Integrating Contrastive Learning with Dynamic Models for Reinforcement Learning from Images

Bang You[a,b,*], Oleg Arenz[b], Youping Chen[a], Jan Peters[b]

[a]*School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China*
[b]*Intelligent Autonomous Systems Lab, Technische Universität Darmstadt, Darmstadt 64289, Germany*

Abstract

Recent methods for reinforcement learning from images use auxiliary tasks to learn image features that are used by the agent's policy or Q-function. In particular, methods based on contrastive learning that induce linearity of the latent dynamics or invariance to data augmentation have been shown to greatly improve the sample efficiency of the reinforcement learning algorithm and the generalizability of the learned embedding. We further argue, that explicitly improving Markovianity of the learned embedding is desirable and propose a self-supervised representation learning method which integrates contrastive learning with dynamic models to synergistically combine these three objectives: (1) We maximize the InfoNCE bound on the mutual information between the state- and action-embedding and the embedding of the next state to induce a linearly predictive embedding without explicitly learning a linear transition model, (2) we further improve Markovianity of the learned embedding by explicitly learning a non-linear transition model using regression, and (3) we maximize the mutual information between the two nonlinear predictions of the next embeddings based on the current action and two independent augmentations of the current state, which naturally induces transformation invariance not only for the state embedding, but also for the nonlinear transition model. Experimental evaluation on the Deepmind control suite shows that our proposed method achieves higher sample efficiency and better generalization than state-of-art methods based on contrastive learning or reconstruction.

Keywords: Deep learning in robotics and automation, Reinforcement learning, Contrastive learning, Sensor-based control

---

[*]Corresponding author

   Email addresses: `youbang@hust.edu.cn` (Bang You), `oleg.arenz@tu-darmstadt.de` (Oleg Arenz), `ypchen@mail.hust.edu.cn` (Youping Chen), `jan.peters@tu-darmstadt.de` (Jan Peters)

1. Introduction

Deep reinforcement learning (RL) is a promising framework for enabling robots to perform complex control tasks from high-dimensional sensory inputs in unstructured environments, including household chores [1], manufacturing [2], and transportation [3]. Specifically, end-to-end reinforcement learning from images enables robots to learn new skills without relying on object-specific detection and tracking systems. However, by operating on high-dimensional observation spaces that are typically governed by complex dynamics, reinforcement learning from images often requires many environment interactions to learn successful policies, which is not possible for a wide range of real robotic tasks.

Hafner et al. [4, 5] substantially improved the sample efficiency of image-based reinforcement learning, by learning a predictive model on a learned latent embedding of the state. They alternately optimize a policy based on the learned models of the latent reward and transition distributions, and improve their models based on newly collected samples from the learned policy. However, they still need many samples to learn a model of the environment. Furthermore, the learned world model is usually inaccurate, and the resulting prediction errors lead to suboptimal policies.

Another solution to improve sample efficiency for reinforcement learning from images is to use auxiliary tasks to learn compact state representations, which can be used by the reinforcement learning agent. One common auxiliary task is to reconstruct raw images with autoencoders and its variations [6, 7, 8]. However, since such approaches learn a representation by minimizing the reconstruction error in the pixel space, they try to capture pixel-level details even when they are task-irrelevant, which can degrade sample efficiency [9].

Recently, contrastive learning objectives, which rely on mutual information estimation, have been shown to improve sample efficiency of reinforcement learning algorithms. In particular, two different types of contrastive learning have been shown to be effective for reinforcement learning from images. The first line of research [10] maximizes the mutual information between two independent transformations of the same image, which increases the robustness of representations similar to data augmentation. The second line of research [11, 12], maximizes mutual information between consecutive states, aiming to learn representations that have approximately linear latent dynamics.

We further argue that improving latent Markovianity—that is, maximizing the predictability of the next embedded state based on the current embedded state and action—is important, since reinforcement learning assumes Markovian states and actions and an agent using non-Markovian representations is, thus, not able to exploit non-Markovian effects. Although mutual information based approaches that maximize predictive information [13] already improve Markovianity, we argue that it is more effective to explicitly minimize the prediction errors of a learned model of the latent dynamics.

Based on the three desired properties that we identified, we propose a self-supervised representation learning method that integrates **co**ntrastive learning with **dy**namic models, CoDy, for reinforcement learning from images. Our

method consists of three auxiliary tasks on the learned embedding. Namely, (1) we minimize the prediction error between the current state- and action-embeddings and the true embedding of the next state to increase Markovianity, (2) we maximize the InfoNCE [12] bound on the temporal mutual information between the current state- and action-embeddings and the true embedding of the next state to increase linearity of the latent dynamics, and (3) we maximize the multi-view mutual information between the predicted embeddings at the next time step for two independent data augmentations to improve invariance to data augmentations.

Compared to aforementioned contrastive learning methods, our proposed method offers three appealing properties. Firstly, our method effectively improves Markovianity of the latent embedding by explicitly learning a nonlinear transition models. Secondly, instead of directly maximizing the mutual information of the augmented images, we maximize the multi-view mutual information between the predicted embeddings of next states, which additionally encourages the latent transition model to be invariant to data augmentations. Thirdly, our mutual information objectives take into account the actions such that the state representation does not implicitly depend on the actions in the replay buffer.

We train our auxiliary tasks with a standard soft actor-critic [14] reinforcement learning agent for learning continuous policies from images. The main contributions of our work are as follows.

- To improve the Markovianity of state embeddings, we propose a self-supervised representation learning method that combines contrastive learning with a nonlinear prediction task. Our method learns state embeddings while inducing Markovianity, transformation invariance, and linearity of latent dynamics.

- We propose a novel multi-view mutual information objective that maximizes the agreement between the *predicted* embeddings of the next states for different transformations of the current state, which induces transformation invariance not only for the state embeddings, but also for the latent dynamics.

- We evaluate our method on a set of challenging image-based benchmark tasks and show that it achieves better sample efficiency and generalization than state-of-art reconstruction-based, contrastive-learning-based and model-based methods.

The remainder of the paper is organized as follows. We present the problem statement and preliminaries in Section 1.1. In Section 2 we discuss previous work related to state representation learning. We present the proposed framework and the auxiliary tasks in Section 3. Section 4 contains details on our implementation of the proposed algorithm and the results of the experimental evaluation. In Section 5 we draw a conclusion and discuss limitations and future work.

1.1. Problem Statement and Preliminaries

We describe the problem of learning continuous control policies from high-dimensional observations. Our algorithm is built on top of soft actor critic (SAC) [14], which is a model-free off-policy reinforcement learning algorithm with entropy regularization. We also introduce contrastive predictive coding [12] used for mutual information estimation.

1.1.1. Problem Statement and Notation

We formulate the problem of learning continuous control policies from images as an infinite-horizon Markov decision process (MDP). An MDP can be formulated by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S}$ is the state space, and $\mathcal{A}$ is the action space, $P(s_{t+1}|s_t, a_t)$ is a stochastic dynamic model, $r(s, a)$ is the reward function and γ the discount factor. The state and action space fulfill the (first-order) Markov property, that is, the distribution of the next state s_{t+1} is conditionally independent of all prior states and actions, $s_{t'<t}$ and $a_{t'<t}$, given the current state s_t and action a_t. At every time step t, the agent observes the current state and chooses its action based on its stochastic policy $\pi(a_t|s_t)$ and obtains a reward $r(s_t, a_t)$. Our goal is to optimize the policy to maximize the agent's expected cumulative reward.

We specifically focus on image-based reinforcement learning, that is, the state space is provided in terms of images. A single image is usually not Markovian, since it contains little information about object velocities (which could be estimated using previous images). Following common practice [15] in reinforcement learning from images, we stack the k most recent images together and define the state as $s_t = (o_t, o_{t-1}, o_{t-2}, \cdots, o_{t-k+1})$, where o_t is the image observed at time t. While this problem could also be framed as a partially observable MDP (POMDP), please note that we assume that the agent observes s_t, which is assumed to be a Markovian state. Hence, in contrast to POMDP methods, we neither need to use previous observations $s_{t'<t}$ to better estimate an unobserved hidden state, nor do we need to learn a policy that actively chooses informative actions. Instead, we focus on representation learning, that is, we want to learn an embedding $\phi_\alpha : \mathcal{S} \to \mathbb{R}^d$, parameterized by α, that maps the high-dimensional state s to a lower-dimensional representation $z = \phi_\alpha(s)$ to increase sample efficiency. Along with the embedding ϕ_α, we want to learn a policy $\pi(a|s) = \pi(a|\phi_\alpha(s))$ that maximizes the expected cumulative rewards

$$J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_t \middle| a_t \sim \pi(\cdot|z_t), s_{t+1} \sim P(\cdot|s_t, a_t), s_0 \sim p_0(s_0) \right], \quad (1)$$

where $p_0(s_0)$ is the distribution of the initial state and $\gamma \in (0, 1)$ is a discount factor to ensure finite returns.

1.1.2. Maximum Entropy Reinforcement Learning

Maximum entropy reinforcement learning optimizes a policy to maximize the sum of the expected cumulative rewards and the expected entropy of the

policy [16]. Unlike standard reinforcement learning, in the maximum entropy reinforcement learning framework the agent gets an additional reward that is proportional to the entropy, $H(\pi(\cdot|s_t) = -\int_{\mathcal{A}} \pi(\cdot|s_t) \log(\pi(\cdot|s_t)))\, \mathrm{d}a$, of the policy at every time step to encourage stochasticity. Soft actor-critic [14] is an off-policy maximum entropy reinforcement learning algorithm that learns an entropy-regularized stochastic policy π_ω parameterized by ω, and two Q functions Q_{σ_1} and Q_{σ_2} with parameters σ_1 and σ_2 respectively, to find an optimal control policy. The soft Q function can be learned by minimizing the Bellman error,

$$L_Q(\sigma_i) = \mathbb{E}_{(s,a,r,s',d)\sim D}\left[(Q_{\sigma_i}(s,a) - \mathbb{T}(r,s',d)))\right], \tag{2}$$

where state s, action a, reward r, next state s' and a termination flag d are sampled from the replay buffer D. The target value $\mathbb{T}(r,s',d)$ is computed from the Q function

$$\mathbb{T}(r,s',d) = r + \gamma(1-d)\Big(\min_{i=1,2} Q_{\sigma_{\mathrm{targ,i}}}(s',a') - \alpha \log \pi_\omega(a'|s')\Big), \tag{3}$$

where $a' \sim \pi_\omega(\cdot|s')$, and $Q_{\sigma_{\mathrm{targ,i}}}$ denotes the target Q function, which uses an exponential moving average of the parameters σ_i for $i = 1, 2$. This particular parameter update has been shown to stabilize training.

The policy can be optimized by minimizing

$$L_\pi(\omega) = \mathbb{E}_{s,a\sim D,\pi}[\alpha \log \pi_\omega(a|s) - Q(s,a)], \tag{4}$$

where the states s and actions a are sampled from the replay buffer and the stochastic policy, respectively. $Q(s,a) = \min_{i=1,2} Q_{\sigma_i}$ is the minimum of both Q function approximators.

1.1.3. Contrastive Predictive Coding

For two random variables x_1 and x_2, the mutual information $I(x_1, x_2)$ is defined as the Kullback-Leibler divergence between the joint distribution $p(x_1, x_2)$ and the product of the marginal distributions $p(x_1)p(x_2)$,

$$I(x_1, x_2) = \int_{\mathbf{x_1},\mathbf{x_2}} p(x_1, x_2) \log \frac{p(x_1, x_2)}{p(x_1)p(x_2)} \mathrm{d}x_1 \mathrm{d}x_2,$$

which is typically intractable to compute.

Contrastive predictive coding [12] introduced the InfoNCE loss for maximizing a lower bound on the mutual information between the current embedding and a future embedding. Specifically, given an anchor $x_{1,i}$, a positive sample $x_{2,i} \sim p(x_2|x_{1,i})$ and K negative samples $x_{2,j} \sim p(x_2)$, the InfoNCE loss corresponds to a binary cross-entropy loss for discriminating samples from the joint distribution $p(x_{1,i})p(x_2|x_{1,i})$ from samples from the marginals, that is,

$$L_{\mathrm{NCE}} = -\mathbb{E}_{p(x_1,x_2)}\left[\log \frac{exp(f(x_{1,i}, x_{2,i}))}{\sum_{j=1}^{K} exp(f(x_{1,i}, x_{2,j}))}\right], \tag{5}$$

where the learnable function $f(\cdot, \cdot)$ measures the similarity between the anchor and samples.

Minimizing the InfoNCE loss is equivalent to maximizing a lower bound on the mutual information between x_1 and x_2,

$$I(x_1, x_2) \geq \log(K) - L_{\mathrm{NCE}}. \tag{6}$$

Contrastive predictive coding uses a recurrent neural network to extract a context m_t from the embeddings of previous observation, and maximizes the InfoNCE bound of the mutual information between the current context and the k-step future state embedding z_{t+k}. Oord et al. [12] propose a simple log-bilinear model for the classifier, that is,

$$f(m_t, x_{t+k}) = \exp\left(m_t^\top W \phi(x_{t+k})\right),$$

where ϕ is the encoder. The embedding can thus be learned by minimizing the InfoNCE loss both with respect to the parameters of the encoder ϕ, and with respect to the matrix W which parameterizes f. By maximizing the inner product between the context m_t and a linear transformation of the future embedding $W\phi(x_{t+k})$, the InfoNCE loss with log-bilinear classifier favors linearly predictive embeddings [11], although the mutual information objective does not impose linearity *perse*.

2. Related Prior Work

Learning expressive state representations is an active area of research in robot control and reinforcement learning. We will now briefly discuss robotic-prior based and reconstruction based approaches, but will focus on methods based on mutual information, which are most related to our work.

2.1. Robotic Prior-based Methods

Prior work in this area has explored using prior knowledge about the world or dynamics, called robotic priors, for learning state representations [17, 18]. These robotic priors can be defined as loss functions to be minimized without additional semantic labels. Jonschkowski and Brock [17] use prior knowledge, such as temporal continuity or causality, to learn representations that improve the performance of reinforcement learning. Lesort et al. [19] propose to stabilize the learned representation by forcing two states corresponding to the same reference point to be close to each other. For example, the fixed starting position of the arm can be used as a reference point, which acts as a reference or calibration coordinate. Morik et al. [20] combine the idea of robotic priors with LSTMs to learn task relevant state representations, which have been shown to be robust to states that are incorrectly classified as being close to each other. However, these robotic prior based approaches often rely on significant expert knowledge and hence limit the generalizability of the learned representations across tasks.

2.2. Reconstruction-based Approaches

Many existing approaches use a reconstruction loss to learn a mapping from observations to state representations [6, 21]. Reconstruction-based methods aim to reconstruct their input under constraints on their latent representations, e.g. their dimensionality. Autoencoders can be used to encode high-dimensional inputs into a low-dimensional latent state space [6]. In order to improve training stability, Yarats et al. [8] incorporate variational autoencoders into an off-policy learning algorithm. However, there is no guarantee that the learned representation will encode useful information for the task at hand. To alleviate this problem, constraints on the latent dynamics have been proved effective in learning useful task-oriented representations [7, 22, 23]. These methods encourage the encoder to capture information necessary to predict the next state. However, since they extract representations by minimizing the reconstruction error in pixel space, these reconstruction-based methods aim to capture the full diversity of the environment, even if it is irrelevant to the task.

2.3. Mutual Information Based Approaches

Recent literature on unsupervised representation learning focuses on extracting latent embeddings by maximizing different lower bounds on the mutual information between the representations and the inputs. Commonly used bounds include MINE [24], which estimates mutual information based on the Donsker-Varadhan representation [25] of the Kullback-Leibler divergence, and InfoNCE [12], which uses a multi-sample version of noise-contrastive estimation [26].

Hjelm et al. [27] maximize mutual information between the input and its representations. They propose to maximize both, this global mutual information and local mutual information that considers small patches of the input. The local mutual information objective should encourage the encoder to capture features shared by different patches, and thus put less focus on pixel-level noise. Bengio et al. [28] considered simple reinforcement learning problems, and propose to learn independently controllable features, by learning a separate policy for every dimension of the embedding that only affects variation of that respective dimension. Anand et al. [11] proposed to learn state representations by maximizing the mutual information of observations across spatial and temporal axes. Their key idea is to improve temporal coherence of observations. However, they do not take into account the action for their temporal mutual information objective and hence the predictive information is specific to actions that were used in the training set.

More recently, several works apply mutual information objectives for representation learning in a deep reinforcement learning setting. Oord et al. [12] introduced the InfoNCE bound on the mutual information and already evaluated it in a reinforcement learning setting. They maximize the mutual information between the current context, which is computed from all previous state embeddings, and the embedding several time steps in the future. Laskin et al. [10] learn useful state representations by maximizing the mutual information

between the features of independent transformations of the same observations, improving transformation invariance of the learned embedding. The importance of data augmentation for deep reinforcement learning from images has also been stressed by Yarats et al. [29]. Laskin et al. [10] generate transformed images for data augmentation, where a random patch is cropped from a stack of temporally sequential frames sampled from the replay buffer. Moreover, they use separate online and target encoders for the anchor, and positive/negative samples, respectively, when computing the InfoNCE loss on the mutual information, rather than using same encoder for the anchor, and positive/negative samples, which was proposed by Oord et al. [12]. Our method shares several aspects with their method, CURL, since we also apply the InfoNCE [12] bound to maximize the mutual information between two embeddings that result from different image crops in one of our auxiliary tasks. However, we do not consider the embeddings directly, but the predicted embeddings at the next time step, additionally targeting latent dynamics invariant to the transformation of the current embedding. Lee et al. [13] apply the conditional entropy bottleneck [30] to maximize the mutual information between the current embedding and the future state and reward, while compressing away information that is not also contained in the next state or reward. Their conditional entropy bottleneck objective is conditioned on multiple future states and rewards for multi-step prediction. Instead, we propose to compress away task-irrelevant information by using data augmentation without compromising the ability to predict the *embedding* of the next state.

Model-based reinforcement learning methods iteratively build a predictive model of the environment from high-dimensional images along with a policy based on that model [5, 31, 32, 33]. For example, Dreamer [4] learns models of the latent dynamics and rewards and uses them for reinforcement learning in the latent MDP. By iteratively applying the policy on the real system, new data is collected to improve the models. Some recent model-free reinforcement learning approaches learn latent dynamic models to compute intrinsic reward for solving many reinforcement learning tasks with sparse rewards [34, 35]. For instance, Li et al. [35] compute intrinsic rewards by estimating the novelty of the next state based on the prediction error of a dynamic model that is smoothly updated during training. Learning predictive dynamic models has also been shown to be promising for many robotic control tasks. For example, Li et al. [36] propose a neural fuzzy-based dynamic model for reliable trajectory tracking, which effectively facilitates the control performance of the wheel-legged robot. We also learn a model of the latent dynamics, but we use it only for learning better representations. However, making better use of the learned models is a natural extension for future work.

In this paper, we propose an approach for representation learning in the context of deep reinforcement learning, that is based on mutual information, without relying on robotic prior knowledge or pixel-reconstruction. Most aforementioned mutual information based methods either improve temporal coherence of observations or maximize the similarity of two independently augmented images, which cannot effectively guarantee the Markovianity of state and action
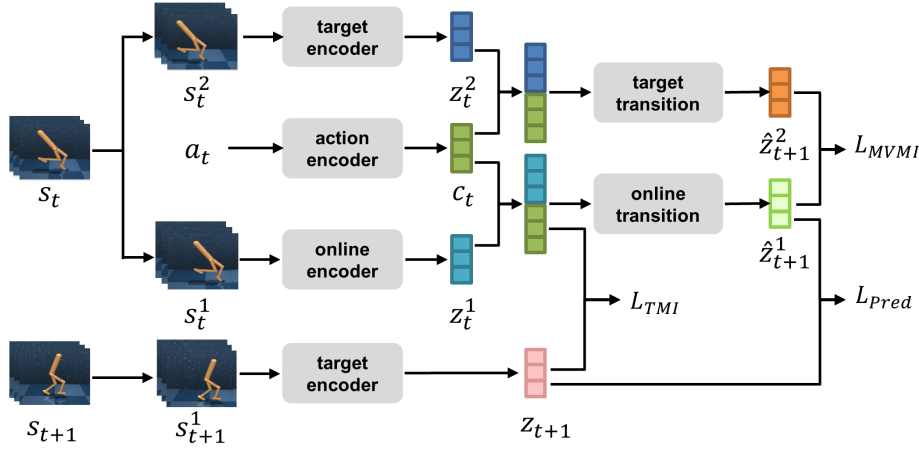
Figure 1: Our framework contains three auxiliary tasks and respective loss functions. The temporal mutual information loss L_{TMI} aims to maximize the InfoNCE bound of the mutual information between the current state-action embedding and the next state-embedding, that is, $I([c_t, z_t^1], z_{t+1})$. The prediction loss L_{Pred} is given by the squared ℓ_2 error between the predicted next embedding and the actual embedding of next state. The multi-view mutual information loss uses the InfoNCE bound to maximize the mutual information between the predicted next embeddings based on two different state augmentation s_t^2 and s_t^1, $I(\hat{z}_{t+1}^1, \hat{z}_{t+1}^2)$. The parameters of the target encoder and target transition function are an exponential moving average of the parameters of the online models. The online encoder is used by the Q-functions and the policy of the soft actor-critic agent, which is trained along with the auxiliary tasks.

embeddings. However, non-Markovian embeddings can make it harder to learn an optimal policy or Q-function, since reinforcement learning agents assume their states and actions to be Markovian. To alleviate this problem, our proposed method imposes a nonlinear dynamic model on the latent state space. Moreover, instead of directly maximizing the agreement of embedding of augmented images, we propose to maximize the mutual information between the predicted embedding at the next time step, additionally improving data augmentation invariance of the latent dynamic models. The comparative experiments presented in Section 4 have shown that the proposed method outperforms leading reconstruction-based and contrastive-based methods on typically challenging image-based benchmark tasks.

3. Integrating Contrastive Learning with Dynamic Models

We will now present the architecture of the proposed framework and our method, which consists of three auxiliary tasks, in detail. We will also show how to train the representations together with the policy and the Q-function.

3.1. The Mechanism and Architecture for Mutual Information Maximization

Our mutual information maximization architecture is presented in Figure 1. We apply stochastic data augmentation to raw states s_t in order to obtain

9

two independent views (image crops) of the current states, $s_t^1 \sim A(s_t)$ and $s_t^2 \sim A(s_t)$, where $A(\cdot)$ denotes the distribution of augmented images. In order to extract compact representations from states, we use an online encoder $\phi_\alpha : \mathcal{S} \to \mathbb{R}^d$ with parameters α and a target encoder $\phi_\beta : \mathcal{S} \to \mathbb{R}^d$ with parameters β to transform augmented observations s_t^1 and s_t^2 into representations z_t^1 and z_t^2, respectively. An action encoder $\psi_\gamma : \mathcal{A} \to \mathbb{R}^n$ with parameters γ maps actions a_t into a feature vector c_t. Finally, the representations of states and actions are concatenated together. An online transition model g_υ with parameters υ and a target transition model g_μ with parameters μ (e.g., neural networks) predict the representation of the next state based on a given state- and action-embedding,

$$\hat{z}_{t+1}^1 = g_\upsilon(z_t^1, c_t), \tag{7}$$

$$\hat{z}_{t+1}^2 = g_\mu(z_t^2, c_t). \tag{8}$$

At the next timestep, the target encoder maps the state s_{t+1} into the latent representations z_{t+1}. Motivated by He et al. [37], the parameters of the target state encoder and the target transition model are computed as an exponential moving average (EMA) of the parameters of the online encoder and online transition model, respectively,

$$\beta = \tau\alpha + (1 - \tau)\beta, \tag{9}$$

$$\mu = \tau\upsilon + (1 - \tau)\mu, \tag{10}$$

where $\tau \in [0, 1)$ is the coefficient of the exponential moving average.

3.2. A Prediction Task for Improving Latent Markovianity

The Markov assumption is critical in reinforcement learning and states that the distribution over the next state is conditionally independent of all previous states and actions, given the current state and action. When the Markov assumption is violated, the learned policy and value function in general cannot make use of the environmental information that is contained in the previous state but not in the current one. By providing the state embedding z instead of the original state s as input to the agent's policy and Q-function, we effectively change the state space from the agent's perspective.

Although the original state space is assumed Markovian, the learned state representation is not Markovian in general. For example, consider a simple linear system, where the state space is given by the position and velocity of a point mass, and the action corresponds to a bounded change of acceleration. While the state space is Markovian, a representation that discards the velocity is no longer Markovian, since the position at time step $t - 1$ can improve our estimate of the current velocity and thus the prediction of the next position.

However, we argue that a maximally compressed embedding of a Markovian state, that keeps sufficient information for learning an optimal policy and Q-function, should also be Markovian, which we can prove by contradiction: Assume a non-Markovian embedding of a Markovian state that is maximally

compressed while keeping all information that is useful for predicting the expected cumulative reward. As the current state—due to Markovianity of the state space—contains all information for predicting the next state, and thus also for predicting the next embedded state, a non-Markovian state representation necessarily discards information that would be useful for predicting the next embedded state. Hence, either we lost information that would be useful for predicting the expected cumulative reward, or the state representation contains information that is unnecessary for predicting the expected cumulative reward, and, thus, not maximally compressed.

While latent Markovianity is a necessary condition for learning a concise and sufficient representation of the state, it is clearly not sufficient, since even a constant embedding, $\phi(s) = \text{const}$, is Markovian. However, we hypothesize that introducing an auxiliary objective to induce Markovianity improves the effectiveness of the learned embedding by improving its consistency. Strict enforcement of latent Markovianity seems challenging, but we can improve latent Markovianity in the sense that we reduce the predictive information about the next embedding that was present in previous state embeddings but not in the current one. We hypothesize that improving Markovianity by means of an auxiliary prediction tasks, improves the consistency and thereby the sample efficiency and generalizability for the learned embedding. This hypothesis is consistent with the experimental results of Lee et al. [13] and Anand et al. [11], where auxiliary prediction tasks were shown to improve the effectiveness of the learned embedding. We improve Markovianity by introducing the auxiliary task of predicting the next state embedding z_{t+1} based on the embeddings of the current state and action, z_t^1 and c_t and a learned non-linear transition model g_v. The prediction error is defined as

$$L_{\text{pred}}(\alpha, \gamma, v) = \left\| z_{t+1}^1 - \hat{z}_{t+1}^1 \right\|_2^2, \tag{11}$$

with the squared ℓ_2 norm denoted by $\|\cdot\|_2^2$ and the prediction $\hat{z}_{t+1}^1$. By minimizing this forward prediction error, the transition model forces the state encoder to learn predictive features.

3.3. A Temporal Mutual Information Based Prediction Task

Although minimizing the prediction error as discussed in Section 3.2 should be more effective in improving Markovianity compared to maximizing a bound on the mutual information between embeddings of consecutive time steps, the latter approach has been shown to be very effective [13, 11]. The better performance of methods that maximize predictive information using the InfoNCE bound with log-bilinear classifiers, may be caused by implicitly inducing linearly predictive representations [11], or by the fact that these approaches only rely on a discriminative model [38]. Hence, we propose to use both auxiliary tasks, the prediction task discussed in Section 3.2 for inducing Markovianity more strongly, and a task based on mutual information, which induces linearly predictive representations without relying on a generative model. Namely, we optimize the state encoder ϕ_α and action encoder ψ_γ to maximize the temporal

11

mutual information, between the current state- and action-embeddings and the next state-embedding, that is,

$$\max_{\alpha,\gamma} I([z_t^1, c_t], z_{t+1}) = \max_{\alpha,\gamma} \mathbb{E}\left[\log \frac{p(z_t^1, c_t, z_{t+1})}{p(z_t^1, c_t)p(z_{t+1})}\right], \quad (12)$$

where p denotes the joint distribution of these variables, as well as their associated marginal distributions.

We maximize the temporal mutual information by minimizing the InfoNCE loss (Eq. 5), and use a log-bilinear classifier to induce linearity. More specifically, let (z_t^1, c_t, z_{t+1}) denote samples drawn from the joint distribution $p(z_t^1, c_t, z_{t+1})$ which we refer to as positive sample pairs, and N_1 denotes a set of negative samples sampled from the marginal distribution $p(z_{t+1})$. The InfoNCE loss for maximizing the lower bound of this temporal mutual information is given by

$$L_{\text{TMI}}(\alpha, \gamma) = -\mathbb{E}_{p(z_t^1, c_t, z_{t+1})}\left[\mathbb{E}_{N_1}\left[\log \frac{h_1(z_t^1, c_t, z_{t+1})}{\sum_{z_{t+1}^* \in N_1 \cup z_{t+1}} h_1(z_t^1, c_t, z_{t+1}^*)}\right]\right], \quad (13)$$

where $h_1(z_t^1, c_t, z_{t+1})$ is a log-bilinear score function given by

$$h_1(z_t^1, c_t, z_{t+1}) = \exp\left(m(z_t^1, c_t)^\top W_1 z_{t+1}\right). \quad (14)$$

Here, the function $m(\cdot, \cdot)$ concatenates the representations of states and actions and W_1 is a learned matrix. The term $m(z_t^1, c_t)W_1$ in Equation 14 performs a linear transformation to predict the next state representations z_{t+1}, which forces the encoders of states and actions to capture linearly predictive representations.

In practice, we randomly draw a minibatch of state-action pairs and corresponding next states (s_t, a_t, s_{t+1}) from the replay buffer. We obtain a minibatch of positive sample pairs (z_t^1, c_t, z_{t+1}) by feeding s_t, a_t and s_{t+1} into their corresponding encoder, respectively. For a given positive sample pair (z_t^1, c_t, z_{t+1}), we construct N_1 by replacing z_{t+1} with all features z_{t+1}^* from other sample pairs $(z_t^{1*}, c_t^*, z_{t+1}^*)$ in the same minibatch.

3.4. A Mutual Information Based Task for Improving Transformation Invariance

The temporal mutual information objective presented in Section 3.3 can encourage the encoder of states and actions to capture task-irrelevant information which may help in discriminating the positive and negative samples in the InfoNCE loss (Eq. 13). Lee et al. [13] propose to learn a more compressed representation by applying the conditional entropy bottleneck, which additionally punishes the conditional mutual information between the past states and the embedding, conditioned on the future. However, in general, information that is no longer available at the next time step might still be relevant for the agent at the current step. Instead, we use data augmentation to add an inductive bias on which information is not relevant for the agent, which has been shown crucial by Yarats et al. [29].

12

For example, some lower-level factors presented in the observed states and actions, including noise or external perturbations, are typically not useful for the agent. For our experiments, we consider a multi-view setting, inspired by Laskin et al. [10], where we assume each view of the same scene shares the same task-relevant information while the information not shared by them is task-irrelevant. Intuitively, this assumption represents an inductive bias that the way we see the same scene should not affect the internal state of the environment. Instead, the representations should extract task-relevant information that is shared by different views for predicting the next embedding. Hence, we propose to maximize the multi-view mutual information $I(\hat{z}_{t+1}^1, \hat{z}_{t+1}^2)$ between the predicted representation of next states $\hat{z}_{t+1}^1$ and $\hat{z}_{t+1}^2$, based on independent views s_t^1 and s_t^2 of the same scene with respect to the state embedding ϕ_α, the action embedding ψ_γ and the nonlinear transition model g_υ,

$$\max_{\alpha,\gamma,\upsilon} I(\hat{z}_{t+1}^1, \hat{z}_{t+1}^2) = \max_{\alpha,\gamma,\upsilon} \mathbb{E}\left[\log \frac{p(\hat{z}_{t+1}^1, \hat{z}_{t+1}^2)}{p(\hat{z}_{t+1}^1)p(\hat{z}_{t+1}^2)} \right]. \tag{15}$$

Since mutual information measures the amount of information shared between random variables, this objective forces the encoder of states to capture transformation invariant information about higher-level factors (e.g. presence of certain objects) from multiple views of the same scene. Notably, instead of modelling the mutual information between the representations of two augmented states $I(z_t^1, z_t^2)$ [10], the mutual information objective that we consider imposes transformation invariance not only on the state embedding, but also on the transition model.

We employ InfoNCE to maximize a lower bound of the above multi-view mutual information objective. By $(\hat{z}_{t+1}^1, \hat{z}_{t+1}^2)$ we denote samples drawn from the joint distribution $p(\hat{z}_{t+1}^1, \hat{z}_{t+1}^2)$ which we refer to positive sample pairs, and by N_2 we denote a set of negative samples sampled from the marginal distribution $p(\hat{z}_{t+1}^2)$. The multi-view mutual information objective is thus given by

$$L_{\mathrm{MVMI}}(\alpha,\gamma,\upsilon) = - \mathbb{E}_{p(\hat{z}_{t+1}^1, \hat{z}_{t+1}^2)} \left[\mathbb{E}_{N_2} \left[\log \frac{h_2(\hat{z}_{t+1}^1, \hat{z}_{t+1}^2)}{\sum_{\hat{z}_{t+1}^{2*} \in N_2 \cup \hat{z}_{t+1}^2} h_2(\hat{z}_{t+1}^1, \hat{z}_{t+1}^{2*})} \right] \right], \quad (16)$$

with score function $h_2(\cdot, \cdot)$ which maps feature pairs onto scalar-valued scores. We employ a log-bilinear model,

$$h_2(\hat{z}_{t+1}^1, \hat{z}_{t+1}^2) = \exp\left(\hat{z}_{t+1}^1 W_2 \hat{z}_{t+1}^2 \right), \tag{17}$$

with weight transformation matrix W_2. Minimizing L_{MVMI} with respect to ϕ_α, ψ_γ, g_υ and W_2 maximizes the mutual information between the predicted representations of next states. In practice, we randomly sample a minibatch of state-action pairs and corresponding next states (s_t, a_t, s_{t+1}) from the replay buffer. We obtain positive sample pairs $(\hat{z}_{t+1}^1, \hat{z}_{t+1}^2)$ by feeding the above minibatch into our mutual information framework. For a given positive sample pair $(\hat{z}_{t+1}^1, \hat{z}_{t+1}^2)$, we construct N_2 by replacing $\hat{z}_{t+1}^2$ with all features $\hat{z}_{t+1}^{2*}$ from other sample pairs $(\hat{z}_{t+1}^{1*}, \hat{z}_{t+1}^{2*})$ in the same minibatch.

Finally, the total loss function L_{CoDy} for the auxiliary tasks consists of the prediction loss L_{pred}, the temporal mutual information loss L_{TMI} and multi-view mutual information loss L_{MVMI} weighted by hyperparameters λ and η

$$L_{\mathrm{CoDy}}(\alpha, \gamma, v) = L_{\mathrm{MVMI}}(\alpha, \gamma, v) + \lambda L_{\mathrm{TMI}}(\alpha, \gamma) + \eta L_{\mathrm{pred}}(\alpha, \gamma, v). \qquad (18)$$

The online encoder ϕ_α, the action encoder ψ_γ and online transition model g_v is optimized simultaneously by minimizing this total loss function.

Algorithm 1: Training Algorithm for CoDy

Require: parameters α, v, β, μ, γ, ω, σ, $\hat{\sigma}$, batch size B, replay buffer D, learning rates ρ_e, ρ_a and ρ_c

initialize replay buffer D

for *each training step* **do**

    collect experience (s_t, a_t, r_t, s_{t+1}) and add it to the replay buffer D

    for *each gradient step* **do**

        Sample a minibatch of tuple: $\{s_t, a_t, r_t, s_{t+1}\}_1^B \sim D$

        Update soft Q-function:

        $\{\sigma_i, \alpha\} \leftarrow \{\sigma_i, \alpha\} - \rho_c \hat{\nabla}_{\sigma_i} L_Q(\sigma_i)$ for $i \in \{1, 2\}$

        Update policy:

        $\omega \leftarrow \omega - \rho_a \hat{\nabla}_\omega L_\pi(\omega)$

        Update online encoder, online transition model and action encoder:

        $\{\alpha, v, \gamma\} \leftarrow \{\alpha, v, \gamma\} - \rho_e \hat{\nabla}_{\{\alpha, v, \gamma\}} L_{\mathrm{CoDy}}(\alpha, v, \gamma)$

        Update target Q-function:

        $\hat{\sigma}_i \leftarrow \tau \sigma_i + (1 - \tau_Q)\hat{\sigma}_i$ for $i \in \{1, 2\}$

        Update target encoder and transition model:

        $\beta = \tau \alpha + (1 - \tau_e)\beta$

        $\mu = \tau v + (1 - \tau_e)\mu$

    end

end

3.5. Joint Policy and Auxiliary Task Optimization

We train our auxiliary tasks jointly with SAC, a model-free off-policy reinforcement learning agent, by adding Eq. 18 as an auxiliary objective during policy training. The policy takes the representations computed with the online encoder to choose what action to take, and to approximate Q-values. Since the reward function can provide some task-relevant information, we allow the gradient of the Q-function to back-propagate through the online encoder in order to further capture task-relevant representations of observations. We do not backpropagate the actor loss through the embedding because this degrades performance by implicitly changing the Q-function during the actor update, as noted by Yarats et al. [8].

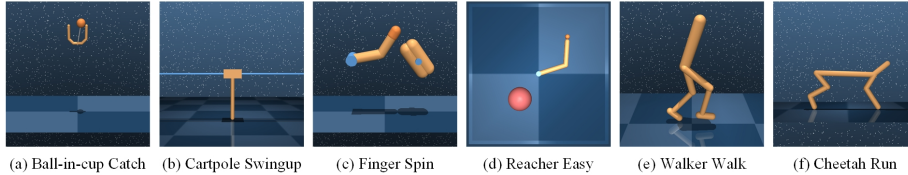(a) Ball-in-cup Catch (b) Cartpole Swingup (c) Finger Spin (d) Reacher Easy (e) Walker Walk (f) Cheetah Run

Figure 2: Continuous control tasks from the Deepmind control suite used in our experiments. (a) The Ball-in-cup Catch task only provides the agent with a sparse reward when the ball is caught. (b) The Cartpole Swingup task attains images from a fixed camera and, hence, the cart can move out of sight. (c) The Finger Spin task requires contacts between the finger and the object. (d) The Reacher Easy task has a sparse reward that is only given when the target location is reached. (e) The Walker Walk task has complex dynamics. (f) The Cheetah Run task has both high action dimensions and contacts with the ground.

The training procedure is presented in Algorithm 1 in detail. α and υ are the parameters of the online encoder ϕ_α and transition model g_υ, respectively. β and μ are the parameters of the target encoder ϕ_β and transition model g_μ, while γ are the parameters of the action encoder ψ_γ. The parameters of policy π, the Q-function and target Q-function are denoted as ω, σ and $\hat{\sigma}$, respectively. ρ_a and ρ_c are the learning rates for the policy and Q-function. ρ_e is the learning rate for the online encoder, online transition model and action encoder. The experience is stored in a replay buffer D, which is initialized with tuples $\{s_t, a_t, r_t, s_{t+1}\}$ by using a random policy. The algorithm proceeds by alternating between collecting new experience from the environment, and updating the parameters of the soft Q-function, policy and auxiliary prediction model. The parameters of the policy network and Q-function network are optimized by minimizing the SAC policy loss (Eq.4) and actor loss (Eq.2), respectively. The parameters of the online encoder, online transition model and action encoder are optimized by minimizing Eq. 18 jointly. The parameters of the target Q-function network are given by an exponential moving average of the parameters of the online Q-function.

4. Experimental Evaluation

We evaluate the data efficiency and performance of our method and compare it against state-of-the-art methods—both a model-based reinforcement learning method and model-free reinforcement learning methods with auxiliary tasks—on various benchmark tasks. We test the generalization of our method to unseen tasks and compare it with other methods. We will now describe the experimental setup and present the results. We will also show the results of an ablation study to show the effects of the different auxiliary tasks and a visualization of the learned representations. Our code is open-sourced and available at `https://github.com/BangYou01/Pytorch-CoDy`.

4.1. Experimental Setup

We implement the proposed algorithm on the commonly used PlaNet [5] test bed, which consist of a range of challenging imaged-based continuous control tasks (see Figure 2) from the Deepmind control suite [39]. Specifically, six tasks are considered: Ball-in-cup Catch, Cartpole Swingup, Reacher Easy, Finger Spin, Walker Walk and Cheetah Run. Every task offers a unique set of challenges, including sparse rewards, complex dynamics as well as contacts. We refer to Tassa et al. [39] for more detailed descriptions.

We parameterize the online encoder of states, the action encoder and the transition model using feed forward neural networks. The online encoder of observations consists of four convolution layers following a single fully-connected layer. We use a kernel of size 3×3 with 32 channels and set stride to 1 for all convolutional layers. We employ ReLU activations after each convolutional layer. The output of the convolutional neural network is fed into a single fully-connected layer with 50-dimensional output. The action encoder consists of two fully-connected layers with ReLU activation functions. The hidden dimension is set to 512 and the output dimension is set to 16 for the action encoder. The transition model uses three fully-connected layers with ReLU activation functions. Its hidden dimension is set to 1024 and the output dimension is set to 50. The target encoder and transition model share the same network architecture with the online encoder and transition model, respectively. We stack 3 consecutive frames as an observation input, where each frame is an RGB rendering image with size $3 \times 84 \times 84$. We follow Lee et al. [13] by augmenting states by randomly shifting the image by $[-4, 4]$. We set $\lambda = 100$ and $\eta = 1000$ for all tasks.

We use the publicly released implementation of SAC by Yarats et al. [8]. The Q-function consists of three fully-connected layers with ReLU activation functions. The policy network is also parameterized as a 3-layer fully-connected network that outputs the mean and covariance for a Gaussian policy. The hidden dimension is set to 1024 for both the Q-function and policy network. Table 1 presents the remaining hyperparameters in detail. Following common practice [27, 8, 10], we treat action repeat as a hyperparameter to the agent. The number of repeated actions and the batch size for each task are listed in Table 2.

We compare our algorithm with the following leading baselines for continuous control from images: CURL [10], which combines a model-free RL agent with a contrastive learning objective that captures mutual information between two augmented images, PISAC [13], which maximizes the predictive information between the past and the future to learn latent representations which are used by a model-free RL agent, SAC+AE [8], which uses a regularized autoencoder for learning a mapping from high-dimensional states to compact embeddings, Dreamer [4], a model-based reinforcement learning method which learns a predictive dynamic model for planning, and Pixel SAC which uses a vanilla SAC operating purely from pixels.

We evaluate the performance of every agent after every 10K environment steps by computing an average return over 10 episodes. For each method, the

16

Table 1: Shared hyperparameters used for the comparative experiments

Parameter	Value
Replay buffer capacity	100000
Initial steps	1000
Optimizer	Adam
Q-function EMA τ_Q	0.01
Critic target update freq	2
Learning rate (ρ_a, ρ_c)	10^{-3}
Learning rate (ρ_e)	10^{-5}
Encoder EMA τ_e	0.05
Discount	0.99
Initial temperature	0.1

Table 2: Per-task hyperparameters

Task	Action Repeat	Batch size
Ball-in-cup Catch	4	256
Finger Spin	2	256
Reacher Easy	4	256
Cartpole Swingup	8	256
Walker Walk	2	256
Cheetah Run	4	512

SAC agent performs one gradient update per environment step to ensure a fair comparison. For a more reliable comparison, we run each algorithm with five different random seeds for each task. All figures show the average reward and 95% confidence interval unless specified otherwise.

4.2. Sample Efficiency

Figure 3 compares our algorithm with PISAC [13], CURL [10], SAC-AE [8], Dreamer [4] and Pixel SAC [14]. We use the version of PISAC that uses the same implementation of the SAC algorithm [8] to ensure a fair comparison to other model-free approaches. The evaluation data of Dreamer was provided to us by the author. The proposed algorithm achieves state-of-the-art performance on all the tasks against all the leading baselines, both model-based and model-free.

Table 3: Scores achieved by our method (mean and standard error for 5 seeds) and baselines at 100k and 500k environment steps. * indicates the best average return among these methods. The bold font indicates that the upper-bound on the reward (based on the given standard error intervals) for the given method, is larger or equal than the respective lower bound for every other image-based method.

| 500K step scores	CoDy(Ours)	PISAC	CURL	SAC-AE	Dreamer	Pixel SAC	State SAC
Finger Spin	**937**± **41***	**916 ± 58**	**854± 48**	**839± 68**	320±35	530±24	927 ± 43
Cartpole Swingup	**869± 4***	**857 ± 12**	837± 15	748±47	711± 94	436±94	870 ± 7
Reacher Easy	**957± 16***	**922 ± 32**	891± 30	678±61	581± 160	191±40	975 ± 5
Cheetah Run	**656± 43***	510 ± 27	492±22	476± 22	**571± 109**	250±26	772 ± 60
Walker Walk	**943± 17***	822 ± 98	897± 26	836±24	**924± 35**	97±62	964 ± 8
Ball-in-cup Catch	**970± 4***	961 ± 3	957± 6	831±25	**966± 8**	355±77	979 ± 6
100K step scores							
Finger Spin	**887 ± 39***	789 ± 34	750 ± 37	751± 57	33 ± 19	315 ± 78	672 ± 76
Cartpole Swingup	**784 ± 18***	591 ± 70	547 ± 73	305± 17	235± 73	263 ± 27	812 ± 45
Reacher Easy	**624 ± 42***	482 ± 91	460 ± 65	321 ± 26	148 ± 53	160 ± 48	919 ± 123
Cheetah Run	**323 ± 29***	**310 ± 28**	266 ± 27	264± 12	159± 60	160 ± 13	228 ± 95
Walker Walk	**673 ± 94***	**518 ± 70**	482 ± 28	362 ± 22	216 ± 56	105 ± 20	604 ± 317
Ball-in-cup Catch	**948 ± 6***	847 ± 21	741 ± 102	222 ± 21	172 ± 96	244 ± 55	957 ± 26
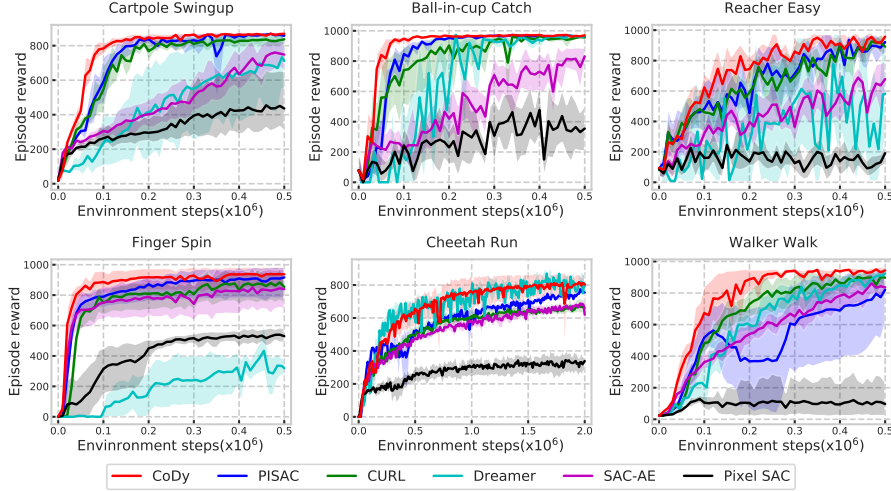
Figure 3: We compared the performance of CoDy with existing methods on six tasks from the Deepmind control suite. On all tasks, CoDy performs best in terms of sample efficiency.

Following CURL [10], in Table 3 we also compare performance at a fixed number of environment interactions (100k and 500K). Dreamer's results provided by the author didn't show the performance of the agent at exactly 100k and 500K environment interactions, and hence we interpolated between nearby values. We compare our algorithm with above baselines and an upper bound performance achieved by SAC [14] that operates directly from internal states instead of images. Averaged over 5 seeds, our algorithm achieves better sample-efficiency at 100K environment interactions and asymptotic performance at 500K environment interactions against existing baselines on all tasks. Furthermore, our algorithm matches the upper bound performance of SAC that is trained directly from internal states on several tasks at 500K environment interactions.

4.3. Generalization to Unseen Tasks

We test the generalization of our method by transferring the learned representations without additional fine-tuning to unseen tasks that share the same environment dynamics. Specifically, we learn representations and a SAC agent on a source task, fix the representations and then train a new SAC agent for a target task using the fixed representations. We use Cartpole Balance and Cartpole Swingup as the source task and the target task, respectively, which share the same environment dynamics but differ in their initial pole positions. We also use Walker Walk as the source task and Walker Stand and Walker Run as the target task, which all have different reward functions. Figure 4 compares the generalization of our method against PISAC, CURL, SAC-AE and a Pixel SAC that was trained from scratch. Our algorithm achieves not only better gen-
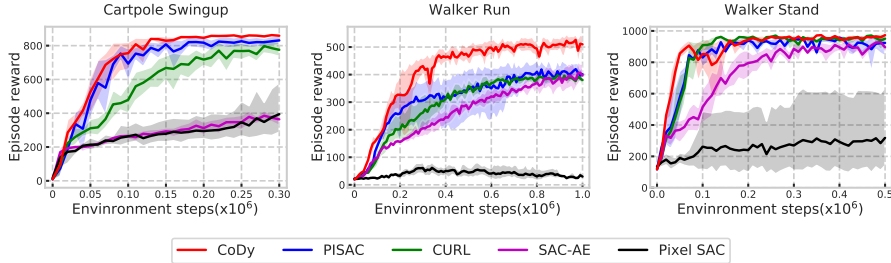
18

Figure 4: Generalization comparisons on the Deepmind control suite. Generalization of an encoder trained on the Cartpole Balance task and evaluated on the unseen Cartpole Swingup task (left). Generalization of an encoder trained on the Walker Walk task and evaluated on the unseen Walker Run (centre) and Walker Stand (right) task, respectively.

eralization than baselines, but also much better performance than vanilla Pixel SAC trained from scratch. We attribute the improved generalization compared to other contrastive learning based methods to our additional focus on improving latent Markovianity. The source and target tasks differ only in their reward function or the inital position, and share the same state-action dynamics. A Markovian representation is consistent with the state-action dynamics, in the sense that it does not discard information that would be useful for predicting aspects of the state that it chose to encode. We conjecture that this consistency improves the generalization of CoDy to unseen tasks.

4.4. Ablation Studies

We perform ablation studies to disentangle the individual contributions of the multi-view objective and the temporal mutual information objective, as well as the role of the prediction loss. We investigate three ablations of our CoDy model: Non-Tem CoDy, which only optimizes multi-view mutual information objective and the prediction error; Non-Pred CoDy, which only optimizes the multi-view and temporal mutual information objective, and Non-MV CoDy, which only optimizes the temporal mutual information objective and the prediction error. We present the performance of these ablations in Figure 5. CoDy achieves better or at least comparable performance and sample efficiency to all its own ablations across all tasks. This indicates that all three auxiliary tasks in CoDy play an important role in improving the performance on the benchmark tasks. Notably, CoDy outperforms Non-Pred CoDy across all tasks, which indicates that improving the Markovianity of embedding by minimizing the prediction error of dynamic models effectively achieves better sample efficiency. By comparing the performance of CoDy and Non-MV CoDy, we observe that our multi-view mutual information objective based on predicted embeddings helps to achieve better performance.
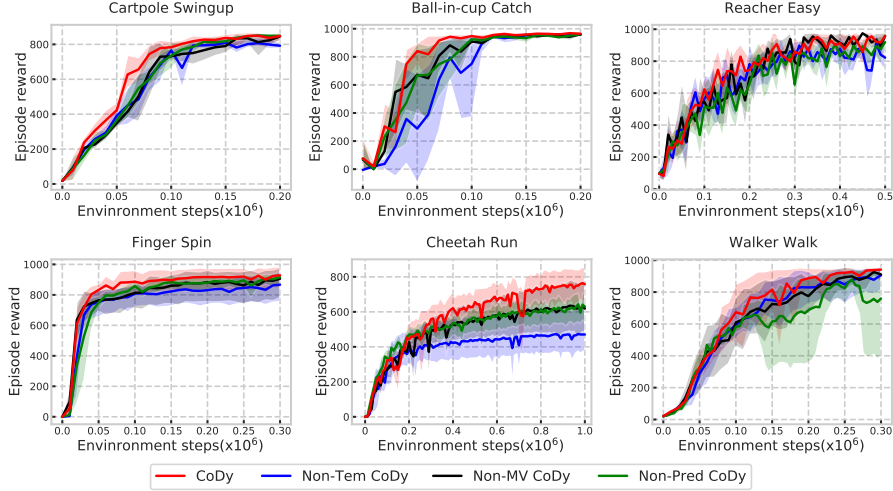
19

Figure 5: Performance on Deepmind control suite under consideration for ablated variants of our method.

4.5. Representation Visualization

We visualize the representations learned by CoDy using t-SNE [40] to inspect the learned embedding. With t-SNE visualization, there tend to be many overlapping points in the 2D space, which makes it difficult to view the overlapped representation examples. Therefore, we quantize t-SNE points into a 2D grid with a 30×20 interface by RasterFairy [41]. Figure 6 visualizes the representations learned with CoDy after training has completed on the Walker Walk task. Observations with similar robot configurations appear close to each other, which indicates that the latent space learned with CoDy meaningfully organizes the variation in robot configurations. Similar visualization for the remaining 5 tasks from the Deepmind control suite are shown in Appendix A.

5. Discussion and Conclusion

We presented CoDy, a self-supervised representation learning method which integrates contrastive learning with dynamic models for improving the sample efficiency of model-free reinforcement learning agents. Our method aims to learn state representations that are Markovian, linearly predictive and transformation invariant by minimizing three respective auxiliary losses during reinforcement learning from images. We compared our method with state-of-the-art approaches on a set of challenging image-based benchmark tasks. The results showed that our method can achieve better sample efficiency and performance compared to all the leading baselines on the majority of tasks, including reconstruction-based, contrastive-based as well as model-based methods. Fur-
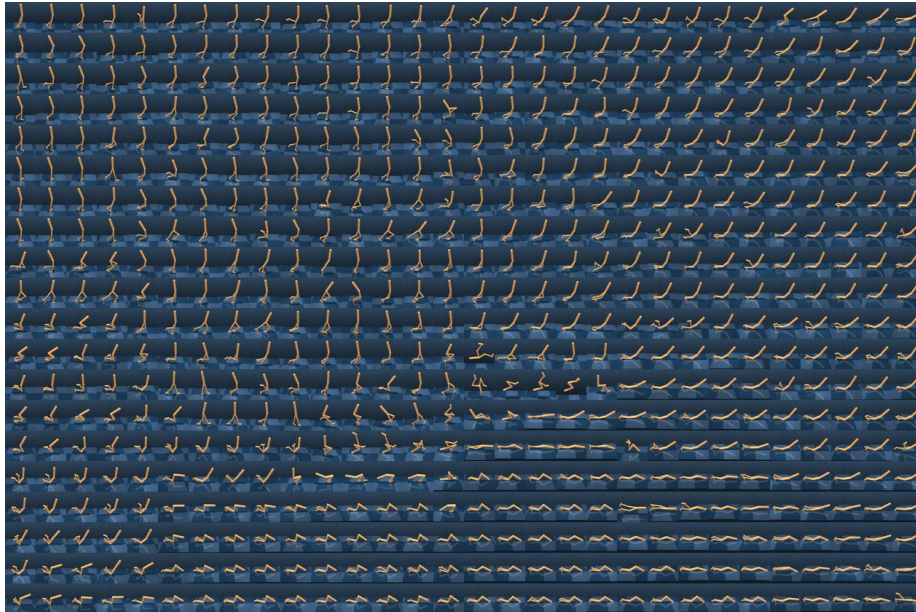
20

Figure 6: We visualize the representations learned with CoDy after training has completed on the Walker Walk task using t-SNE Visualization on a 30×20 grid.

thermore, we found representations learned with our method can achieve better generalization to unseen tasks than other model-free methods.

Although our method performed better than previous methods in our experiments, it is also a bit more complex by using three auxiliary tasks. We propose to learn a transition model in latent space and showed that it can be helpful for learning compact state embeddings, however, we do not make direct use of this model during reinforcement learning, which seems wasteful. Making use of the learned transition model for planning is a natural extension for future research. Moreover, we plan to extend our method to multimodal data, which incorporates other modalities, such as tactile and sound, into our representation learning model.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.
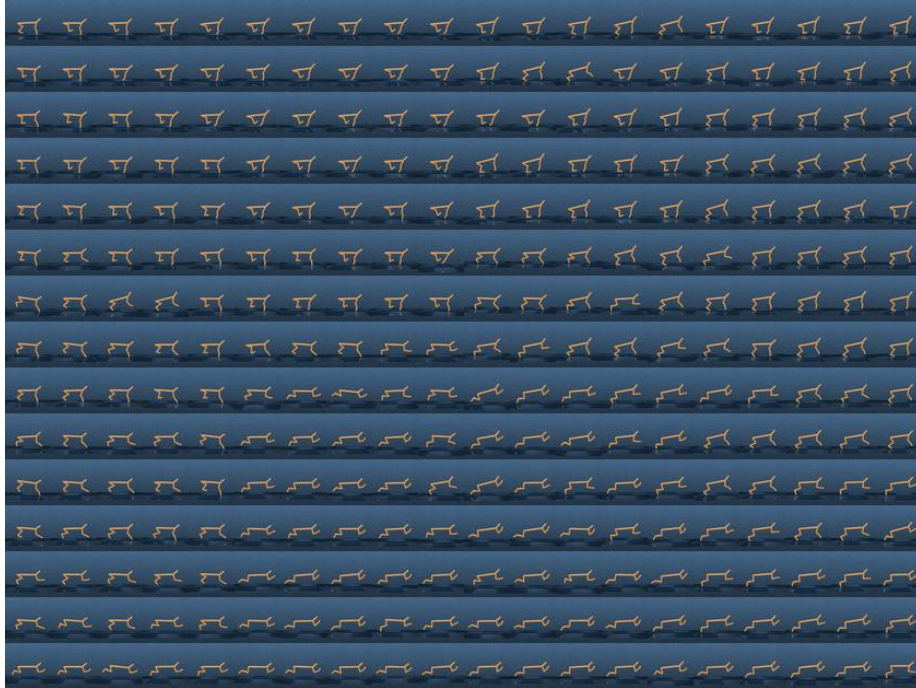
Acknowledgement

Figure A.1: t-SNE Visualization of representations learned with CoDy after training has completed on Cheetach Run task. The grid size is 20×15.

Appendix

A. Additional Visualizations of the Learned Representation

Additional results for the experiment in Section 4.5 that demonstrate that the latent space learned by CoDy meaningfully organizes the variations in robot configurations are shown in Figure A.1-A.5.

B. Computation Time

In order to evaluate whether our algorithm increases the time cost while improving the sample efficiency, we test computation time of our method and all baselines during policy learning. Specifically, we use a single NVIDIA RTX 2080 GPU and 16 CPU cores for each training run and record training time per 1000 environment steps after initializing the replay buffer. Table B.1 compares our method against PISAC, CURL, SAC-AE, Dreamer and Pixel SAC with respect to wallclock time. Pixel SAC has minimum time cost on all tasks
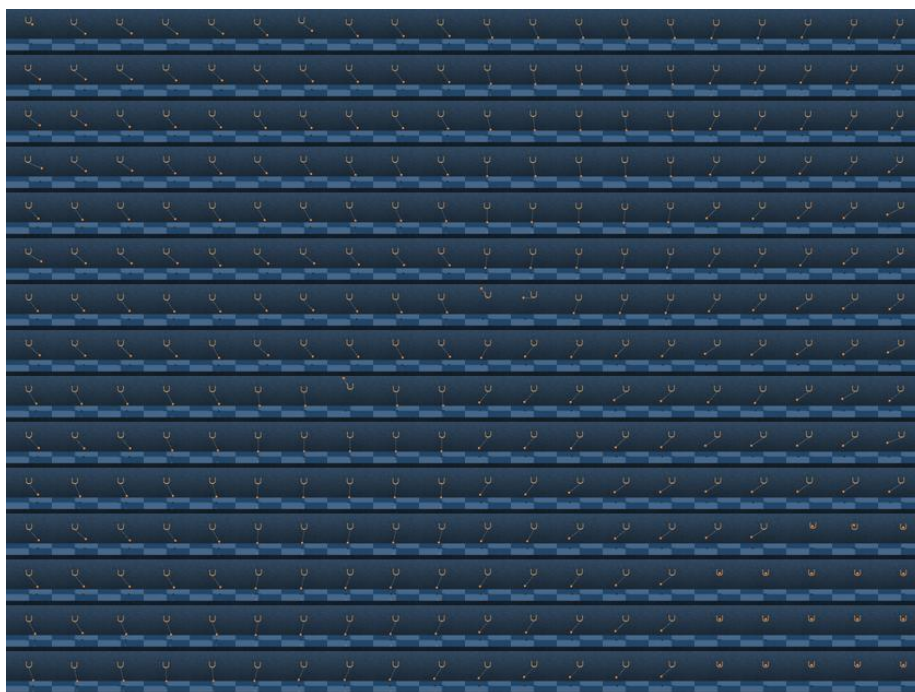
Figure A.2: t-SNE Visualization of representations learned with CoDy after training has completed on Ball-in-cup Catch task. The grid size is 20×15.
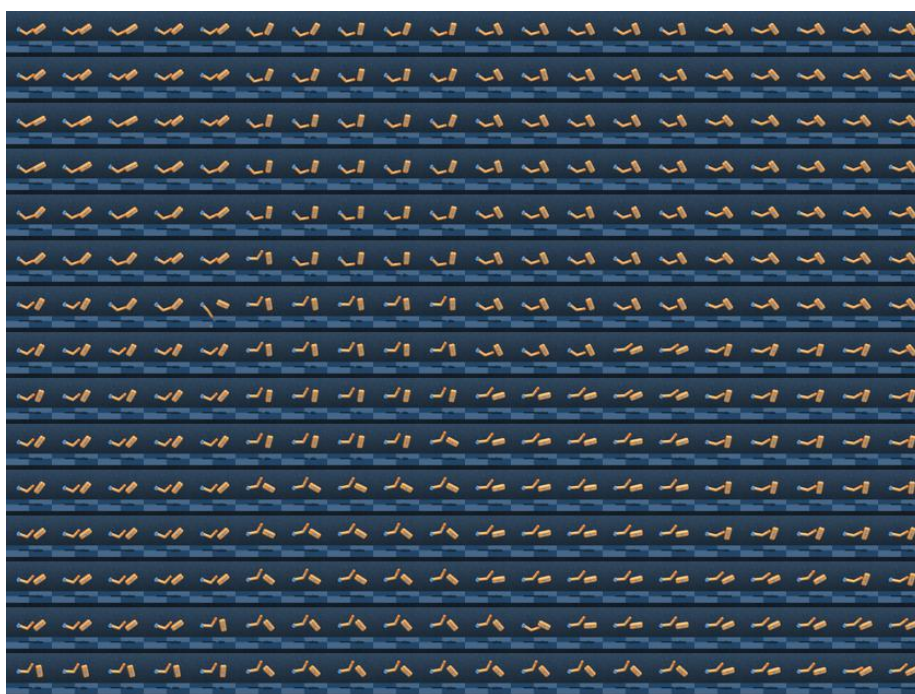
Figure A.3: t-SNE Visualization of representations learned with CoDy after training has completed on Finger Spin task. The grid size is 20×15.
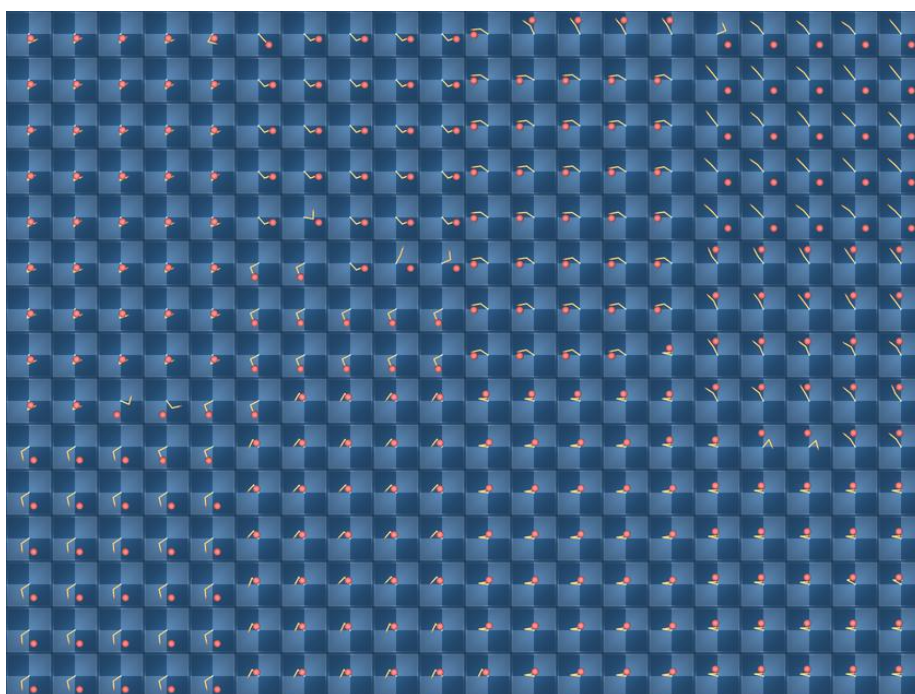
Figure A.4: t-SNE Visualization of representations learned with CoDy after training has completed on Reacher Easy task. The grid size is 20×15.
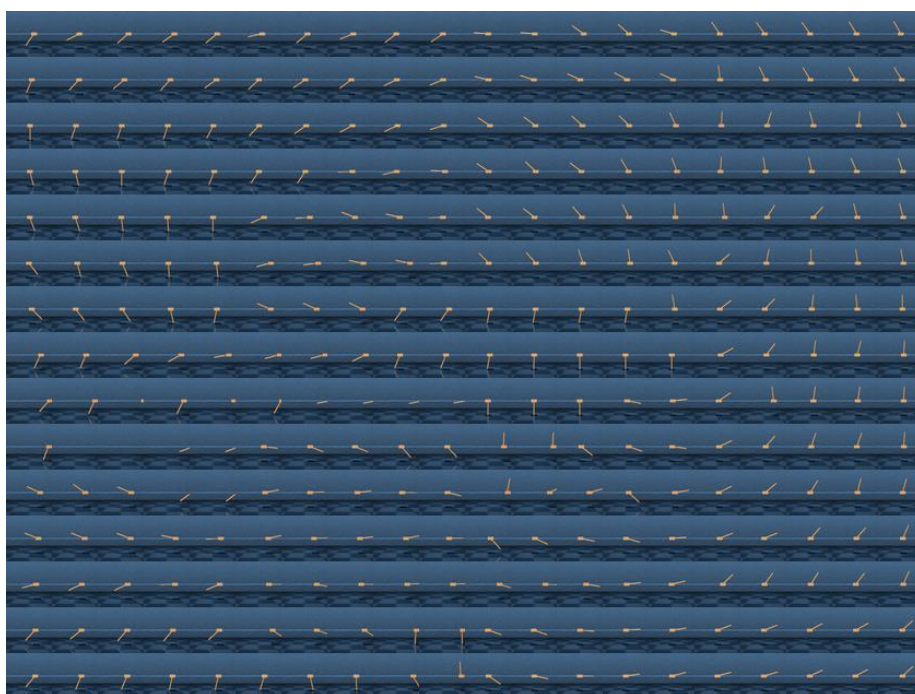
Figure A.5: t-SNE Visualization of representations learned with CoDy after training has completed on Cartpole Swingup task. The grid size is 20×15.

Table B.1: Computation time in seconds per 1000 environment steps(mean and standard error for 5 seeds) comparison to existing methods on six tasks from the Deepmind control suite. Bolded font indicates minimum mean among these methods.

	CoDy(Ours)	PISAC	CURL	SAC-AE	Dreamer	Pixel SAC
Cartpole Swingup	14± 1	21 ± 1	25 ± 1	10± 0	18± 1	**5± 0**
Ball-in-cup Catch	28± 0	47 ± 1	50± 0	19 ± 0	18± 1	**9± 0**
Finger Spin	56± 1	80 ± 1	98± 0	37 ± 0	**18± 0**	**18± 0**
Walker Walk	56± 1	80 ± 1	98± 1	38 ± 0	19± 0	**18± 0**
Reacher Easy	28± 1	43 ± 1	50± 0	19 ± 0	19± 0	**9± 0**
Cheetah Run	51± 0	38 ± 1	49± 1	19 ± 1	19± 1	**9± 0**

among these methods, since Pixel SAC operates directly from pixels without any auxiliary tasks or learned dynamic model. Besides, we observe that the time cost of each method highly depends on the chosen hyperparameters, especially the batch size. This may explain why our method requires less training time than CURL as well as more training time than Dreamer and SAC-AE, as the batch size of CoDy is smaller than CURL's and larger than Dreamer's and SAC-AE's. Moreover, we notice that the computation time of each method is highly related to the amount of action repeat across different tasks. Hence, we assume that most of the training time is spent on gradient updates, rather than image rendering of simulated environments. When PISAC has the same amount of action repeat and batch size as our method, it has on the majority of tasks a higher computation cost than CoDy, since a single sample of PISAC contains more image frames.

References

[1] S. Levine, C. Finn, T. Darrell, P. Abbeel, End-to-end training of deep visuomotor policies, The Journal of Machine Learning Research (2016) 1334–1373.

[2] F. Li, Q. Jiang, S. Zhang, M. Wei, R. Song, Robot skill acquisition in assembly process using deep reinforcement learning, Neurocomputing (2019) 92–102.

[3] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, A. Farhadi, Target-driven visual navigation in indoor scenes using deep reinforcement learning, in: 2017 IEEE international conference on robotics and automation (ICRA), IEEE, 2017, pp. 3357–3364.

[4] D. Hafner, T. Lillicrap, J. Ba, M. Norouzi, Dream to control: Learning behaviors by latent imagination, in: International Conference on Learning Representations, 2020.

[5] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, J. Davidson, Learning latent dynamics for planning from pixels, in: International Conference on Machine Learning, PMLR, 2019, pp. 2555–2565.

[6] J. Mattner, S. Lange, M. Riedmiller, Learn to swing up and balance a real pole based on raw visual input data, in: International Conference on Neural Information Processing, Springer, 2012, pp. 126–133.

[7] M. Watter, J. T. Springenberg, J. Boedecker, M. Riedmiller, Embed to control: a locally linear latent dynamics model for control from raw images, in: Proceedings of the 28th International Conference on Neural Information Processing Systems, 2015, pp. 2746–2754.

[8] D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, R. Fergus, Improving sample efficiency in model-free reinforcement learning from images, in: Proceedings of the AAAI Conference on Artificial Intelligence, 12, 2021, pp. 10674–10681.

[9] A. Zhang, R. T. McAllister, R. Calandra, Y. Gal, S. Levine, Learning invariant representations for reinforcement learning without reconstruction, in: International Conference on Learning Representations, 2020.

[10] M. Laskin, A. Srinivas, P. Abbeel, Curl: Contrastive unsupervised representations for reinforcement learning, in: International Conference on Machine Learning, PMLR, 2020, pp. 5639–5650.

[11] A. Anand, E. Racah, S. Ozair, Y. Bengio, M.-A. Côté, R. D. Hjelm, Unsupervised state representation learning in atari, in: Proceedings of the 33rd International Conference on Neural Information Processing Systems, 2019, pp. 8769–8782.

[12] A. v. d. Oord, Y. Li, O. Vinyals, Representation learning with contrastive predictive coding, arXiv preprint arXiv:1807.03748 (2018).

[13] K.-H. Lee, I. Fischer, A. Liu, Y. Guo, H. Lee, J. Canny, S. Guadarrama, Predictive information accelerates learning in rl, Advances in Neural Information Processing Systems (2020) 11890–11901.

[14] T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, in: International Conference on Machine Learning, PMLR, 2018, pp. 1861–1870.

[15] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing atari with deep reinforcement learning, in: Advances in neural information processing systems, 2013.

[16] B. D. Ziebart, A. Maas, J. A. Bagnell, A. K. Dey, Maximum entropy inverse reinforcement learning, in: Proceedings of the 23rd national conference on Artificial intelligence, 2008, pp. 1433–1438.

[17] R. Jonschkowski, O. Brock, Learning state representations with robotic priors, Autonomous Robots (2015) 407–428.

[18] R. Jonschkowski, R. Hafner, J. Scholz, M. Riedmiller, Pves: Position-velocity encoders for unsupervised learning of structured state representations, in: Proceedings of the Robotics: Science and Systems (RSS 2017) Workshop on New Frontiers for Deep Learning in Robotics, 2017.

[19] T. Lesort, M. Seurin, X. Li, N. Díaz-Rodríguez, D. Filliat, Deep unsupervised state representation learning with robotic priors: a robustness analysis, in: 2019 International Joint Conference on Neural Networks (IJCNN), IEEE, 2019, pp. 1–8.

[20] M. Morik, D. Rastogi, R. Jonschkowski, O. Brock, State representation learning with robotic priors for partially observable environments, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2019, pp. 6693–6699.

[21] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, P. Abbeel, Deep spatial autoencoders for visuomotor learning, in: 2016 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2016, pp. 512–519.

[22] H. Van Hoof, N. Chen, M. Karl, P. van der Smagt, J. Peters, Stable reinforcement learning with autoencoders for tactile and visual data, in: 2016 IEEE/RSJ international conference on intelligent robots and systems (IROS), IEEE, 2016, pp. 3928–3934.

[23] C. Finn, I. Goodfellow, S. Levine, Unsupervised learning for physical interaction through video prediction, in: Proceedings of the 30th International Conference on Neural Information Processing Systems, 2016, pp. 64–72.

[24] M. I. Belghazi, A. Baratin, S. Rajeshwar, S. Ozair, Y. Bengio, A. Courville, D. Hjelm, Mutual information neural estimation, in: Proceedings of the 35th International Conference on Machine Learning, PMLR, 2018, pp. 531–540.

[25] M. D. Donsker, S. S. Varadhan, Asymptotic evaluation of certain markov process expectations for large time, i, Communications on Pure and Applied Mathematics 28 (1975) 1–47.

[26] M. Gutmann, A. Hyvärinen, Noise-contrastive estimation: A new estimation principle for unnormalized statistical models, in: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, JMLR Workshop and Conference Proceedings, 2010, pp. 297–304.

[27] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, Y. Bengio, Learning deep representations by mutual information estimation and maximization, in: International Conference on Learning Representations, 2018.

[28] E. Bengio, V. Thomas, J. Pineau, D. Precup, Y. Bengio, Independently controllable features, arXiv preprint arXiv:1703.07718 (2017).

[29] D. Yarats, I. Kostrikov, R. Fergus, Image augmentation is all you need: Regularizing deep reinforcement learning from pixels, in: International Conference on Learning Representations, 2021.

[30] I. Fischer, The conditional entropy bottleneck, Entropy (2020) 999.

[31] C. Finn, S. Levine, Deep visual foresight for planning robot motion, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2017, pp. 2786–2793.

[32] M. Zhang, S. Vikram, L. Smith, P. Abbeel, M. Johnson, S. Levine, Solar: Deep structured representations for model-based reinforcement learning, in: International Conference on Machine Learning, PMLR, 2019, pp. 7444–7453.

[33] P. Agrawal, A. Nair, P. Abbeel, J. Malik, S. Levine, Learning to poke by poking: experiential learning of intuitive physics, in: Proceedings of the 30th International Conference on Neural Information Processing Systems, 2016, pp. 5092–5100.

[34] D. Pathak, P. Agrawal, A. A. Efros, T. Darrell, Curiosity-driven exploration by self-supervised prediction, in: International conference on machine learning, PMLR, 2017, pp. 2778–2787.

[35] J. Li, X. Shi, J. Li, X. Zhang, J. Wang, Random curiosity-driven exploration in deep reinforcement learning, Neurocomputing 418 (2020) 139–147.

[36] J. Li, J. Wang, H. Peng, L. Zhang, Y. Hu, H. Su, Neural fuzzy approximation enhanced autonomous tracking control of the wheel-legged robot under uncertain physical interaction, Neurocomputing 410 (2020) 342–353.

[37] K. He, H. Fan, Y. Wu, S. Xie, R. Girshick, Momentum contrast for unsupervised visual representation learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9729–9738.

[38] H. Kim, J. Kim, Y. Jeong, S. Levine, H. O. Song, Emi: Exploration with mutual information, in: International Conference on Machine Learning, PMLR, 2019, pp. 3360–3369.

[39] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. d. L. Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq, et al., Deepmind control suite, arXiv preprint arXiv:1801.00690 (2018).

[40] L. Van der Maaten, G. Hinton, Visualizing data using t-sne, Journal of machine learning research (2008).

[41] M. Klingemann, Rasterfairy, https://github.com/Quasimondo/RasterFairy (2015).

Bang You is the Ph.D candidate of the Huazhong University of Science and Technology and the co-training Ph.D student of the Technical University of Darmstadt. He started his Ph.D study on Mechatronics in 2019 in the Huazhong University of Science and Technology. He studies in the Technical University of Darmstadt from 2020. His research interests include reinforcement learning, computer vision and robotics.

Oleg Arenz received his doctorate in 2020 from the Technical University of Darmstadt under the supervision of Prof. Gerhard Neumann and was also affiliated as a research assistant at the University of Lincoln for a time. He is currently a staff scientist in the Intelligent Autonomous Systems group at the Technical University of Darmstadt. His research focuses on imitation learning, inverse reinforcement learning, variational inference, human-robot interactions and shared control.

Youping Chen is a professor of the School of Mechanical Science and Engineering, Huazhong University of Science and Technology and the senior member of Chinese Mechanical Engineering Society (CSME). He received his B.S. and M.S. degrees on Mechanical Engineering from Shanghai Jiao Tong University in 1982 and 1984 respectively. He received his Ph.D degree on Mechanical Engineering in 1990 from the Huazhong University of Science and Technology. He published more than 160 literatures on Scopus including machine vision, robotics and mechatronics.

Jan Peters is a full professor (W3) for Intelligent Autonomous Systems at the Computer Science Department of the Technische Universitaet Darmstadt. Jan Peters has received the Dick Volz Best 2007 US PhD Thesis Runner-Up Award, the Robotics: Science & Systems - Early Career Spotlight, the INNS Young Investigator Award, and the IEEE Robotics & Automation Society's Early Career Award as well as numerous best paper awards. In 2015, he received an ERC Starting Grant and in 2019, he was appointed as an IEEE Fellow.