102.10774*, 2021.
- 420 William McGill. Multivariate information transmission. *Transactions of the IRE Professional Group*  
421 *on Information Theory*, 4(4):93–111, 1954.
- 422 Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and  
423 projection for dimension reduction, 2020. URL <https://arxiv.org/abs/1802.03426>.
- 424 Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic  
425 language models. *arXiv preprint arXiv:1206.6426*, 2012.
- 426 Yao Mu, Yuzheng Zhuang, Fei Ni, Bin Wang, Jianyu Chen, Jianye HAO, and Ping Luo. DOMINO:  
427 Decomposed mutual information optimization for generalized context in meta-reinforcement  
428 learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors,  
429 *Advances in Neural Information Processing Systems*, 2022.
- 430 Kento Nozawa and Issei Sato. Understanding negative samples in instance discriminative self-  
431 supervised representation learning. *Advances in Neural Information Processing Systems*, 34:  
432 5784–5797, 2021.
- 433 Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive  
434 coding. *arXiv preprint arXiv:1807.03748*, 2019.
- 435 Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational  
436 bounds of mutual information. In *International Conference on Machine Learning*, pages 5171–  
437 5180. PMLR, 2019.
- 438 Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John  
439 Wiley & Sons, 2014.
- 440 Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah  
441 Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine*  
442 *Learning Research*, 22(268):1–8, 2021. URL <http://jmlr.org/papers/v22/20-1364.html>.
- 443 Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy  
444 meta-reinforcement learning via probabilistic context variables. In *International conference on*  
445 *machine learning*, pages 5331–5340. PMLR, 2019.
- 446 John A Rice and John A Rice. *Mathematical statistics and data analysis*, volume 371. Thom-  
447 son/Brooks/Cole Belmont, CA, 2007.
- 448 Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with  
449 hard negative samples, 2021. URL <https://arxiv.org/abs/2010.04592>.
- 450 Tong Sang, Hongyao Tang, Yi Ma, Jianye Hao, Yan Zheng, Zhaopeng Meng, Boyan Li, and Zhen  
451 Wang. Pandr: Fast adaptation to new environments from offline experiences via decoupling policy  
452 and environment representations, 2022.
- 453 Younggyo Seo, Kimin Lee, Ignasi Clavera Gilaberte, Thanard Kurutach, Jinwoo Shin, and Pieter  
454 Abbeel. Trajectory-wise multiple choice learning for dynamics generalization in reinforcement  
455 learning. *Advances in Neural Information Processing Systems*, 33:12968–12979, 2020.
- 456 Lucy Xiaoyang Shi, Joseph J. Lim, and Youngwoon Lee. Skill-based model-based reinforcement  
457 learning. In *Conference on Robot Learning*, 2022.
- 458 Jiaming Song and Stefano Ermon. Understanding the limitations of variational mutual information  
459 estimators. In *International Conference on Learning Representations*, 2020.
- 460 John Tabak. *Geometry: the language of space and form*. Infobase Publishing, 2014.

- 461 Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015*  
462 *ieee information theory workshop (itw)*, pages 1–5. IEEE, 2015.
- 463 Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control.  
464 In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033.  
465 IEEE, 2012.
- 466 Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine*  
467 *learning research*, 9(11), 2008.
- 468 Bernie Wang, Simon Xu, Kurt Keutzer, Yang Gao, and Bichen Wu. Improving context-based  
469 meta-reinforcement learning with self-supervised trajectory contrastive learning, 2021.
- 470 Michael L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):  
471 3021, 2021. doi: 10.21105/joss.03021. URL <https://doi.org/10.21105/joss.03021>.
- 472 Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-  
473 parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision*  
474 *and pattern recognition*, pages 3733–3742, 2018.
- 475 Raymond W Yeung. A new outlook on shannon’s information measures. *IEEE transactions on*  
476 *information theory*, 37(3):466–474, 1991.
- 477 Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey  
478 Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning.  
479 In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.
- 480 Wenxuan Zhou, Lerrel Pinto, and Abhinav Gupta. Environment probing interaction policies. In  
481 *International Conference on Learning Representations*, 2019.

482 **A Proof of Lemma 1**

483 Given a query  $x$  and a set  $Y = \{y_1, \dots, y_K\}$  of  $K$  random samples containing one positive sample  $y_1$   
 484 and  $K - 1$  negative samples from the distribution  $p(y)$ , A  $K$ -sample InfoNCE estimator is obtained  
 485 by comparing pairs sampled from the joint distribution  $x, y_1 \sim p(x, y)$  to pairs  $x, y_k$  built using a set  
 486 of negative examples  $y_{2:K}$ . InfoNCE is obtained by comparing positive pairs  $(x, y_1)$  and negative  
 487 pairs  $(x, y_k)$ , where  $y_k \sim y_{2:K}$ , as follows:

$$I_{\text{InfoNCE}}(x; y|\psi, K) = \mathbb{E}_{p(x, y_1)p(y_{2:K})} \left[ \log \left( \frac{f_\psi(x, y_1)}{\frac{1}{K} \sum_{k=1}^K f_\psi(x, y_k)} \right) \right] \quad (5)$$

488 **Step 1.** Let us prove the  $K$ -sample InfoNCE estimator is upper bounded by  $\log K$ . According to Mu  
 489 et al. [2022],  $\frac{f_\psi(x, y_1)}{\sum_{k=1}^K f_\psi(x, y_k)} = \frac{f_\psi(x, y_1)}{f_\psi(x, y_1) + \sum_{k=2}^K f_\psi(x, y_k)} \leq 1$ . So we have:

$$\begin{aligned} I_{\text{InfoNCE}}(x; y|\psi, K) &= \mathbb{E}_{p(x, y_1)p(y_{2:K})} \left[ \log \left( \frac{f_\psi(x, y_1)}{\frac{1}{K} \sum_{k=1}^K f_\psi(x, y_k)} \right) \right] \\ &= \mathbb{E}_{p(x, y)} \left[ \mathbb{E}_{p(y_{2:K})} \log \left( \frac{K \cdot f_\psi(x, y_1)}{\sum_{k=1}^K f_\psi(x, y_k)} \right) \right] \\ &\leq \log K \end{aligned} \quad (6)$$

490 Hence, we have  $I_{\text{InfoNCE}}(x; y|\psi, K) \leq \log K$ .

491 **Step 2.** We have the  $I(x; y) \geq I_{\text{InfoNCE}}(x; y|\psi, K)$  which are rightmost and the leftmost side in  
 492 Lemma 1 according to:

**Proposition 1** [Poole et al., 2019] A  $K$ -sample estimator is an asymptotically tight lower bound to  
 the MI, i.e.,

$$I(x; y) \geq I_{\text{InfoNCE}}(x; y|\psi, K), \lim_{x \rightarrow +\infty} I_{\text{InfoNCE}}(x; y|\psi, K) \rightarrow I(x; y)$$

493

494 *Proof.* See Poole et al. [2019] for a neat proof of how the multi-sample estimator (e.g., InfoNCE)  
 495 lower bounds MI.

496 **Step 3.** In this research, the context encoder  $\psi$  in  $f_\psi(x, y)$  is implemented using an RNN to  
 497 approximate  $\frac{p(y|x)}{p(y)}$  [Oord et al., 2019]. For most deep learning platforms, with a powerful learner for  
 498  $\psi$  and a finite size  $K$  such that  $I(x; y) \geq \log K$ , we can reasonably expect  $I_{\text{InfoNCE}} \approx \log K$  after a  
 499 few training epochs. Therefore, during training, when  $K \ll +\infty$ , we always have  $I(x; y) \geq \log K$ .

500 *Proof.* See Chen et al. [2021] for more detailed proof.

501 **Step 4.** Let us prove that the  $K$ -sample InfoNCE bound is asymptotically tight. The specific  
 502 choice of context encoder  $\psi$  relates to the tightness of the  $K$ -sample NCE bound. InfoNCE [Oord  
 503 et al., 2019] sets  $f_\psi(x, y) \propto \frac{p(y|x)}{p(y)}$  to model a density ratio which preserves the MI between  $x$   
 504 and  $y$ , where  $\propto$  stands for ‘proportional to’ (i.e. up to a multiplicative constant). Let us plug in

505  $f_\psi(x, y) = f_\psi^*(x, y) = \frac{p(y|x)}{p(y)}$  into InfoNCE, and we have

$$\begin{aligned}
I_{\text{InfoNCE}}(x; y|\psi, K) &= \mathbb{E} \left[ \log \left( \frac{f_\psi^*(x, y_1)}{\sum_{k=1}^K f_\psi^*(x, y_k)} \right) \right] + \log K \\
&= -\mathbb{E} \left[ \log \left( 1 + \frac{p(y)}{p(y|x)} \sum_{k=2}^K \frac{p(y_k|x)}{p(y_k)} \right) \right] + \log K \\
&\approx -\mathbb{E} \left[ \log \left( 1 + \frac{p(y)}{p(y|x)} (K-1) \mathbb{E}_{y_k \sim p(y)} \frac{p(y_k|x)}{p(y_k)} \right) \right] + \log K \\
&= -\mathbb{E} \left[ \log \left( 1 + \frac{p(y_1)}{p(y_1|x)} (K-1) \right) \right] + \log K \\
&\approx -\mathbb{E} \left[ \log \frac{p(y)}{p(y|x)} \right] - \log(K-1) + \log K \\
&= I(x; y) - \log(K-1) + \log K
\end{aligned} \tag{7}$$

506 Now taking  $K \rightarrow +\infty$ , the last two terms cancel out.

507 **Putting it together.** Combining  $I(x; y) \geq \log K$  with Proposition 1 and Equation 6, we have Lemma 508 1:

$$I_{\text{InfoNCE}}(x; y|\psi, K) \leq \log K \leq I(x; y). \tag{8}$$

509 Besides, according to Equation 7, with sample size  $K \rightarrow +\infty$ ,  $K$ -sample InfoNCE bound is sharp 510 and approaches to the true MI  $I(x; y)$ , i.e.,  $I_{\text{InfoNCE}}(x; y|\psi, K) \approx \log K \approx I(x; y)$ .

## 511 B Proof for Lemma 2

512 **Step 1.** According to Lemma 1, we have  $I_{\text{InfoNCE}}(c; \pi_c; \tau_c|\psi, K) \leq \log K \leq I_{\text{SaMI}}(c; \pi_c; \tau_c)$  (shown 513 in Figure 3).

514 **Step 2.** Let us prove SaNCE is a  $K$ -sample SaNCE estimator and upper bounded by  $\log K$ . Because 515  $\frac{f_\psi(c, \pi_c, \tau_c^+)}{f_\psi(c, \pi_c, \tau_c^+) + \sum_{k=2}^K f_\psi(c, \pi_c, \tau_{c,k}^-)} \leq 1$  [Mu et al., 2022], so we have:

$$\begin{aligned}
I_{\text{SaNCE}}(c; \pi_c; \tau_c|\psi, K) &= \mathbb{E}_{p(c_1, \pi_{c_1}, \tau_{c_1}^+) p(\tau_{c_1, 2:K}^-)} \left[ \log \left( \frac{K \cdot f_\psi(c_1, \pi_{c_1}, \tau_{c_1}^+)}{f_\psi(c_1, \pi_{c_1}, \tau_{c_1}^+) + \sum_{k=2}^K f_\psi(c_1, \pi_{c_1}, \tau_{c_1, k}^-)} \right) \right] \\
&= \mathbb{E}_{p(c_1, \pi_{c_1})} \left[ \mathbb{E}_{p(\tau_{c_1, 2:K}^-)} \log \left( \frac{K \cdot f_\psi(c_1, \pi_{c_1}, \tau_{c_1}^+)}{f_\psi(c_1, \pi_{c_1}, \tau_{c_1}^+) + \sum_{k=2}^K f_\psi(c_1, \pi_{c_1}, \tau_{c_1, k}^-)} \right) \right] \\
&\leq \log K
\end{aligned} \tag{9}$$

516 Hence, we have  $I_{\text{SaNCE}}(c; \pi_c; \tau_c|\psi, K) \leq \log K$  (similar to Equation 6).

517 **Step 3.** With the definition of  $K^*$ , we can prove  $I_{\text{InfoNCE}}(c; \pi_c; \tau_c|\psi, K) \leq I_{\text{SaNCE}}(c; \pi_c; \tau_c|\psi, K)$  518 with the same sample size  $K$ . In task  $e_1$  with context embedding  $c_1$ , SaNCE obtains positive and 519 negative samples from the current task  $e_1$ . Hence the variable  $c = c_1$  is constant, we have:

$$\begin{aligned}
I_{\text{SaNCE}}(c; \pi_c; \tau_c|\psi, K) &= \mathbb{E}_{p(c_1, \pi_{c_1}, \tau_{c_1}^+) p(\tau_{c_1, 2:K}^-)} \left[ \log \left( \frac{K \cdot f_\psi(c_1, \pi_{c_1}, \tau_{c_1}^+)}{f_\psi(c_1, \pi_{c_1}, \tau_{c_1}^+) + \sum_{k=2}^K f_\psi(c_1, \pi_{c_1}, \tau_{c_1, k}^-)} \right) \right] \\
&\leq \mathbb{E}_{p(c_1, \pi_{c_1}, \tau_{c_1}^+) p(\tau_{c_1, 2:K^*_{\text{SaNCE}}}^-)} \left[ \log \left( \frac{K^*_{\text{SaNCE}} \cdot f_\psi(c_1, \pi_{c_1}, \tau_{c_1}^+)}{f_\psi(c_1, \pi_{c_1}, \tau_{c_1}^+) + \sum_{k=2}^{K^*_{\text{SaNCE}}} f_\psi(c_1, \pi_{c_1}, \tau_{c_1, k}^-)} \right) \right] \\
&= \mathbb{E}_{p(\pi_{c_1}, \tau_{c_1}^+) p(\tau_{c_1, 2:K^*_{\text{SaNCE}}}^-)} \left[ \log \left( \frac{K^*_{\text{SaNCE}} \cdot f_\psi(\pi_{c_1}, \tau_{c_1}^+)}{f_\psi(\pi_{c_1}, \tau_{c_1}^+) + \sum_{k=2}^{K^*_{\text{SaNCE}}} f_\psi(\pi_{c_1}, \tau_{c_1, k}^-)} \right) \right] \text{(} c_1 \text{ is constant.)} \\
&\approx \log K^*_{\text{SaNCE}}
\end{aligned} \tag{10}$$

520 The required sample size  $K_{\text{SaNCE}}^* = |c_1| \cdot |\pi| \cdot M = |\pi| \cdot M$ . With  $K \rightarrow K_{\text{SaNCE}}^*$ ,  
 521  $I_{\text{SaNCE}}(c; \pi_c; \tau_c | \psi, K) \approx I_{\text{SaMI}}(c; \pi_c; \tau_c)$ . Correspondingly, for InfoNCE, in the current task  $e_1$   
 522 with context embedding  $c_1$ , positive samples are trajectories  $\tau_1$  generated after executing the skill  
 523  $\pi_1$  in task  $e_1$ , and negative samples are trajectories  $\{\tau_{c_2}^-\}$  from other tasks  $\{e_2, \dots\}$ . Under the  
 524 definition of  $K^*$ , we have:

$$\begin{aligned} & I_{\text{InfoNCE}}(c; \pi_c; \tau_c | \psi, K) \\ &= \mathbb{E}_{p(c, \pi_c, \tau_{c_1}) p(\tau_{c_2:|c|}, 2:K)} \left[ \log \left( \frac{K \cdot f_\psi(c, \pi_c, \tau_{c_1})}{f_\psi(c, \pi_c, \tau_{c_1}) + \sum_{k=2}^K f_\psi(c, \pi_c, \tau_{c_2:|c|, k})} \right) \right] \\ &\leq \mathbb{E}_{p(c, \pi_c, \tau_{c_1}) p(\tau_{c_2:|c|}, 2:K_{\text{InfoNCE}}^*)} \left[ \log \left( \frac{K_{\text{InfoNCE}}^* \cdot f_\psi(c, \pi_c, \tau_{c_1})}{f_\psi(c, \pi_c, \tau_{c_1}) + \sum_{k=2}^{K_{\text{InfoNCE}}^*} f_\psi(c, \pi_c, \tau_{c_2:|c|, k})} \right) \right] \\ &\approx \log K_{\text{InfoNCE}}^*, \end{aligned} \quad (11)$$

525 where  $K_{\text{InfoNCE}}^* = |c| \cdot |\pi| \cdot M \approx |c| \cdot K_{\text{SaNCE}}^*$ . In real-world robotic control tasks, the sample space  
 526 size significantly increases due to multiple environmental features  $e = \{e^0, e^1, \dots, e^N\}$ .  $|c|$  is the  
 527 number of different tasks and increases exponentially due to the permutations and combinations of  $N$   
 528 environmental features. When current task  $e_1$  has context embedding  $c_1$ , the  $c_{2:|c|}$  refer to the context  
 529 embedding for the other tasks. With  $K \rightarrow K_{\text{InfoNCE}}^*$ ,  $I_{\text{InfoNCE}}(c; \pi_c; \tau_c | \psi, K) \approx I_{\text{SaMI}}(c; \pi_c; \tau_c)$ .  
 530 Hence, during the training process,  $I_{\text{InfoNCE}}(c; \pi_c; \tau_c | \psi, K) \leq I_{\text{SaNCE}}(c; \pi_c; \tau_c | \psi, K)$  with the same  
 531 sample size  $K$ .

532 **Step 4.** What remains is to show  $I_{\text{SaMI}}(c; \pi_c; \tau_c) \leq I(c; \tau_c)$ . The MI for three variables is also called  
 533 interaction information. According to the definition in McGill [1954], the SaMI can be presented as:

534 **Proposition 2** For the case of three variables, the MI could be written as  $I_{\text{SaMI}}(c; \pi_c; \tau_c) = I(c; \tau_c) -$   
 535  $I(c; \tau_c | \pi_c)$ .

536 Using this proposition, one can see that in the case of three variables, interaction information quantifies  
 537 how much the information shared between two variables differs from what they share if the third  
 538 variable is known. Several properties of interaction information in the case of three variables have  
 539 been studied in the literature. Specifically, Yeung [1991] showed that

$$-\min\{I(x; \tau_c | \pi_c), I(c; \pi_c | \tau_c), I(\tau_c; \pi_c | c)\} \leq I(c; \tau_c; \pi_c) \leq \min\{I(c; \tau_c), I(c; \pi_c), I(\tau_c; \pi_c)\} \quad (12)$$

540 Combining Equation 12 and Proposition 2, we have  $I_{\text{SaMI}}(c; \pi_c; \tau_c) \leq I(c; \tau_c)$ .

541 **Putting it together.** Hence, we have Lemma 2: we always have  $I_{\text{InfoNCE}}(c; \pi_c; \tau_c | \psi, K) \leq$   
 542  $I_{\text{SaNCE}}(c; \pi_c; \tau_c | \psi, K) \leq \log K \leq I_{\text{SaMI}}(c; \pi_c; \tau_c) \leq I(c; \tau_c)$  (shown in Figure 3), while  
 543 learning a skill-aware context encoder  $\psi$  with SaNCE estimator.  $K_{\text{SaNCE}}^* \ll K_{\text{InfoNCE}}^*$   
 544 so that  $I_{\text{SaNCE}}(c; \pi_c; \tau_c | \psi, K)$  is a much tighter lower bound of the true  $I_{\text{SaMI}}(c; \pi_c; \tau_c)$  than  
 545  $I_{\text{InfoNCE}}(c; \pi_c; \tau_c | \psi, K)$ .

## 546 C Sample size of $I_{\text{Sa+InfoNCE}}$

547 We illustrate the sample size of  $I_{\text{Sa+InfoNCE}}(c; \pi_c; \tau_c | \psi, K)$  in this section. Sa+InfoNCE plugs SaNCE  
 548 into InfoNCE, and has positive samples  $\tau_{c_1}^+$  from task  $e_1$  after executing skill  $\pi_{c_1}^+$ , and negative  
 549 samples are trajectories  $\tau_{c_{1:K}}^-$  from executing skills  $\pi_{c_{1:K}}^-$  in task  $e_{1:K}$ , respectively. Therefore, this  
 550 is equivalent to observing the variable  $c$ , then observing the variable  $\pi_c$ , i.e., sampling from the  
 551 distribution  $p(\pi_c, \tau_c | c)p(c)$ . We have:

$$\begin{aligned} & I_{\text{Sa+InfoNCE}}(c; \pi_c; \tau_c | \psi, K) \\ &= \mathbb{E}_{p(c) p(\pi_{c_1}^+, \tau_{c_1}^+ | c) p((\pi_{2:|c|}^-, \tau_{2:|c|}^-)_{2:K})} \left[ \log \left( \frac{K \cdot f_\psi(c, \pi_{c_1}^+, \tau_{c_1}^+)}{f_\psi(c, \pi_{c_1}^+, \tau_{c_1}^+) + \sum_{k=2}^K f_\psi(c, \pi_{2:|c|, k}^-, \tau_{2:|c|, k}^-)} \right) \right] \\ &\leq \mathbb{E}_{p(c) p(\pi_{c_1}^+, \tau_{c_1}^+ | c) p((\pi_{2:|c|}^-, \tau_{2:|c|}^-)_{2:K_{\text{Sa+InfoNCE}}^*})} \left[ \log \left( \frac{K_{\text{Sa+InfoNCE}}^* \cdot f_\psi(c, \pi_{c_1}^+, \tau_{c_1}^+)}{f_\psi(c, \pi_{c_1}^+, \tau_{c_1}^+) + \sum_{k=2}^{K_{\text{Sa+InfoNCE}}^*} f_\psi(c, \pi_{2:|c|, k}^-, \tau_{2:|c|, k}^-)} \right) \right] \\ &\approx \log K_{\text{Sa+InfoNCE}}^*. \end{aligned} \quad (13)$$

552 It should be noted that such a combination will increase the size of the negative sample space, i.e.,  
553  $K_{\text{Sa+InfoNCE}}^* = \left( \sum_{i=1}^{|c|} |\pi_{ci}^-| + |\pi_{ci}^+| \right) \cdot M \geq K_{\text{SaNCE}}^*$ . Figure 4 illustrates the differences in the size  
554 of the negative sample space, in which the negative sample space may vary across tasks (as shown by  
555 the non-lining-up bars in Figure 4 (c)). This is because we define negative samples as trajectories  
556 with low returns, so the size of the negative sample space is influenced by sampling randomness.  
557 With the same number  $K$  of samples,  $I_{\text{Sa+InfoNCE}}$  is less precise and looser than  $I_{\text{SaNCE}}$ . Hence, a  
558 trade-off between sample diversity and the precision of the  $K$ -sample estimator is required.

## 559 D Environmental setup

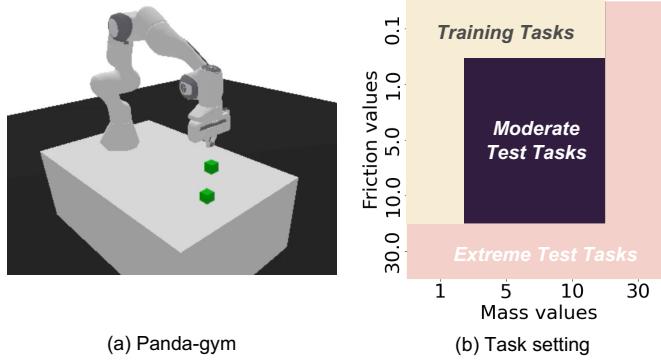


Figure 8: (a) Modified Panda-gym benchmarks, (b) the training tasks, the moderate test tasks and the extreme test tasks. The moderate test task setting just has a combinatorial interpolation. The extreme test task setting has unseen ranges of environmental features and is an extrapolation.

### 560 D.1 Modified Panda-gym

561 We modified the original Pick&Place task in Panda-gym [Gallouédec et al., 2021] by setting the  
562  $z$  dimension (i.e., the desired height) of the cube’s goal position equal to 0<sup>3</sup> and maintaining the  
563 freedom<sup>4</sup> of grippers, so the agent can explore whether it is supposed to push the cube or grasp it.  
564 Skills in this benchmark are defined as:

- 565 • **Pick&Place skill:** This skill specifically refers to the agent trying to use the gripper to  
566 grasp the cube, pick it up off the table and place it in the goal position. We determine the  
567 Pick&Place skill by detecting no contact points between the table and the cube, two contact  
568 points between the robot’s end effector and the cube, and the cube’s height being greater  
569 than half its width, meaning it is on the table.
- 570 • **Push skill:** This skill involves the agent moving cubes on the table to the goal position,  
571 either by dragging or sliding the cube to the goal location. We confirm the Push skill by  
572 detecting that the cube’s height equals half its width.
- 573 • **Other skills:** Besides the Pick&Place and Push skills, any other different modes of behaviour  
574 are classified as other skills. For example, with the Flip skill, the agent applies an initial  
575 force to the cube, causing it to roll on the surface or leave the table. In this case, we detect  
576 that the cube’s height is greater than half its width, and the number of contact points between  
577 the cube and the table is greater than zero.

578 Some elements in the RL framework are defined as the following:

<sup>3</sup>If  $z$  is not equal to 0, Pick&Place skill is always needed to solve tasks.

<sup>4</sup>In the original Push task, the grippers are blocked to ensure the agent can only push cubes. However, this hinders the agent from learning Pick&Place skills, leading to failure when it encounters an "unpushable" scenario.

579 **State space:** we use feature vectors which contain cube position (3 dimensions), cube rotation  
 580 (3 dimensions), cube velocity (3 dimensions), cube angular velocity (3 dimensions), end-effector  
 581 position (3 dimensions), end-effector velocity (3 dimensions), gripper width (1 dimension), desired  
 582 goal (3 dimensions) and achieved goal (3 dimensions). Environmental features are not included in  
 583 the state.

584 **Action space:** the action space has 4 dimensions, the first three dimensions are the end-effector's  
 585 position changes and the last dimension is the gripper's width change.

586 During training, we randomly select a combination of environmental features from a training  
 587 set sampling combinations from sets: mass = 1.0 and friction  $\in \{0.1, 1.0, 5.0, 10.0\}$ ; mass  $\in$   
 588  $\{1.0, 5.0, 10.0\}$  and friction = 0.1. At test time, we evaluate each algorithm in all tasks from  
 589 the moderate test task setting, in which mass  $\in \{5.0, 10.0\}$  and friction  $\in \{1.0, 5.0, 10.0\}$   
 590 (shown in Figure 8(b)), and all tasks from extreme test task setting: mass = 30.0 and friction  $\in$   
 591  $\{0.1, 1.0, 5.0, 10.0, 30.0\}$ ; mass  $\in \{1.0, 5.0, 10.0, 30.0\}$  and friction = 30.0 (shown in Figure 8(b)).

## 592 D.2 Modified MuJoCo

593 We extended the modified MuJoCo benchmark introduced in DOMINO [Mu et al., 2022] and CaDM  
 594 [Lee et al., 2020]. Hence, in our extension, there are four new-added environments (Walker, Crippled  
 595 Hopper, Crippled Walker, HumanoidStandup) compared with the original benchmark. Additionally,  
 596 in our experiments, we used a different task set design (Table 3) than in the DOMINO and CaDM  
 597 papers. For Hopper, Walker, Half-cheetah, Ant, HumanoidStandup and SlimHumanoid, we use the  
 598 environments from the MuJoCo physics engine and use implementation available from Clavera et al.  
 599 [2019b] and Seo et al. [2020], and scale the mass of every rigid link by scale factor  $m$ , and scale  
 600 damping of every joint by scale factor  $d$ . For Crippled Ant, Crippled Hopper, Crippled Walker and  
 601 Crippled Half-cheetah, we use implementation available from Seo et al. [2020] and scale the mass  
 602 of every rigid link by scale factor  $m$ , scale damping of every joint by scale factor  $d$ , and randomly  
 603 select one leg, and make it crippled to change the dynamic transitions. Generalisation performance is  
 604 measured in two different regimes: moderate and extreme, where the moderate draws Environmental  
 605 features from a closer range to the training range, compared to the extreme. We have provided our  
 606 settings for training, extreme and moderate test tasks in Table 3.

## 607 D.3 Implementation details

608 In this section, we provide the implementation details for SaMI. We show the pseudo-code for using  
 609 SaNCE during meta-training and meta-testing in Algorithm 1 and 2. Our codebase is built on top  
 610 of the publicly released implementation Stable Baselines3 by Raffin et al. [2021] as well as the  
 611 implementation of InfoNCE by Oord et al. [2019]. A public and open-source implementation of  
 612 SaMI is available at <https://github.com/uoee-agents/SaMI>.

613 **Base algorithm.** We use SAC [Haarnoja et al., 2018] for downstream evaluation of the learned context  
 614 embedding. SAC is an off-policy actor-critic method that uses the maximum entropy framework for

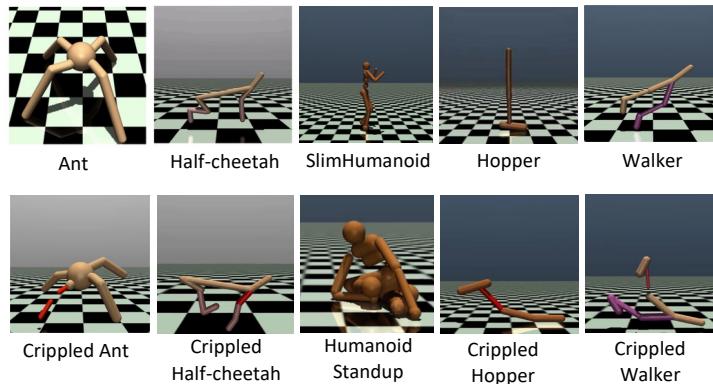


Figure 9: Ten environments in modified MuJoCo benchmark.

Table 3: Environmental features used for MuJoCo benchmark.

	Training	Test (Moderate)	Test (Extreme)	Episode Length
Half-cheetah	$m \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$ $d \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$	$m \in \{0.40, 0.50, 1.50, 1.60\}$ $d \in \{0.40, 0.50, 1.50, 1.60\}$	$m \in \{0.20, 0.40, 1.60, 1.80, 4.00\}$ $d \in \{0.20, 0.40, 1.60, 1.80, 4.00\}$	1000
Ant	$m \in \{0.40, 0.50, 1.50, 1.60\}$ $d \in \{0.40, 0.50, 1.50, 1.60\}$	$m \in \{0.20, 0.40, 1.60, 1.80\}$ $d \in \{0.20, 0.40, 1.60, 1.80\}$	$m \in \{0.20, 0.40, 1.60, 1.80, 4.00\}$ $d \in \{0.20, 0.40, 1.60, 1.80, 4.00\}$	1000
Hopper	$m \in \{0.75, 1.0, 1.25\}$ $d \in \{0.75, 1.0, 1.25\}$	$m \in \{0.40, 0.50, 1.50, 1.60\}$ $d \in \{0.40, 0.50, 1.50, 1.60\}$	$m \in \{0.20, 0.40, 1.60, 1.80, 4.00\}$ $d \in \{0.20, 0.40, 1.60, 1.80, 4.00\}$	1000
Crippled Hopper	$m \in \{0.75, 1.0, 1.25\}$ $d \in \{0.75, 1.0, 1.25\}$	$m \in \{0.40, 0.50, 1.50, 1.60\}$ $d \in \{0.40, 0.50, 1.50, 1.60\}$	$m \in \{0.20, 0.40, 1.60, 1.80, 4.00\}$ $d \in \{0.20, 0.40, 1.60, 1.80, 4.0\}$ Crippled Legs $R_1 \in \{0, 1, 2\}$	1000
SlimHumanoid	$m \in \{0.80, 0.90, 1.0, 1.15, 1.25\}$ $d \in \{0.80, 0.90, 1.0, 1.15, 1.25\}$	$m \in \{0.60, 0.70, 1.50, 1.60\}$ $d \in \{0.60, 0.70, 1.50, 1.60\}$	$m \in \{0.40, 0.50, 1.70, 1.80\}$ $d \in \{0.40, 0.50, 1.70, 1.80\}$	1000
HumanoidStandup	$m \in \{0.80, 0.90, 1.0, 1.15, 1.25\}$ $d \in \{0.80, 0.90, 1.0, 1.15, 1.25\}$	$m \in \{0.60, 0.70, 1.50, 1.60\}$ $d \in \{0.60, 0.70, 1.50, 1.60\}$	$m \in \{0.40, 0.50, 1.70, 1.80, 4.00\}$ $d \in \{0.40, 0.50, 1.70, 1.80, 4.00\}$	1000
Walker	$m \in \{0.75, 1.0, 1.25\}$ $d \in \{0.75, 1.0, 1.25\}$	$m \in \{0.40, 0.50, 1.50, 1.60\}$ $d \in \{0.40, 0.50, 1.50, 1.60\}$	$m \in \{0.20, 0.40, 1.60, 1.80, 4.00\}$ $d \in \{0.20, 0.40, 1.60, 1.80, 4.00\}$	2000
Crippled Walker	$m \in \{0.75, 1.0, 1.25\}$ $d \in \{0.75, 1.0, 1.25\}$ Crippled Joints (right leg) = $\{0, 1, 2\}$	$m \in \{0.40, 0.50, 1.50, 1.60\}$ $d \in \{0.40, 0.50, 1.50, 1.60\}$ Crippled Joints (right leg) $\in \{0, 1, 2\}$	$m \in \{0.20, 0.40, 1.60, 1.80, 4.00\}$ $d \in \{0.20, 0.40, 1.60, 1.80, 4.00\}$ Crippled Joints (left leg) $\in \{3, 4, 5\}$	2000
Crippled Ant	$m \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$ $d \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$ Crippled Legs $R_1 = \emptyset$ or $R_1 \in \{0, 1, 2, 3\}$	$m \in \{0.40, 0.50, 1.50, 1.60\}$ $d \in \{0.40, 0.50, 1.50, 1.60\}$ Crippled Legs $R_1 \in \{0, 1, 2, 3\}$	$m \in \{0.20, 0.40, 1.60, 1.80\}$ $d \in \{0.20, 0.40, 1.60, 1.80\}$ Crippled Legs $\{R_1, R_2\} \in \{0, 1, 2, 3, 4, 5\}$ ( $R_1 \neq R_2$ )	2000
Crippled Half-cheetah	$m \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$ $d \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$ Crippled Joints (front leg) $R_1 \in \{3, 4, 5\}$	$m \in \{0.40, 0.50, 1.50, 1.60\}$ $d \in \{0.40, 0.50, 1.50, 1.60\}$ Crippled Joints (back leg) $R_1 \in \{0, 1, 2\}$	$m \in \{0.20, 0.40, 1.60, 1.80\}$ $d \in \{0.20, 0.40, 1.60, 1.80\}$ Crippled Joints $\{R_1, R_2\} \in \{0, 1, 2, 3, 4, 5\}$ ( $R_1 \neq R_2$ )	2000

Table 4: Hyperparameters used in Panda-gym and MuJoCo benchmarks. Most hyperparameter values are unchanged across tasks except for contrastive batch size and SaNCE loss coefficient.

Hyperparameter	Value
Replay buffer size	100,000
Contrastive batch size	MuJoCo 12, Panda-gym 256
SaNCE loss coefficient $\alpha$	MuJoCo 1.0, Panda-gym 0.01
Context embedding dimension	6
Hidden state dimension	128
Learning rate (actor, critic and encoder)	1e-3
Training frequency (actor, critic and encoder)	128
Gradient steps	16
Momentum context encoder $\psi^*$ soft-update rate	0.05
SAC target soft-update rate	critic 0.01, actor 0.05
SAC batch size	256
Discount factor	0.99
Optimizer	Adam

615 soft policy iteration. At each iteration, SAC performs soft policy evaluation and improvement steps.  
 616 We use the same SAC implementation across all baselines and other methods. In the Meta-training  
 617 phase, we trained agents for 1.6 million timesteps in each environment on the Panda-gym and MuJoCo  
 618 benchmarks. For meta-testing, we tested 100 episodes in each environment, with tasks randomly  
 619 sampled from the moderate and extreme task sets.

620 **Encoder architecture.** For our method, the context encoder  $\psi$  is modelled as a Long Short-Term  
 621 Memory (LSTM) that produces a 128-dimensional hidden state vector, subsequently processed  
 622 through a single-layer feed-forward network to generate a 6-dimensional context embedding. We  
 623 hope an agent can complete the three steps of "explore effectively, infer, adapt" within an episode.  
 624 Therefore, we initialise the hidden state and cell state of LSTM to zero at the start of each episode.  
 625 The actor and critic both use the same context encoder to embed trajectories. For contrastive learning,  
 626 SaNCE utilises a momentum encoder  $\psi^*$  to generate positive and negative context embeddings  
 627 [Laskin et al., 2020, He et al., 2020]. Formally, denoting the parameters of  $\psi$  as  $\theta_\psi$  and those of  $\psi^*$   
 628 as  $\theta_{\psi^*}$ , we update  $\theta_{\psi^*}$  by:

$$\theta_{\psi^*} \leftarrow m \cdot \theta_\psi + (1 - m) \cdot \theta_{\psi^*}. \quad (14)$$

629 Here  $m \in [0, 1]$  is a soft-update rate. Only the parameters  $\theta_\psi$  are updated by back-propagation. The  
 630 momentum update in Equation 14 makes  $\theta_{\psi^*}$  evolve more smoothly by having them slowly track the  
 631  $\theta_\psi$  with  $m \ll 1$  (e.g.,  $m = 0.05$  in this research). This means that the target values are constrained to  
 632 change slowly, greatly improving the stability of learning.

633 **Hyperparameters.** A full list of hyperparameters is displayed in Table 4.

634 **Hardware.** For each experiment run we use a single NVIDIA Volta V100 GPU with 32GB memory  
 635 and a single CPU.

---

**Algorithm 1** SaNCE Meta-training

---

**Require:** Batch of training tasks  $\{e_n\}_{n=1,\dots,N}$  from  $\xi_{train}(e)$ , soft-update rate  $m$ ;

- 1: Initialize RL replay buffer  $\mathcal{B}_{RL}$ , encoder replay buffer  $\mathcal{B}_{enc}$ ;
- 2: Initialize parameters  $\psi$  for context encoder,  $\psi^*$  for momentum context encoder and  $\phi$  for the off-policy SAC;
- 3: **while** not done **do**
- 4:   **for** each task  $e_n$  **do**
- 5:     **for** Roll-out time steps **do**
- 6:       **for** time step  $t <$  maximum episode length  $T$  **do**
- 7:         Update context embedding  $c_n \sim \psi(c_n | \tau_{c_n, 0:t})$
- 8:         Roll-out policy  $\pi_{c_n}(a_t | s_t, c_n)$  and accumulate transition  $(s_t, a_t, r_t, s_{t+1})$ ;
- 9:        **end for**
- 10:      Add trajectory  $\tau_{c_n} = \{s_0, a_0, r_0, s_1, r_1, \dots, s_T, a_T, r_T\}$  to replay buffer  $\mathcal{B}_{RL}^n$  and  $\mathcal{B}_{enc}^n$ ;
- 11:    **end for**
- 12:   **end for**
- 13:   **for** each training step **do**
- 14:     **for** each task  $e_n$  **do**
- 15:       Sample RL batch  $\{\tau_{c_n}\} \sim \mathcal{B}_{RL}^n$ ;
- 16:       Sample a positive sample  $\tau_{c_n}^+$  for generating query with highest return, positive samples  $\{\tau_{c_n}^+\}$  and negative samples  $\{\tau_{c_n}^-\}$  for encoding positive and negative embeddings;
- 17:       Update  $\phi$  with RL loss  $\mathcal{L}_{RL}$ ;
- 18:       Update  $\psi$  with SaNCE loss  $\mathcal{L}_{SaNCE}$  and RL loss  $\mathcal{L}_{RL}$ ;
- 19:        $\theta_{\psi^*} \leftarrow m \cdot \theta_\psi + (1 - m) \cdot \theta_{\psi^*}$ ;
- 20:     **end for**
- 21:   **end for**
- 22: **end while**

---

---

**Algorithm 2** SaNCE Meta-testing

---

**Require:** Batch of training tasks  $\{e_n\}_{n=1,\dots,N}$  from  $\xi_{test}(e)$ ;

- 1: **while** not done **do**
- 2:   **for** each task  $e_n$  **do**
- 3:     **for** each episode **do**
- 4:       **for** time step  $t <$  maximum episode length  $T$  **do**
- 5:         Update context embedding  $c_n \sim \psi(c_n | \tau_{c_n, 0:t})$
- 6:         Roll-out policy  $\pi_{c_n}(a_t | s_t, c_n)$  and accumulate transition  $(s_t, a_t, r_t, s_{t+1})$ ;
- 7:        **end for**
- 8:     **end for**
- 9:   **end for**
- 10: **end while**

---

636 **E Additional results**

637 **E.1 Balance contrastive and RL updates: loss coefficient  $\alpha$**

638 While past work has learned hyperparameters to balance the contrastive loss coefficient  $\alpha$  relative  
 639 to the RL objective [Jaderberg et al., 2016, Bachman et al., 2019], we use both the contrastive and  
 640 RL objectives together with equal weight  $\alpha = 1.0$  to be optimal for the MuJoCo benchmark, and  
 641  $\alpha = 0.01$  for the Panda-gym benchmark, we also analyse the effect of loss coefficient  $\alpha$  for CCM,  
 642 SaTESAC and SaCCM in MuJoCo (Figure 11) and Panda-gym (Figure 10) benchmarks.

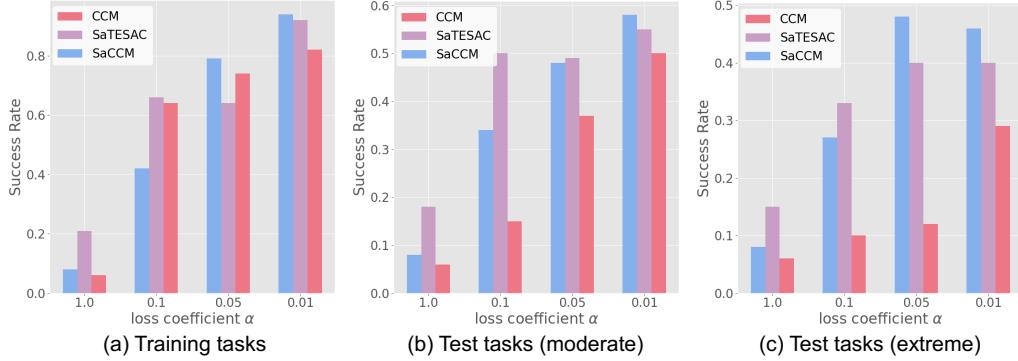


Figure 10: Loss coefficient  $\alpha$  analysis of Panda-gym benchmark in training and test (moderate and extreme) tasks.

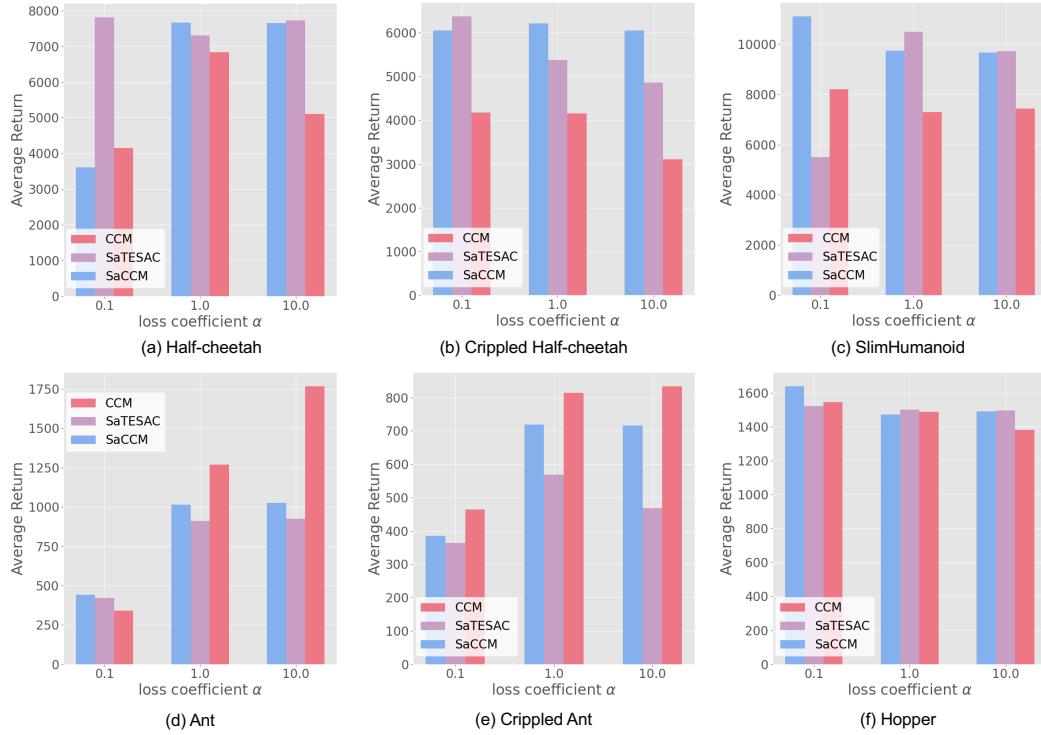


Figure 11: Loss coefficient  $\alpha$  analysis of MuJoCo benchmark in training tasks.

643 **E.2 Result of log- $K$  curse analysis**

644 **E.2.1 Buffer size**

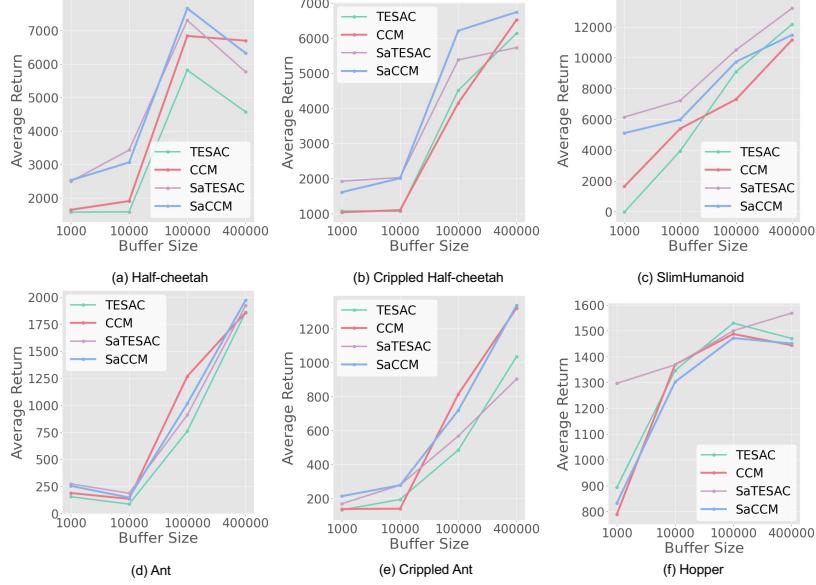


Figure 12: Comparison of different buffer sizes in MuJoCo benchmark in training tasks (over 5 seeds). Buffer size = 400000, 100000, 10000, and 1000.

645 **E.2.2 contrastive batch size**

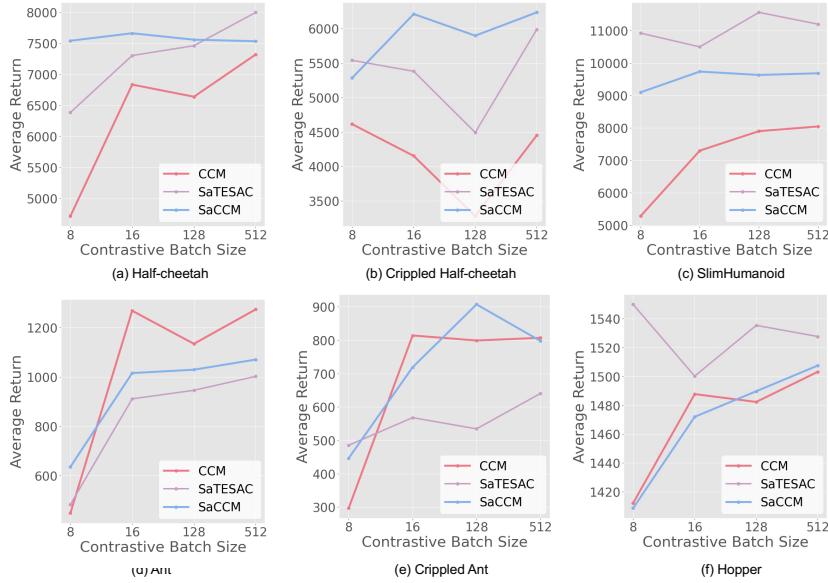


Figure 13: Comparison of different contrastive batch sizes in MuJoCo benchmark in training tasks (over 5 seeds). contrastive batch size = 512, 128, 16, and 8.

646 **F Further skill analysis**

647 **E.1 Panda-gym**

648 **E.1.1 Visualisation of context embedding**

649 We visualise the context embedding via UMAP [McInnes et al., 2020] (Figure ??) and t-SNE [Van der  
 650 Maaten and Hinton, 2008] (Figure 15). When the mass of the cube is high (30 Kg and 10 Kg), the  
 651 agent learned the Push skill (the yellow bounding box in Figure 1(a)), whereas with lower masses,  
 652 the agent learned the Pick&Place skill. However, as shown in Figure 15(b), CCM did not display  
 653 clear skill grouping. This indicates that SaMI extracts high-quality skill-related information from the  
 trajectories and helps with embodying diverse skills.

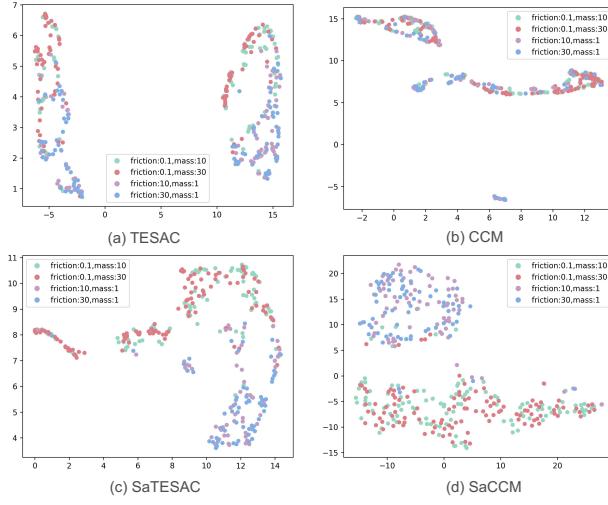


Figure 14: UMAP visualisation of context embedding extracted from trajectories collected in Panda-gym environments.

654

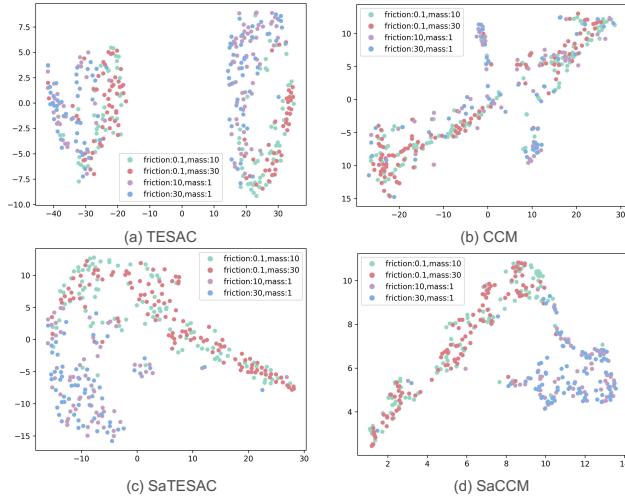


Figure 15: t-SNE visualisation of context embedding extracted from trajectories collected in Panda-gym environments.

655 F.1.2 Heatmap of Panda-gym benchmark

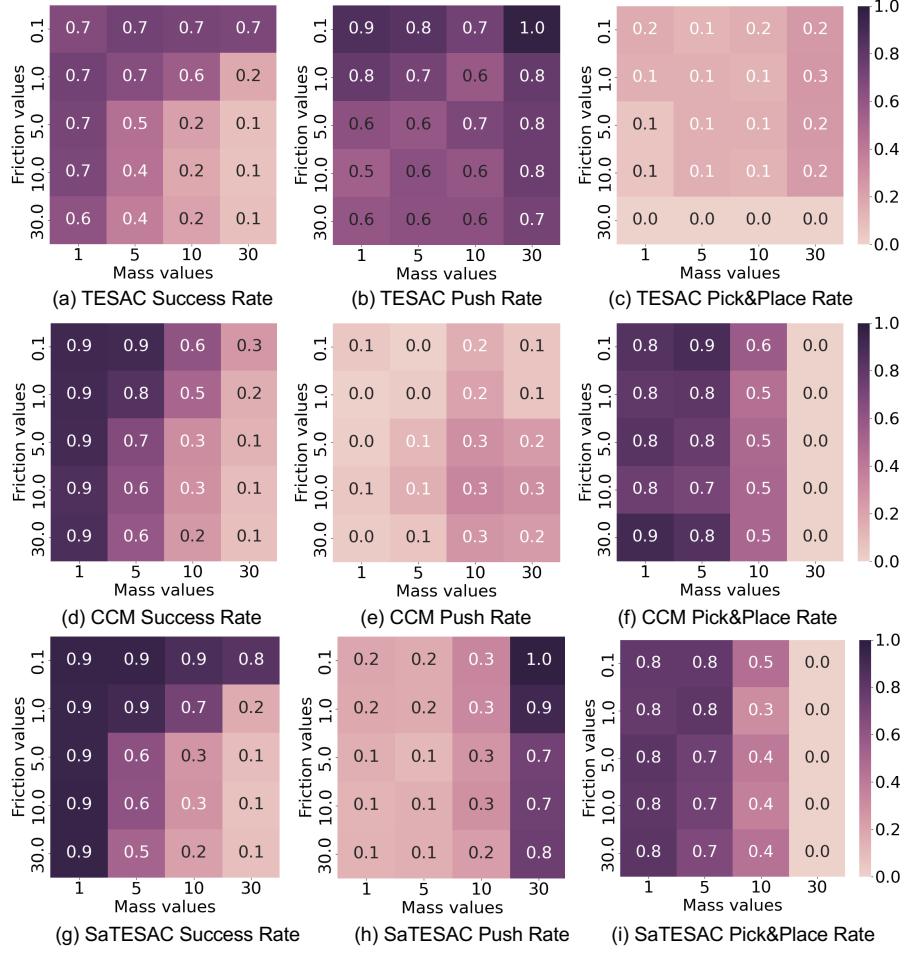


Figure 16: Heatmap of success rate and learned skills of SaCCM. For each grid, the  $(i, j)^{th}$  location shows the probability of the skills executed in 100 evaluations with ( $\text{mass} = i$ ,  $\text{friction} = j$ ). We determine the skills by detecting contact points between the end effector and the cube and between the cube and the table.

656 This section provides the heatmap results and further analysis of TESAC, CCM, and SaTESAC on  
 657 the Panda-gym benchmark. Initially, from the heatmap results of SaTESAC and SaCCM (Figure  
 658 6), we observed that with higher cube masses (30 and 10 kg), the agent adopted the Push skill (as  
 659 indicated by the clustered points in the yellow bounding box in Figure 1(a)). At lower masses, the  
 660 agent adopted the Pick&Place skill.

661 In contrast to CCM, as seen in Figures 16(e-f), CCM predominantly learned the Pick&Place skill,  
 662 resulting in a decline in success rates for tasks at  $mass = 30$ , as the agent could not lift the cube off  
 663 the table using the Pick&Place skill, as depicted in Figure 16(d). The visualisation of the context  
 664 embedding (Figure 15) did not show clear grouping across different tasks.

665 Finally, TESAC only mastered the Push skill. The Push skill is relatively simpler to learn than the  
 666 Pick&Place skill, as it does not require the agent to master manipulating fingers to pick up cubes.  
 667 Consequently, TESAC's success rate notably decreased in environments with higher friction.

668 **F.2 MuJoCo**

669 SaMI helps RL agents be versatile and embody multiple skills. Further, we have displayed rendered  
 670 videos (<https://github.com/uo-e-agents/SaMI>) to better demonstrate the different skills learned.

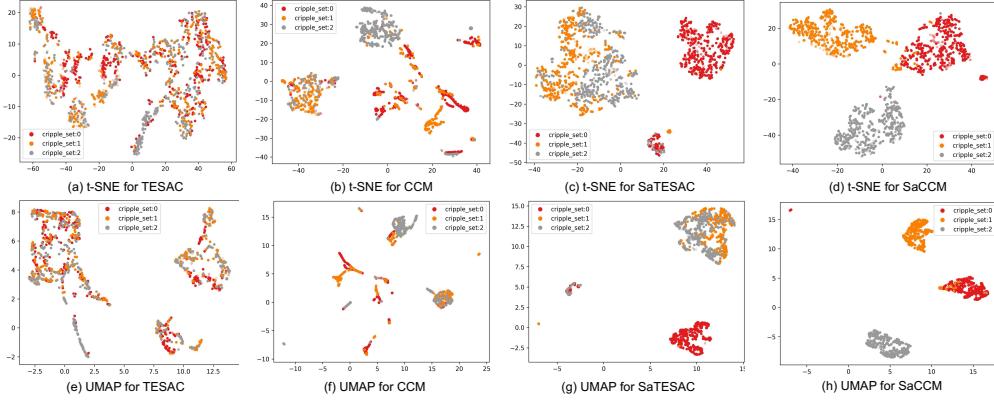


Figure 17: t-SNE and UMAP visualisation of context embedding extracted from trajectories collected in Crippled Half-cheetah environment. "cripple\_set" refers to the index of the crippled joint. The figure shows the context embedding of tasks in three moderate test settings. Combined with the video demos <sup>2</sup> for skill analysis, the Crippled Half-Cheetah robot executed three different forward running skills.

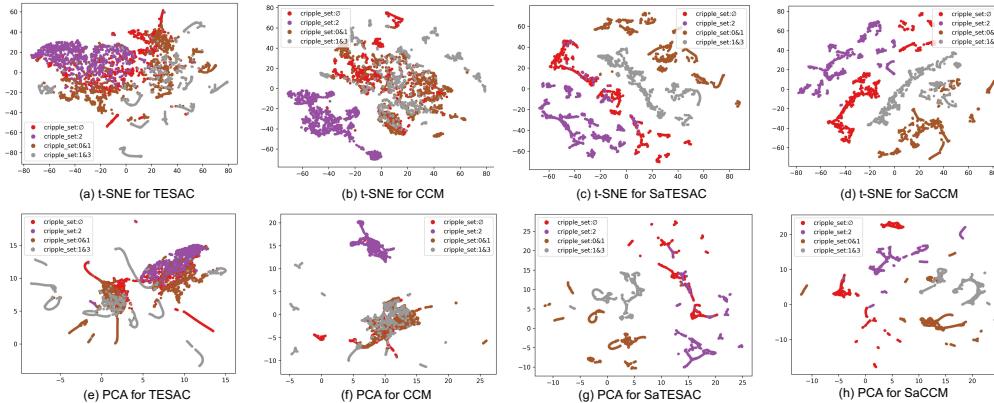


Figure 18: t-SNE and UMAP visualisation of context embedding extracted from trajectories collected in Crippled Ant environment. "cripple\_set" refers to the index of the crippled joint. When 3 or 4 legs are available, the Ant Robot (trained with SaCCM) rolls to adapt to varying mass and damping. However, with only 2 adjacent legs during zero-shot generalisation, it switches to walking. If 2 opposite legs are available, the Ant can still roll but eventually tips over. Please refer to the video demos <sup>2</sup>.

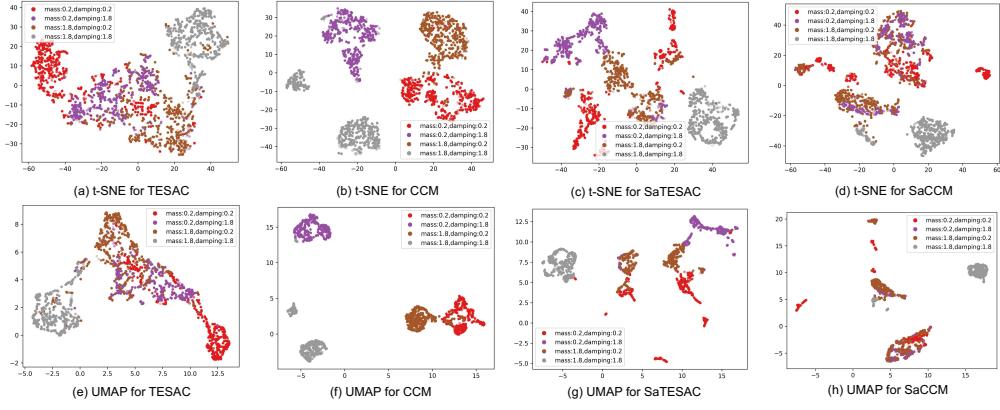


Figure 19: t-SNE and UMAP visualisation of context embedding extracted from trajectories collected in Half-cheetah environment. During zero-shot generalisation, SaCCM demonstrates different skills for running forwards at various speeds, as well as skills for doing flips and faceplanting.

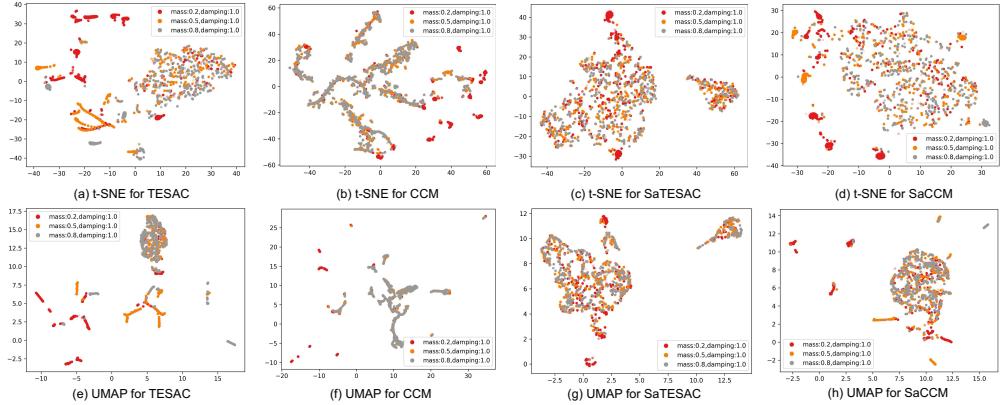


Figure 20: t-SNE and UMAP visualisation of context embedding extracted from trajectories collected in Ant environment. The Ant robot learned a single skill, rolling, to adapt to different mass and damping values.

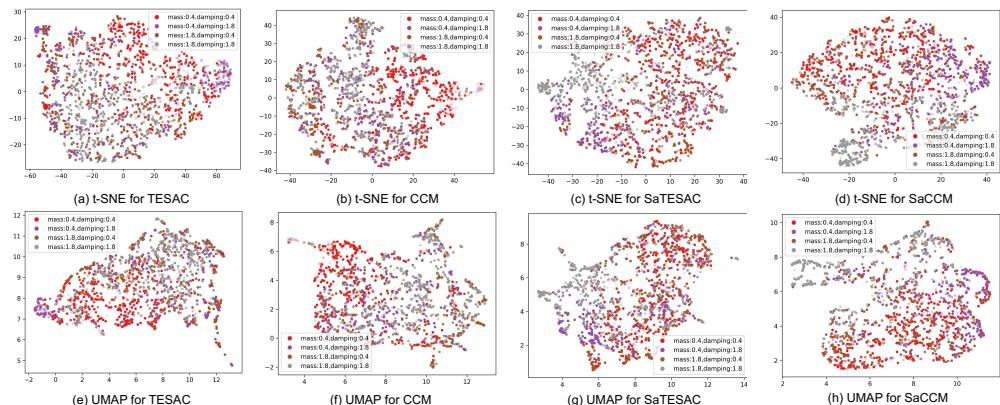


Figure 21: t-SNE and UMAP visualisation of context embedding extracted from trajectories collected in SlimHumanoid environment. The Humanoid Robot crawls on the ground using one elbow. When the damping is relatively high (damping=1.8), the Humanoid Robot can crawl forward stably, but when the damping is low (damping=0.4), it tends to roll.

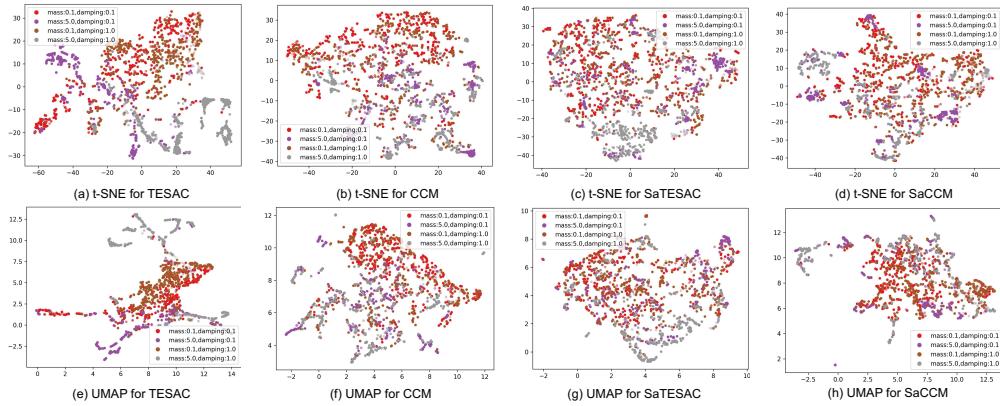


Figure 22: t-SNE and UMAP visualisation of context embedding extracted from trajectories collected in HumanoidStandup environment. SaCCM and SaTESAC learned a sitting posture that makes it easier to stand up, allowing it to generalise well when mass and damping change.

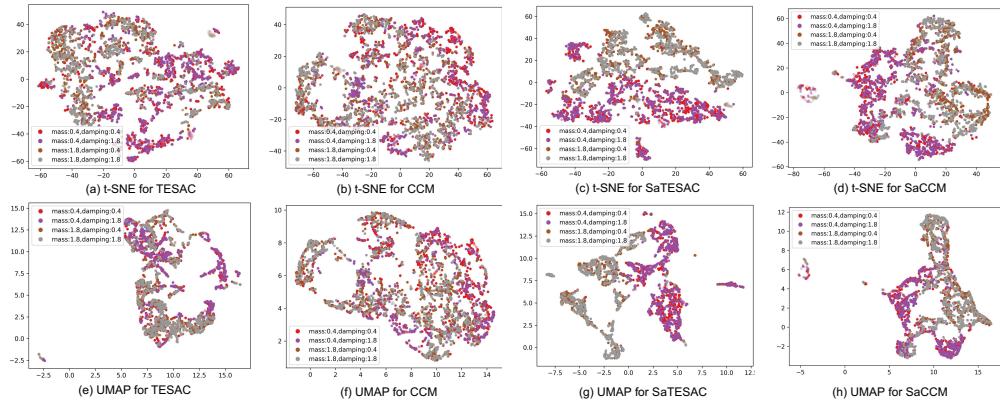


Figure 23: t-SNE and UMAP visualisation of context embedding extracted from trajectories collected in Hopper environment. Combined with the video demos<sup>2</sup> for skill analysis, the plots for SaCCM and SaTESAC show two skills: 1) when the mass is low, the Hopper hops in an upright posture; 2) when the mass is higher, the Hopper hops forward on the floor.

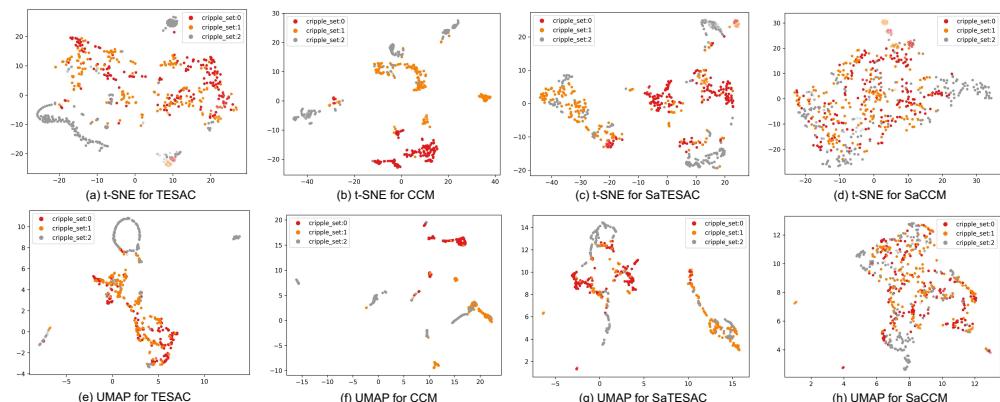


Figure 24: t-SNE and UMAP visualisation of context embedding extracted from trajectories collected in Crippled Hopper environment. "cripple\_set" refers to the index of the crippled joint. SaCCM and SaTESAC learned to take a big hop forward at the beginning (i.e., effective exploration) and then switch to different skills based on environmental feedback.

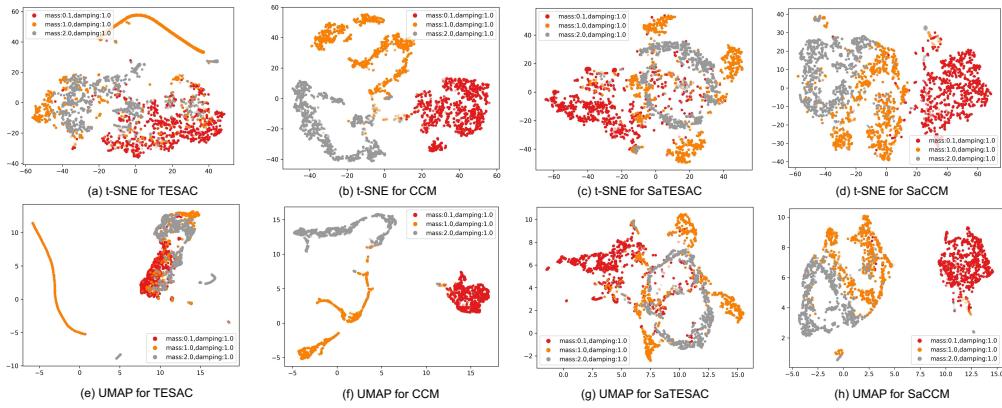


Figure 25: t-SNE and UMAP visualisation of context embedding extracted from trajectories collected in Walker environment. The Walker learned a single skill, hopping forward on the floor through both right and left legs.

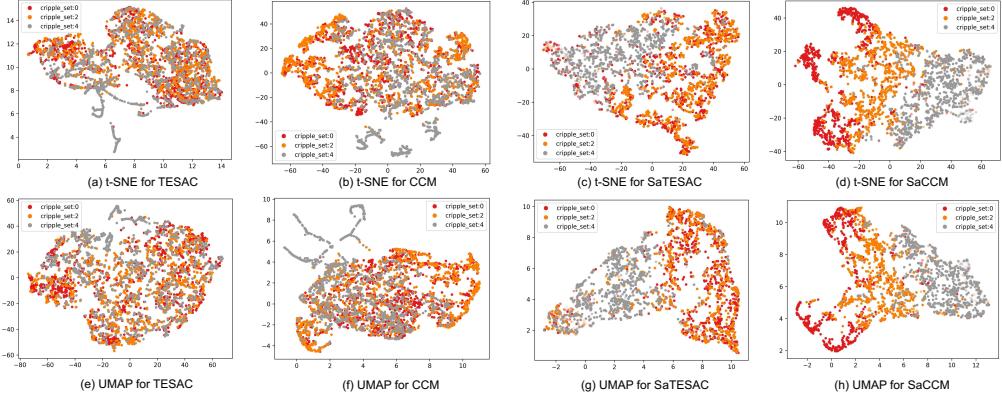


Figure 26: t-SNE and UMAP visualisation of context embedding extracted from trajectories collected in Crippled Walker environment. "cripple\_set" refers to the index of the crippled joint. The Crippled Walker (trained with SaTESAC and SaCCM) learned to hop forward using the right leg (cripple\_set:0 and cripple\_set:2) in the training and moderate test tasks and switched to hopping forward using the left leg (cripple\_set:4) in the extreme test tasks.

Table 5: Environmental features used for MuJoCo benchmark from DOMINO and CaDM.

	Training	Test (Moderate)	Test (Extreme)	Episode Length
Half-cheetah	$m \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$ $d \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$	$m \in \{0.40, 0.50, 1.50, 1.60\}$ $d \in \{0.40, 0.50, 1.50, 1.60\}$	$m \in \{0.20, 0.40, 1.60, 1.80\}$ $d \in \{0.20, 0.40, 1.60, 1.80\}$	1000
Ant	$m \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$ $d \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$	$m \in \{0.40, 0.50, 1.50, 1.60\}$ $d \in \{0.40, 0.50, 1.50, 1.60\}$	$m \in \{0.20, 0.40, 1.60, 1.80\}$ $d \in \{0.20, 0.40, 1.60, 1.80\}$	1000
Hopper	$m \in \{0.75, 1.0, 1.25\}$ $d \in \{0.75, 1.0, 1.25\}$	$m \in \{0.40, 0.50, 1.50, 1.60\}$ $d \in \{0.40, 0.50, 1.50, 1.60\}$	$m \in \{0.20, 0.40, 1.60, 1.80\}$ $d \in \{0.20, 0.40, 1.60, 1.80\}$	500
SlimHumanoid	$m \in \{0.80, 0.90, 1.0, 1.15, 1.25\}$ $d \in \{0.80, 0.90, 1.0, 1.15, 1.25\}$	$m \in \{0.60, 0.70, 1.50, 1.60\}$ $d \in \{0.60, 0.70, 1.50, 1.60\}$	$m \in \{0.40, 0.50, 1.70, 1.80\}$ $d \in \{0.40, 0.50, 1.70, 1.80\}$	500
Crippled Ant	$m \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$ $d \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$ Crippled Joints: {0, 1, 2}	$m \in \{0.40, 0.50, 1.50, 1.60\}$ $d \in \{0.40, 0.50, 1.50, 1.60\}$ Crippled Joints: {3}	$m \in \{0.20, 0.40, 1.60, 1.80\}$ $d \in \{0.20, 0.40, 1.60, 1.80\}$ Crippled Joints: {3}	1000
Crippled Half-cheetah	$m \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$ $d \in \{0.75, 0.85, 1.0, 1.15, 1.25\}$ Crippled Joints: {0, 1, 2, 3}	$m \in \{0.40, 0.50, 1.50, 1.60\}$ $d \in \{0.40, 0.50, 1.50, 1.60\}$ Crippled Joints: {4, 5}	$m \in \{0.20, 0.40, 1.60, 1.80\}$ $d \in \{0.20, 0.40, 1.60, 1.80\}$ Crippled Joints: {4, 5}	1000

Table 6: Comparison of average return  $\pm$  standard deviation with baselines in MuJoCo benchmark (over 5 seeds). The **bold text** signifies the highest average return. The numerical results for PPO+DOMINO are copied from Mu et al. [2022]; the numerical results for PPO+CaDM, Vanilla+CaDM, and PE-TS+CaDM are copied from Lee et al. [2020].

	Ant			Half-cheetah		
	Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
PPO+DOMINO	227 $\pm$ 86	216 $\pm$ 52		2472 $\pm$ 803	1034 $\pm$ 476	
PPO+CaDM	268.6 $\pm$ 77.0	228.8 $\pm$ 48.4	199.2 $\pm$ 52.1	2652.0 $\pm$ 1133.6	1224.2 $\pm$ 630.0	1021.1 $\pm$ 676.6
Vanilla+CaDM	1851.0 $\pm$ 113.7	1315.7 $\pm$ 45.5	821.4 $\pm$ 113.5	3536.5 $\pm$ 641.7	1556.1 $\pm$ 260.6	1264.5 $\pm$ 228.7
PE-TS+CaDM	<b>2848.4<math>\pm</math>61.9</b>	<b>2121.0<math>\pm</math>60.4</b>	<b>1200.7<math>\pm</math>21.8</b>	<b>8264.0<math>\pm</math>1374.0</b>	<b>7087.2<math>\pm</math>1495.6</b>	<b>4661.8<math>\pm</math>783.9</b>
SaTESAC	908 $\pm$ 65	640 $\pm$ 117	532 $\pm$ 88	7430 $\pm$ 1026	4058 $\pm$ 890	1780 $\pm$ 102
SaCCM	928 $\pm$ 141	635 $\pm$ 94	555 $\pm$ 88	7154 $\pm$ 965	3849 $\pm$ 689	1926 $\pm$ 218
	SlimHumanoid			Crippled Half-Cheetah		
	Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
PPO+DOMINO	7825 $\pm$ 1256	5258 $\pm$ 1039		2503 $\pm$ 658	1326 $\pm$ 491	
PPO+CaDM	10455.0 $\pm$ 1004.9	4975.7 $\pm$ 1305.7	3015.1 $\pm$ 1508.3	2356.6 $\pm$ 624.3	1454.0 $\pm$ 462.6	1025.0 $\pm$ 296.2
Vanilla+CaDM	1758.2 $\pm$ 459.1	1228.9 $\pm$ 374.0	1487.9 $\pm$ 339.0	2435.1 $\pm$ 880.4	1375.3 $\pm$ 290.6	966.9 $\pm$ 89.4
PE-TS+CaDM	1371.9 $\pm$ 400.0	903.7 $\pm$ 343.9	814.5 $\pm$ 274.8	3294.9 $\pm$ 733.9	2618.7 $\pm$ 647.1	1294.2 $\pm$ 214.9
SaTESAC	<b>10216<math>\pm</math>1620</b>	<b>7886<math>\pm</math>2203</b>	<b>6123<math>\pm</math>1403</b>	<b>5169<math>\pm</math>730</b>	2184 $\pm$ 592	<b>1628<math>\pm</math>281</b>
SaCCM	<b>9312<math>\pm</math>705</b>	<b>7430<math>\pm</math>1587</b>	<b>6473<math>\pm</math>2001</b>	<b>5709<math>\pm</math>744</b>	2795 $\pm$ 446	<b>2115<math>\pm</math>466</b>

671 **G A comparison with DOMINO and CaDM**

672 In this section, we give a brief comparison between our methods and methods from DOMINO [Mu  
673 et al., 2022] and CaDM [Lee et al., 2020] in the MuJoCo benchmark because we are using the exact  
674 same environmental setting (shown in Table 5).

675 DOMINO [Mu et al., 2022] is based on the InfoNCE  $K$ -sample estimator. Their implementation,  
676 PPO+DOMINO, is a model-free RL algorithm with a pre-trained context encoder. This encoder  
677 reduces the demand for large contrastive batch sizes during training by decoupling representation  
678 learning for each modality, simplifying tasks while leveraging shared information. However, a  
679 pre-trained encoder necessitates a large sample volume, with DOMINO training PPO agents for 5  
680 million timesteps on the MuJoCo benchmark. In contrast, SaTESAC and SaCCM, trained for 1.6  
681 million timesteps without pre-trained encoders, achieve considerably higher average returns across  
682 four environments (Table 6). Therefore, it is crucial to focus on extracting mutual information in  
683 contrastive learning that directly optimises downstream tasks, integrating rather than segregating  
684 representation learning from task performance.

685 CaDM [Lee et al., 2020] proposes a context-aware dynamics model adaptable to changes in dynamics.  
686 Specifically, they utilise contrastive learning to learn context embeddings, and then predict the next  
687 state conditioned on them. We copy the numerical results of PPO+CaDM, Vanilla+CaDM, and  
688 PE-TS+CaDM from CaDM [Lee et al., 2020] as their environmental setting is identical to ours, where  
689 PPO+CaDM is a model-free RL algorithm, while Vanilla+CaDM and PE-TS+CaDM are model-  
690 based. The model-free RL approach, PPO+CaDM, is trained for 5 million timesteps on the MuJoCo  
691 benchmark. As shown in Table 6, SaTESAC and SaCCM significantly outperform PPO+CaDM.  
692 The model-based RL algorithms, Vanilla+CaDM and PE-TS+CaDM, require 2 million timesteps  
693 for learning in model-based setups, compared to our fewer samples (i.e., million timesteps). In the  
694 Ant environment, Vanilla+CaDM and PE-TS+CaDM achieve higher returns than SaTESAC and  
695 SaCCM; similarly, in the Half-cheetah environment, PE-TS+CaDM outperforms them. Results in the  
696 SlimHumanoid and Crippled Half-cheetah environments show that skill-aware context embeddings  
697 are notably effective. An insight here is that our method outperforms the model-free CaCM approach,  
698 but not the model-based one. This is consistent with what is empirically found in CaDM [Lee  
699 et al., 2020]: prediction models are more effective when the transition function changes across tasks.  
700 Therefore, we consider that a model-based approach to SaMI could be an interesting extension for  
701 future work.

Table 7: The p-value of the statistical hypothesis tests (paired t-tests) for comparing the effectiveness of SaMI in MuJoCo benchmark (over 5 seeds). \* next to the number means that the algorithm with SaMI has statistically significant improvement over the same algorithm without SaMI at a significance level of 0.05. The “SaTESAC-TESAC” row indicates the p-value for the return improvement brought by SaMI to TESAC; the “SaCCM-CCM” row indicates the p-value for the return improvement brought by SaMI to CCM.

Crippled Ant			Crippled Half-cheetah			
	Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
SaTESAC-TESAC	0.121	0.109	9.54E-07*	0.154	0.0024*	0.0889
SaCCM-CCM	0.913	0.108	0.008*	0.04*	0.307	0.106
Ant			Half-cheetah			
	Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
SaTESAC-TESAC	0.136	0.034*	0.021*	0.346	0.224	0.004*
SaCCM-CCM	0.138	0.275	0.163	0.73	0.791	0.005*
SlimHumanoid			HumanoidStandup			
	Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
SaTESAC-TESAC	0.139	0.456	0.006*	0.037*	0.129	0.003*
SaCCM-CCM	0.113	0.059	0.008*	0.048	0.027*	0.01
Hopper			Crippled Hopper			
	Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
SaTESAC-TESAC	0.747	0.089	0.707	0.459	0.69	0.088
SaCCM-CCM	0.52	0.599	0.969	0.967	0.897	0.0002*
Walker			Crippled Walker			
	Training	Test (moderate)	Test (extreme)	Training	Test (moderate)	Test (extreme)
SaTESAC-TESAC	0.312	0.082	0.003*	0.223	0.048*	0.028*
SaCCM-CCM	0.55	0.541	0.022*	0.794	0.079	0.011*

Table 8: The p-value of the statistical hypothesis tests (paired t-tests) for comparing the effectiveness of SaMI in Panda-gym benchmark (over 5 seeds). \* next to the number means that the algorithm with SaMI has statistically significant improvement over the same algorithm without SaMI at a significance level of 0.05. The “SaTESAC-TESAC” row indicates the p-value for the return improvement brought by SaMI to TESAC; the “SaCCM-CCM” row indicates the p-value for the return improvement brought by SaMI to CCM.

	Training	Test (moderate)	Test (extreme)
SaTESAC-TESAC	0.000260*	0.000160*	0.001310*
SaCCM-CCM	0.000230*	0.002190*	0.001390*

## 702 H Statistical hypothesis tests (paired t-tests)

703 We used a t-test [Rice and Rice, 2007] to conduct a statistical hypothesis test to determine whether  
 704 SaMI brought a statistically significant improvement. we reported the p-value of the t-test in MuJoCo  
 705 (Table 7) and Panda-gym (Table 8) benchmarks. \* next to the number is used to indicate that the  
 706 algorithm with SaMI has statistically significant improvement over the same algorithm without SaMI  
 707 at a significance level of 0.05. From Table 7 and 8, SaMI brings significant improvement on the  
 708 extreme test set in which the RL agent needs to execute diverse skills. The statistically significant test  
 709 aligns with our results in the skill analysis (i.e., video demos, t-SNE and UMAP visualisation). In  
 710 complex environments that require high skill diversity from the RL agent, the statistically significant  
 711 improvement and higher returns/success rates brought by SaMI are evident.