Learning Robust Representation for Reinforcement Learning with Distractions by Reward Sequence Prediction

Qi Zhou[1] Jie Wang*[1,2] Qiyuan Liu[1] Yufei Kuang[1] Wengang Zhou[1,2] Houqiang Li[1,2]

[1]1CAS Key Laboratory of Technology in GIPAS, University of Science and Technology of China
[2]Institute of Artificial Intelligence, Hefei Comprehensive National Science Center

Abstract

Reinforcement learning algorithms have achieved remarkable success in acquiring behavioral skills directly from pixel inputs. However, their application in real-world scenarios presents challenges due to their sensitivity to visual distractions (e.g., changes in viewpoint and light). A key factor contributing to this challenge is that the learned representations often suffer from overfitting task-irrelevant information. By comparing several representation learning methods, we find that the key to alleviating overfitting in representation learning is to choose proper prediction targets. Motivated by our comparison, we propose a novel representation learning approach—namely, **r**eward **s**equence **p**rediction (RSP)—that uses reward sequences or their transforms (e.g., discrete time Fourier transform) as prediction targets. RSP can efficiently learn robust representations as reward sequences rarely contain task-irrelevant information while providing a large number of supervised signals to accelerate representation learning. An appealing feature is that RSP makes no assumption about the type of distractions and thus can improve performance even when multiple types of distractions exist. We evaluate our approach in Distracting Control Suite. Experiments show that our method achieves state-of-the-art sample efficiency and generalization ability in tasks with distractions.

1 INTRODUCTION

Recent deep reinforcement learning (RL) algorithms have achieved great success in learning behaviors directly from pixel inputs Tassa et al. [2018]. However, many of these algorithms suffer from obvious performance degradation in tasks with visual distractions Zhang et al. [2021], Stone et al. [2021], such as variations in background, color, and viewpoint. In the training phase, the distraction significantly reduces the sample efficiency Stone et al. [2021] and makes the optimization unstable. One major reason is that these algorithms can not efficiently extract task-relevant information from pixels and then suffer from large approximation error. In the evaluation phase, learned policies often generalize poorly to new environments where the distractors are different from those in the training environment Song et al. [2019], Stone et al. [2021], Agarwal et al. [2021], Kirk et al. [2021]. The poor generalization comes from the overfitting to the training data. That is, the learned policies select actions based on non-causal features, but these features may significantly change in test environments Song et al. [2019].

Recent work shows that representation learning is the key to improving robustness against distractions. Some methods improve representations by data augmentation Fan et al. [2021], Hansen et al. [2021b]. They encourage the consistency of outputs when applying different transformations to the inputs. However, many of these methods assume the type of distraction and then select proper transformations according to this assumption. Moreover, they often require a clean environment (no distractions exist) for stable optimization. However, the clean environment is unavailable in many real-world tasks. Another line of work learns robust representations by auxiliary tasks, which are additional objectives optimized simultaneously with standard RL objectives Schwarzer et al. [2020], Lee et al. [2019a], Gelada et al. [2019]. These auxiliary tasks improve robustness by preventing representations from exploiting task-irrelevant information Zhang et al. [2021], Agarwal et al. [2021]. However, many of these methods are sample inefficient. They usually require many samples to learn robust representations and even struggle to learn good behaviors when multiple distractions appear simultaneously. Therefore, learning robust representations with high sample efficiency remains challenging, especially when different types of distractions exist at the same time Stone et al. [2021].

* Corresponding author

Proceedings of the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023), PMLR 216:2551–2562.

To tackle this problem, we propose **r**eward **s**equence **p**rediction (RSP), a novel approach that efficiently learns robust representations in tasks with distractions. First, we analyze several representation learning methods . Our results show that representation learning should follow the information bottleneck principle Tishby et al. [2000], Tishby and Zaslavsky [2015], Vera et al. [2018]. That is, the prediction targets used to learn representations should provide sufficient task-relevant information for sample-efficient training while containing little task-irrelevant information for good generalization. Second, we propose to use reward sequences or their transforms (e.g., discrete time Fourier transform) as prediction targets. We show that reward sequences and their transforms provide large amounts of information about the long-term future while having a low correlation with distractors. Then, we propose a TD-style algorithm to efficiently predict long reward sequences and their transforms. This algorithm enjoys the convergence property of contraction operations like the traditional TD-learning of Q functions. Finally, we propose a method that learns the transform of reward sequences by maximizing the information in prediction targets. This method can automatically exploit the property of reward sequences in different tasks.

RSP is compatible with most visual RL algorithms. In our experiments, we combine RSP with DrQ and DrQv2 Yarats et al. [2021a]. We evaluate our method in Distracting Control Suite Stone et al. [2021]. In the multi-distraction setting, RSP achieves up to three times performance improvement in average return and is much more sample-efficient than our baselines. Moreover, in the video-background setting, RSP learns policies that generalize well to unseen distractions. We also provide analyses to show that RSP can learn robust representations that hardly encode task-irrelevant information.

Our contributions consist of four parts. (1) We compare different auxiliary tasks and propose to use reward sequences as prediction targets. We show that using reward sequences as prediction targets can better satisfy the information bottleneck principle (Figure 1) than other auxiliary tasks. (2) We propose a novel TD-style learning method for the efficient computation of prediction targets. We prove that this method enjoys the convergence properties of contraction mappings. (3) We propose a method to learn the transform of reward sequences. It avoids the manual selection of transform for different tasks. (4) Our experiments demonstrate that RSP significantly improves the sample efficiency and generalization when distractions exist. We provide extensive analyses to understand the excellent performance of RSP.

2 RELATED WORK

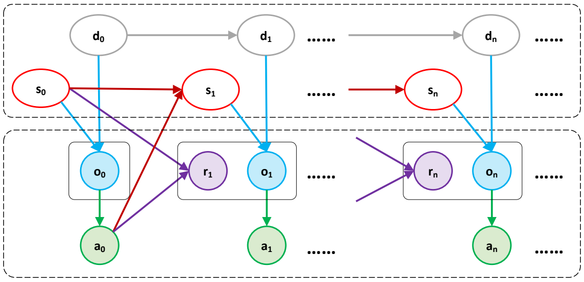
RL with distractions: Recent work that considers distractions usually focus on improving generalization Kirk et al. [2021], Malik et al. [2021], Ghosh et al. [2021], Raileanu and Fergus [2021], Hansen et al. [2021a]. A promising approach to improve generalization is to use regularization, such as $\ell 2$ regularization Igl et al. [2019], Cobbe et al. [2019], dropout Igl et al. [2019], batch normalization Igl et al. [2019], Farebrother et al. [2018] and information bottleneck regularization Igl et al. [2019], Lu et al. [2020], Eysenbach et al. [2021], Fan and Li [2021]. Recently, data augmentation has shown excellent potential to improve the generalization of deep RL Hansen and Wang [2020], Hansen et al. [2021b], Laskin et al. [2020], Cobbe et al. [2019], Lee et al. [2019b], Raileanu et al. [2020], Zhang and Guo [2021], Zhou et al. [2021], Fan et al. [2021]. However, most of these methods assume that the training environment is without distractions, which may be impractical in real-world tasks. Moreover, they often assume the type of distractions to select transforms, which requires prior knowledge about environments. In contrast, some recent methods can directly train policies in environments with distractions. A popular idea is state abstraction Li et al. [2006], Ferns et al. [2011], Ferns and Precup [2014], Zhang et al. [2019, 2020]. For example, Zhang et al. [2021] and Agarwal et al. [2021] propose to group states according to π-bisimulation metric Castro [2020] and the policy similarity metric, respectively. Another kind of method improves generalization ability via invariance. For example, Sonar et al. [2021] introduces the ideas of invariant risk minimization Arjovsky et al. [2019] into policy gradient methods. Though these methods achieve promising generalization, some of them will hurt the sample efficiency, and asymptotic performance Chen and Pan [2022]. Recent work attempts to achieve high sample efficiency of model-based RL in tasks with distractions Fu et al. [2021], Nguyen et al. [2021], Deng et al. [2021]. However, they also suffer from low sample efficiency when multiple distractions exist (Section 6.1). Similar to RSP, CRESP Yang et al. [2022] conduct auxiliary tasks by reward signs. However, there are clear differences between CRESP and RSP. CRESP only considers the generalization ability but RSP can also improve the sample efficiency. CRESP is motivated by the invariance across different environments while RSP is motivated by the information bottleneck principle. CRESP can only predict few steps of rewards (3-7) while RSP use long reward sequences or their transforms as prediction targets. Section 6.2 shows that RSP achieves better performance than CRESP.

Auxiliary Tasks: Previous work uses auxiliary tasks to improve the representations of high-dimensional observations Jaderberg et al. [2016]. Reconstruction-based auxiliary tasks simultaneously learn an encoder and a decoder by minimizing the reconstruction errors Hafner et al. [2019b,a], Ke et al. [2019], Yarats et al. [2021b], Lee et al. [2019a]. They encourage agents to capture all information, whether it is relevant to the control task or not. Recently, Zhang et al. [2021] point out that task-irrelevant information can hinder the agent from learning robust representations and lead to performance degradation. Contrastive-based auxiliary tasks

Srinivas et al. [2020], Zhu et al. [2020], Liu et al. [2021] minimize the distance between embeddings of similar observations while maximizing the distance between embeddings of dissimilar observations. Many of these methods require prior knowledge to define the similarity Zhang et al. [2021]. We empirically show that contrastive-based auxiliary tasks also suffer from the distractions of task-irrelevant information (Figure 1). Model-based auxiliary tasks Gelada et al. [2019], Schwarzer et al. [2020], Yu et al. [2021], Lee et al. [2019a], Hafner et al. [2019a], Okada and Taniguchi [2021], Nguyen et al. [2021], Deng et al. [2021] capture the information about the dynamics in latent spaces. Given current observations, these auxiliary tasks encourage the agent to discriminate the subsequent observations. However, the distractors also provide information to discriminate the subsequent observations, and we observe that model-based auxiliary tasks tend to misuse task-irrelevant information (Figure 1). Learning value functions over multiple time horizons is also a popular auxiliary task Fedus et al. [2019], which can improve the sample efficiency in Atari tasks. In this work, we focus on designing auxiliary tasks for deep reinforcement learning with distractions. Therefore, the auxiliary tasks should not only encourage neural networks to extract useful information but also needs to ignore task-irrelevant information.

3 PRELIMINARIES

3.1 NOTATION



An infinite-horizon Markov decision process (MDP) $\mathcal{M}$ is defined by $(\mathcal{S}, \mathcal{A}, P_s, R_s, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P_s : \mathcal{S}$ is the transition probability function, $R_s : \mathcal{S} \times \mathcal{A} \to [0,1]$ is the reward function, $\gamma \in (0,1)$ is the discount factor. We define an MDP with distractions by $(\mathcal{M}, \mathcal{D}, \mathcal{P}_d, \mathcal{O}, g)$. Here, $\mathcal{M}$ is the original MDP, $\mathcal{D}$ is a set of distractors, P_d is the transition probability functions of distractors, $\mathcal{O}$ is a set of observations, $g : \mathcal{S} \times \mathcal{D} \to \mathcal{O}$ is the observation function. Let $\pi : \mathcal{O} \times \mathcal{A} \to [0,1]$ denote a policy. We illustrate the process in the above figure. In this decision process, only the rewards $\{\mathbf{r}_i\}_{i=1}^{\infty}$ and the observation $\{\mathbf{o}_i\}_{i=0}^{\infty}$ can be observed. The actions $\{\mathbf{a}_i\}_{i=0}^{\infty}$ are selected under a given policy π. The low-dimensional state $\{\mathbf{s}_i\}_{i=0}^{\infty}$ and the distractors $\{\mathbf{d}_i\}_{i=0}^{\infty}$ can not be observed. We assume that states in $\mathcal{S}$ are not equivalent to each other and $g(s_1, d_1) \neq g(s_2, d_2)$ if $s_1 \neq s_2$. Under the assumption, an MDP with distractions still enjoys the Markov property. Moreover, the assumption guarantees that there exists a function $\phi_o : \mathcal{O} \to \mathcal{S}$ mapping observations to origin states. Therefore, we can let $P_o : \mathcal{O} \times \mathcal{A} \times \mathcal{O} \to [0,1]$ denote the transition probability function of observations and define $R_o : \mathcal{O} \times \mathcal{A} \to [0,1]$ by $R_o(o,a) = R_s(\phi_o(o), a)$.

3.2 DRQ AND DRQV2

Both DrQ and DrQv2 are state-of-the-art algorithms in visual control tasks. DrQ Yarats et al. [2020] is a combination of data augmentation and soft actor critic Haarnoja et al. [2018]. It uses optimality invariant state transformations f—which do not change the state-action value—to augment the training data. DrQ uses the data augmentation to improve the accuracy of target values and reduce the variance of stochastic gradients. Specifically, DrQ updates the Q function, which is a neural network parameterized by θ, by minimizing the mean square error J_Q

$$\mathbf{y}_n = \mathbf{r}_n + \frac{\gamma}{K} \sum_{k=1}^{K} Q_{\bar{\theta}} \left(f\left(\mathbf{o}'_n, \mathbf{v}_k\right), \mathbf{a}'_n \right),$$

$$J_Q(\theta) = \frac{1}{NK} \sum_{n=1,k=1}^{N,K} \left(Q_\theta \left(f\left(\mathbf{o}_n, \mathbf{v}_k\right), \mathbf{a}_n \right) - \mathbf{y}_n \right)^2 .$$

Here, $(\mathbf{o}_n, \mathbf{a}_n, \mathbf{r}_n, \mathbf{o}'_n)$ is uniformly sampled from a replay buffer, $\bar{\theta}$ is the parameters of target networks, $\mathbf{a}'_n$ is sampled from the distribution $\pi(\cdot|\mathbf{o}'_n)$, and $\mathbf{v}_k$ is a random parameter of the transform f. DrQ learns policies via maximum entropy RL Haarnoja et al. [2018] for efficient exploration and stable optimization. Therefore, DrQ actually uses the sum of $\mathbf{y}_n$ and an entropy term as the target value.

Recently, Yarats et al. [2021a] propose DrQv2, which makes several modifications to DrQ. DrQv2 replaces the maximum entropy term Haarnoja et al. [2018] with a scheduled noise for adjustable exploration and borrows the idea of target policy smoothing from TD3 Fujimoto et al. [2018] to reduce the bias of Q functions. Furthermore, DrQv2 uses multi-step TD to learn value functions

$$\mathbf{y}_i = \sum_{t=1}^{T} \gamma^{k-1} \mathbf{r}_{n,t} + \frac{\gamma^T}{K} \sum_{k=1}^{K} Q_{\bar{\theta}} \left(f\left(\mathbf{o}_{n,T}, \mathbf{v}_k\right), \mathbf{a}_{n,T} \right),$$

where $\mathbf{r}_{n,1:T}$ is the subsequent rewards after $\mathbf{o}_n$, $\mathbf{o}_{n,1:T}$ is the subsequent observations, and $\mathbf{a}_{n,T}$ is sampled according to the policy $\pi(\cdot|\mathbf{o}_{n,T})$ and the scheduled noise.

3.3 DISCRETE-TIME FOURIER TRANSFORM

Discrete-time Fourier transform (DTFT) is used to analyze the frequency properties of a time series. It converts a se-

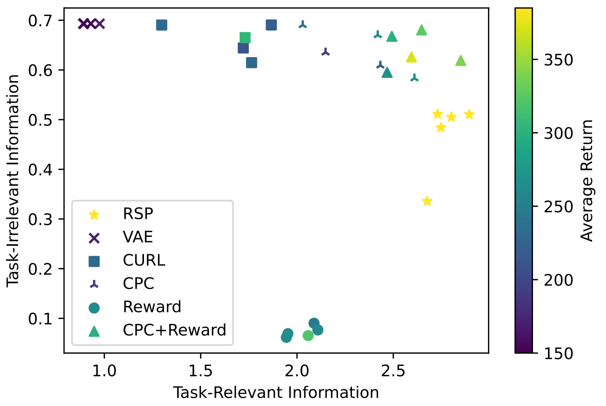


Figure 1: Comparison between different auxiliary tasks in a visual control task with background distractions.

quence $\{c_n\}_{-\infty}^{\infty}$ into a complex-value function $h_c(x)$ by

$$h_c(x) = \sum_{n=-\infty}^{\infty} c_n e^{-nxj}.$$

As DTFT is invertible, the function $h_c(x)$ contains all information about the sequence $\{c_n\}_{-\infty}^{\infty}$ Oppenheim [1999].

4 REWARD SEQUENCE PREDICTION

In this section, we introduce a novel representation learning method—namely, **r**eward **s**equence **p**rediction (RSP)—that learns robust representations in tasks with distractions. First, we compare six methods to study how to select prediction targets for representation learning. Second, we propose RSP that uses reward sequences or their transforms as prediction targets. Then, we propose a TD-style algorithm for the efficient prediction of long reward sequences or their transforms. Finally, we propose a method that automatically learns the transform by maximizing information. In this section, actions are sampled under a fixed policy, and we omit the policy in notations for simplification. We provide proofs of propositions in Appendix 1.

4.1 AUXILIARY TASK DESIGN

This part discuss the relation between the performance and the prediction target used for representation learning. To do this, we analyze how much task-relevant and task-irrelevant information are encoded when using different representation learning methods. We compare six representaion learning methods, including VAE Yarats et al. [2021b], which is a reconstruction-based auxiliary task; CURL Srinivas et al. [2020], which is based on multi-view contrastive learning; one-step CPC Oord et al. [2018], which learns a model in the latent space; one-step reward prediction, which predicts one-step rewards; the combination of CPC and reward prediction; and our method RSP (detailed in the following sections). We compare them in a modified Cartpole Swingup environment, where the background is replaced with random images. We let ϕ_θ denote the encoder learned by auxiliary tasks. We use the InfoNCE objective to estimate the mutual information $I\left(\phi_\theta(\mathbf{o_t}); \mathbf{s_t}\right)$, which stands for task-relevant information. We train a network with a cross-entropy loss to predict background images and use the loss to estimate the mutual information $I\left(\phi_\theta(\mathbf{o_t}); \mathbf{d_t}\right)$, which stands for task-irrelevant information.

Figure 1 shows that all methods except RSP struggle to learn high-return policies. We observe that the performance will be low when whether too much task-irrelevant information (VAE, CPC, CURL, and CPC+Reward) or too little task-relevant information (Reward) are encoded in the learned representations. Moreover, we notice that the prediction targets used in auxiliary tasks almost determine how much task-relevant and task-irrelevant information is encoded. For example, the prediction targets of VAE include all task-irrelevant elements, so representations learned by VAE contain the largest amount of task-irrelevant information. The one-step reward signs almost contain no task-irrelevant information, so representations learned by reward prediction contain the smallest amount of task-irrelevant information. The prediction targets of CPC+Reward contain extra reward information compared with CPC, so representations learned by CPC+Reward encode more task-relevant information than those learned by CPC only.

Our results imply that selecting proper prediction targets for representation learning is the key to improving the robustness against distractions. *The selection of prediction targets should follow the information bottleneck principle*. Specifically, prediction targets should contain as much task-relevant information as possible while being as uncorrelated with distractions as possible.

4.2 PREDICTION TARGETS OF RSP

As discussed in Section 4.1, one-step reward signs rarely contain information about distractions but do not provide sufficient information for representation learning. Therefore, we propose to use reward sequences as prediction targets. That is, given an observation $\mathbf{o}_t$, an action $\mathbf{a}_t$, we encourage representations to encode information about the reward sequence $\mathbf{r}_{t+1:t+T}$, which is sampled under a policy π. Reward sequences provide more task-relevant information than one-step rewards as the inequality $I(\mathbf{s_t}; \mathbf{r}_{t+1}) \leq I(\mathbf{s}_t; \mathbf{r}_{t+1:t+T})$ always holds. We argue that reward sequences rarely contain task-irrelevant information similar to one-step rewards. Proposition 4.1 provides an upper bound for the task-irrelevant information in reward sequences.

Proposition 4.1. *Assume that actions are sampled under a fixed policy π. Let $I(X;Y|Z)$ denote the mutual informa-*

tion between X and Y conditioned on Z. Then, we have

$$I(\mathbf{d}_t; \mathbf{r}_{t+1:t+T} \mid \mathbf{s}_t) \leq \sum_{i=\mathbf{0}}^{T-1} I(\mathbf{d}_{t+i}; \mathbf{a}_{t+i} | \mathbf{s}_{t+i}).$$

The proposition shows that the task-irrelevant information (left-hand side) provided by reward sequences is less than that used to select actions (right-hand side). Given different observations o and o' that correspond to a same state s, the task-irrelevant information used to select actions is negligible if the two action distributions $\pi(\cdot|o)$ and $\pi(\cdot|o')$ are similar. Therefore, the left-hand side will be small if we control the right-hand side by regularizing policies. There are optional regularization terms to control the right-hand side. For example, the commonly used maximum entropy regularization [Haarnoja et al., 2018] can control the right-hand side by encouraging all action distributions to be close to the same uniform distribution. In our implementation of RSP, we regularize policies by the $\ell 2$ norm of actions with a small coefficient, as discussed in Appendix 3.1.

As rewards in the long-term future rarely depend on the current observation and action, we discount rewards according to the time step, similar to the definition of Q values. That is, we consider the expectation of discounted reward

$$e_n(o, a; \pi) \triangleq \mathbb{E}_\pi[\gamma^n \mathbf{r}_{n+1} \mid \mathbf{o}_0 = o, \mathbf{a}_0 = a]. \qquad (1)$$

Based on the discounted reward, we can define two variants of RSP. The first one directly use $\{e_n(o, a; \pi)\}_{n=0}^{L-1}$ as the L-dimensional prediction target $Z_{\pi,1}(o, a)$. That is,

$$[Z_{\pi,1}(o, a)]_i \triangleq e_i(o, a; \pi).$$

The second one considers the DTFT of reward sequences and uses the values at L points as the prediction targets $Z_{\pi,2}(o, a)$. Specifically, it is defined by

$$[Z_{\pi,2}(o, a)]_i \triangleq \sum_{n=0}^{\infty} e_n(o, a; \pi) \exp\left(-\frac{2ni\pi}{L} j\right).$$

The prediction target $Z_{\pi,2}(o, a)$ contains the frequency-domain information about reward sequences. We argue that the frequency-domain information can improve performance in tasks where state/reward sequences are approximately periodic (please see discussion in Appendix 3.5).

4.3 TD-STYLE LEARNING

This part proposes a TD-style method to efficiently predict long reward sequences. First, we note that both two types of prediction targets $Z_{\pi,i}(o, a)$ in Section 4.2 can be unified via contraction mappings $\mathcal{T}_{\pi,i}$. Proposition 4.2 provides the form of the contraction mapping $\mathcal{T}_{\pi,i}$.

Proposition 4.2. *There exist contraction mappings $\mathcal{T}_{\pi,i}$ such that Equations (2) holds for both $i = 1, 2$*

$$\begin{aligned} Z_{\pi,i}(o, a) &= (\mathcal{T}_{\pi,i} Z_{\pi,i})(o, a), \qquad (2) \\ (\mathcal{T}_{\pi,i} Z_{\pi,i})(o, a) &= W_i R_o(o, a) + \Gamma_i \mathbb{E}_{\mathbf{o}', \mathbf{a}'}[Z_{\pi,i}(\mathbf{o}', \mathbf{a}')], \end{aligned}$$

where $W_i \in \mathbb{R}^L$, $\Gamma_i \in \mathbb{R}^{L \times L}$, $\mathbf{o}'$ is sampled with probability $P_o(\mathbf{o}'|o, a)$, $\mathbf{a}'$ is sampled with probability $\pi(\mathbf{a}'|\mathbf{o}')$, and all vectors are column vectors.

For both $i = 1, 2$, we provide complete expressions of W_i and Γ_i in Appendix 1.2. In tabular settings, our TD-style learning method computes prediction targets $Z_{\pi,i}(o, a)$ by repeatedly apply the operator $\mathcal{T}_{\pi,i}$. Thanks to the properties of contraction mappings, this method enjoys the exponential convergence rate similar to the TD-learning of Q values.

In deep RL, we train a network Z_θ to approximate $Z_{\pi,i}$. First, we sample a batch of data $\{(\mathbf{o}_i, \mathbf{a}_i, \mathbf{r}_i, \mathbf{o}'_i, \mathbf{a}'_i)\}_{i=1}^N$ with a size of N from the replay buffer. Then, we compute prediction targets by applying the operator $\mathcal{T}_{\pi,i}$ to the current prediction, i.e., $\mathbf{z} = W_i \mathbf{r} + \Gamma_i Z_\theta(\mathbf{o}', \mathbf{a}')$. Finally, we optimize the network Z_θ by minimize the mean square error

$$J_{RSP} = \frac{1}{N} \sum_{n=1}^{N} \|Z_\theta(\mathbf{o}_n, \mathbf{a}_n) - \mathbf{z}_n\|_2^2. \qquad (3)$$

This TD-style learning procedure can compute prediction targets without sampling long reward sequences from the buffer. Without the TD-style method, using $Z_{\pi,2}$ as prediction targets is impractical as it requires infinite sequences to compute prediction targets. Moreover, similar to the TD-learning of Q values, it significantly reduces the variance of gradients and thus can improve the sample efficiency.

4.4 LEARNING TRANSFORM

Motivated by Proposition 4.2, we can define many prediction targets by different W and Γ. Each pair of W and Γ corresponds to a transform of reward sequences. Therefore, we can view learning the parameters W and Γ as learning transforms of reward sequences. Proposition 4.3 shows how to define transforms of reward sequences by defining contraction mappings.

Proposition 4.3. *For any $W \in \mathbb{R}^L$, if the infinity-norm of $\Gamma \in \mathbb{R}^{L \times L}$ is less than 1, the operator $\mathcal{T}_\pi$ defined by*

$$(\mathcal{T}_\pi Z_\pi)(o, a) = W R_o(o, a) + \Gamma \mathbb{E}_{\mathbf{o}', \mathbf{a}'}[Z_\pi(\mathbf{o}', \mathbf{a}')],$$

is a contraction mapping. The prediction target Z_π defined as the fix point of $\mathcal{T}_\pi$ satisfies the equation

$$Z_\pi(o, a) = \sum_{n=0}^{\infty} \left(\frac{\Gamma}{\gamma}\right)^n W e_n(o, a; \pi).$$

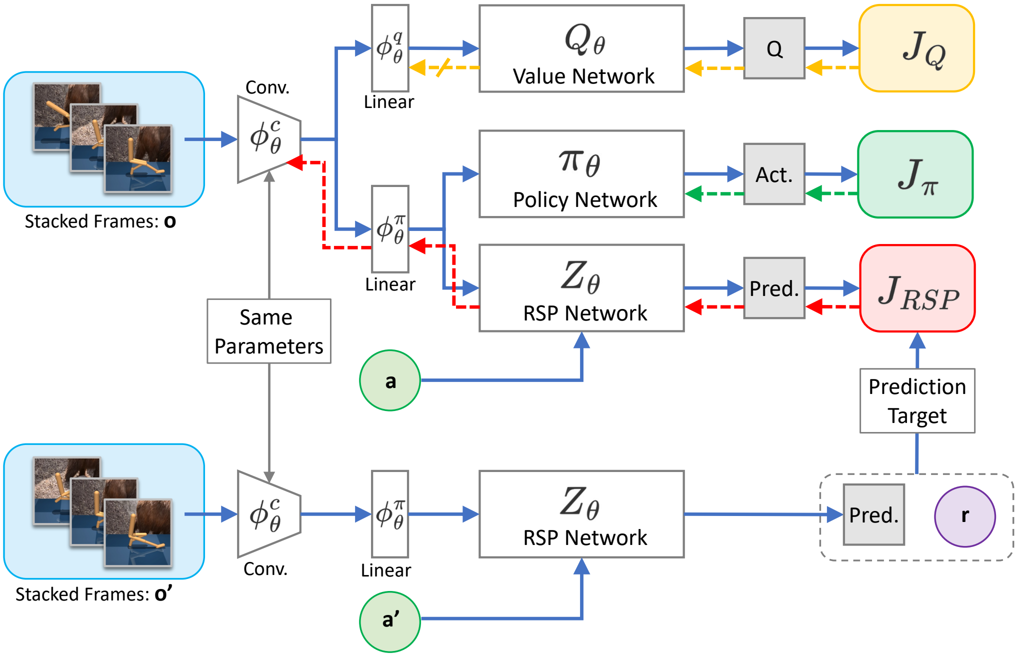


Figure 2: Combination of RSP and DrQ/DrQv2. The dashed line show how gradients flow back to model weights. We prevent the gradient of RL losses from updating the convnet. The action $\mathbf{a}'$ is sampled from a replay buffer. The policy and RSP network share a linear layer. ϕ_θ^q and ϕ_θ^π can share or not share parameters. We provide a comparison in Section 6.

According to Proposition 4.3, we need to control the infinity-norm of Γ to construct a transform of reward sequences. A simple method to achieve this is to express Γ by the product of $D \in \mathbb{R}^{L\times L}$ and $G \in \mathbb{R}^{L\times L}$, where D is a diagonal matrix, $|D_{ii}| < \gamma$ and $\sum_{j=0}^{L-1} |G_{ij}| = 1$ for any $0 \leq i < L$.

As the property of reward sequences varies in different tasks (e.g., sparse or dense rewards), we need to select the transforms (i.e., parameters W and Γ) of reward sequences for different tasks. To automate this process, we propose to learn transforms by maximizing the information in the prediction targets. First, similarly to the TD-learning, we compute the current prediction $Z_\theta(\mathbf{o}, \mathbf{a})$ and the prediction targets $\mathbf{z} = W\mathbf{r} + \Gamma Z_\theta(\mathbf{o}', \mathbf{a}')$. Then, we maximize the mutual information $I(Z_\theta(\mathbf{o}, \mathbf{a}); \mathbf{z})$ to maximizing the predictable information in the target $\mathbf{z}$. That is, we update parameters by InfoNCE loss Oord et al. [2018]:

$$J_{Trans} = \frac{1}{N}\sum_{n=1}^{N} \frac{\mathbf{Sim}(Z_\theta\left(\mathbf{o}_n, \mathbf{a}_n\right), \mathbf{z}_n)}{\sum_{m\neq n}^{N} \mathbf{Sim}(Z_\theta\left(\mathbf{o}_n, \mathbf{a}_n\right), \mathbf{z}_m)}. \tag{4}$$

Here, **Sim** stands for the exponential of cosine similarity.

5 ALGORITHM

This part introduces the overall algorithm (Algorithm 1) that combines our auxiliary tasks with DrQ/DrQv2. We let all networks share a convolutional encoder. We stop the gradients from the actor and the critic before they propagate to the shared convolutional layers. We only allow the gradients of J_{RSP} to update the shared encoder. This stop-gradient trick can stabilize the optimization in some tasks. To regularize the representation used by the policy network π_θ, we let it share the linear encoder ϕ_θ^π with the prediction network. We

Algorithm 1 RSP

```
Initialize the replay buffer $\mathcal{B} \leftarrow \emptyset$
Initialize the parameters $\theta$ of networks
for each episode do
    Sample actions: $\mathbf{a}_0 \sim \pi_\theta(\cdot|\mathbf{o}_0)$
    for each environment step do
        Obtain rewards: $\mathbf{r}_{t+1} \leftarrow R_o(\mathbf{o}_t, \mathbf{a}_t)$
        Sample observations: $\mathbf{o}_{t+1} \sim P_o(\cdot|\mathbf{o}_t, \mathbf{a}_t)$
        Sample actions: $\mathbf{a}_{t+1} \sim \pi_\theta(\cdot|\mathbf{o}_{t+1})$
        $\mathcal{B} \leftarrow \mathcal{B} \cup \{(\mathbf{o}_t, \mathbf{a}_t, \mathbf{r}_{t+1}, \mathbf{o}_{t+1}, \mathbf{a}_{t+1})\}$
    end for
    for each training step do
        Sample a batch of training data
        Compute RL losses $J_\pi$ and $J_Q$
        Compute $J_{RSP}$ by Equation (3)
        $\theta \leftarrow \theta - \lambda\nabla_\theta(J_\pi + J_Q + J_{RSP})$
        Optional: Compute $J_{Trans}$ by Equation (4)
        Optional: Minimize $J_{Trans}$ by gradient descent
    end for
end for
```

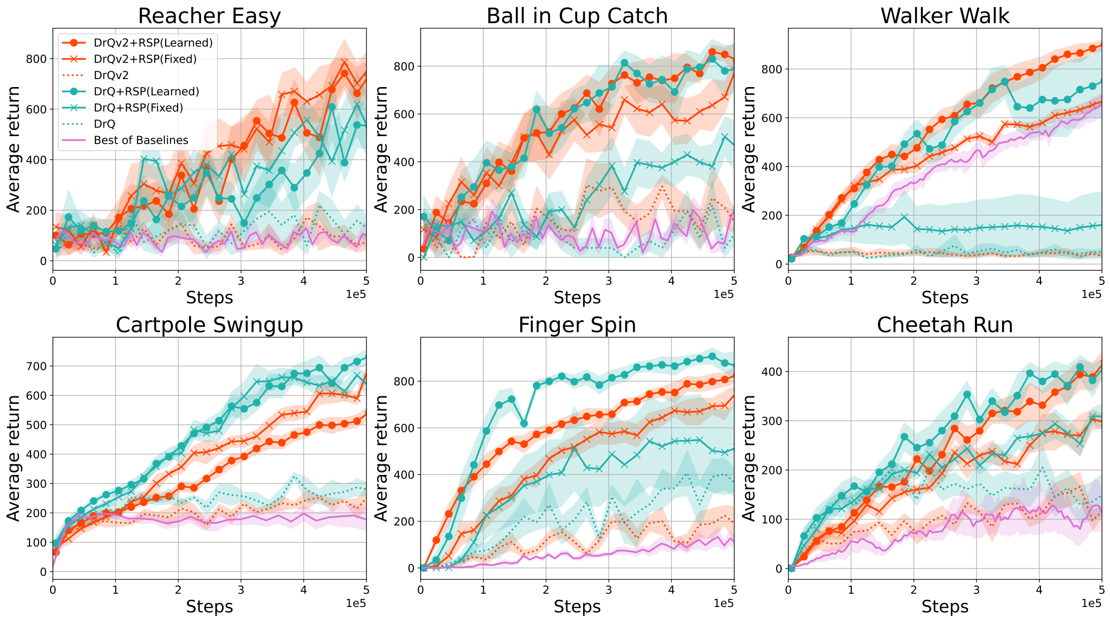


Figure 3: The performance during training. Here, we draw the line of "Best of Baselines" using the baseline that achieves the highest final score. The complete results are shown in Appendix 3.6. We use $Z_{\pi,1}(o, a)$ as prediction targets for environments in the first row and $Z_{\pi,2}(o, a)$ for those in the second row. RSP significantly improves the sample efficiency for both DrQ and DrQv2. Compared with DrQ/DrQv2, RSP(Learned) achieves **100% improvement of final score** in all six tasks.

illustrate the network architectures and the gradient flows in Figure 2. In this algorithm, we can choose either $Z_{\pi,1}(o, a)$ or $Z_{\pi,2}(o, a)$ as prediction targets $Z_{\pi}(o, a)$ according to the task. We can also learn the transform by the **Optional** steps in Algorithm 1 as discussed in Section 4.4. When computing prediction targets, we view the policy that collects data as the policy π that outputs the next action $\mathbf{a}'$. Therefore, we directly sample the action $\mathbf{a}'$ from the buffer.

6 EXPERIMENTS

This section evaluates RSP in Distracting Control Suite (DCS). Our experiments have three goals: 1) to test whether RSP can improve the sample efficiency and generalization in tasks with distractions; 2) to analyze the effect of each component in RSP; 3) to visualize the embedding space learned by RSP. All results are reported over five random seeds. We provide details of experiments in Appendix 3. We will release our code in https://github.com/QiZhou1997/MIRL.

Implementation We combine RSP with two algorithms, DrQ Yarats et al. [2020] and DrQv2 Yarats et al. [2021a]. We evaluate two variants of RSP. The first one use a fixed transform, $Z_{\pi,1}$ or $Z_{\pi,2}$ (selected by human). The second one learns the transform as discussed in Section 4.4. The hyperparameters of RSP can be found in Appendix 3.

6.1 MULTIPLE-DISTRACTION SETTING

This part evaluates RSP in sample efficiency and asymptotic performance. We compare RSP with four state-of-the-art methods that learn representations with distractions. The first three are based on auxiliary tasks, including DBC Zhang et al. [2021], which is a contrastive-based auxiliary task, TPC Nguyen et al. [2021], which learn a latent model without reconstruction, and TIA Fu et al. [2021], which is a reconstruction-based method with a pixel mask mechanism. The last one is SVEA Hansen et al. [2021b], which regularizes representations by strong data augmentation. We evaluate all methods in six visual control tasks, where agents face camera distractions, color distractions, and background distractions simultaneously.

We plot the results in Figure 3. The solid curves correspond to the mean, and the shaded region to the standard deviation. The results show that RSP beat all baselines in six tasks. The four representation learning methods do not beat DrQ/DrQv2, except that TIA outperforms DrQ and DrQv2 in Walker Walker. The poor performance of our baselines indicates the difficulty of learning robust representations with multiple distractions. In contrast, RSP improves the sample efficiency and asymptotic performance for DrQ and DrQv2 in all tasks, demonstrating that RSP can achieve sample-efficient and robust representation learning. Results also show that learning transforms can improve the sample efficiency of RSP in most tasks.

	BiC-Catch	C-Swingup	C-Run	F-Spin	R-Easy	W-Walk
DrQ	747 ± 28	582 ± 42	220 ± 12	646 ± 54	931 ± 14	549 ± 83
DBC	113 ± 133	296 ± 213	133 ± 98	154 ± 149	129 ± 64	119 ± 46
TPC	573 ± 182	706 ± 64	280 ± 48	634 ± 138	936 ± 61	768 ± 33
DrQ+CRESP	665 ± 185	689 ± 49	327 ± 54	778 ± 154	667 ± 82	794 ± 83
DrQ+PSE	$\mathbf{821 \pm 17}$	$\mathbf{749 \pm 19}$	308 ± 12	779 ± 49	955 ± 10	789 ± 28
DrQ+RSP(Learned)	730 ± 79	662 ± 47	$\mathbf{347 \pm 55}$	765 ± 84	$\mathbf{968 \pm 9}$	$\mathbf{829 \pm 58}$
DrQ+RSP(Fixed)	788 ± 78	$\mathbf{752 \pm 16}$	$\mathbf{338 \pm 27}$	$\mathbf{891 \pm 33}$	$\mathbf{960 \pm 15}$	$\mathbf{820 \pm 39}$

Table 1: Performance with unseen distractions at 500K steps. RSP can achieve state-of-the-art generalization.

6.2 EVALUATION IN GENERALIZATION

In this part, we consider two addtional baseliens, PSE Agarwal et al. [2021] and CRESP Yang et al. [2022]. They both achieve state-of-the-art generalization ability in tasks with distractions Agarwal et al. [2021]. We do not evaluate TIA in this part, as it is not concerned about generalization. We use the same settings as PSE and CRESP. For each episode, a video is sampled as background and keeps playing forwards or backwards. We use two videos for training and 30 unseen videos for evaluation. We also use DrQ as the backbone. We provide results in Table 1. The results show that DrQ+RSP achieves state-of-the-art generalization. Moreover, we find that using learned transform does not outperform using fixed transform in generalization. A potential reason is that maximizing information by the loss (4) may cause networks to capture extra task-irrelevant information.

6.3 STOCHASTIC REWARDS

Standard Deepmind Control environments usually use deterministic reward functions. Here, we provide results to test whether RSP can improve performance when rewards are stochastic. We consider two kinds of stochasticity, random delay and Gaussian noise. In the random delay setting, agents receive reward signals 0-3 steps after performing actions. In the Gaussian noise setting, we add Gaussian noise with a standard deviation of 0.1 on reward signals. In all environments, multiple distractions exist. The results in Figure 4 show that RSP can significantly improve the performance of DrQv2 even when rewards are stochastic.

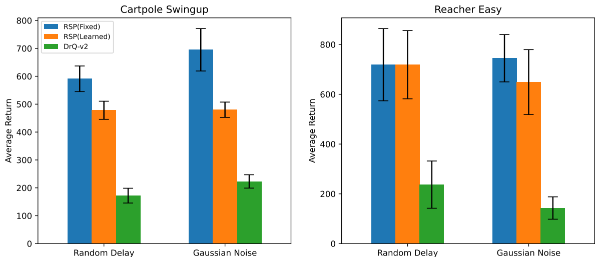


Figure 4: performance in tasks with stochastic rewards.

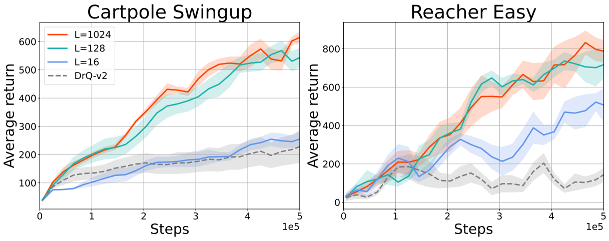


Figure 5: Training curves with different L. Here, L stands for the number of dimensions of predicted targets.

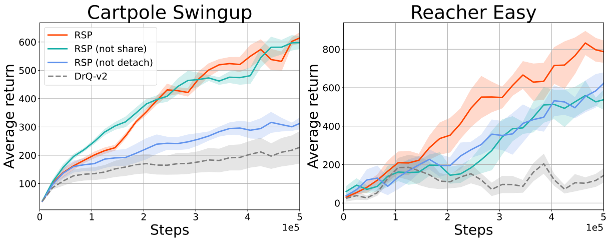


Figure 6: Comparison between different implementations of RSP. Here, "not share" means that the policy network and the value network do not share the linear encoder layer. "not detach" means that we do not prevent the gradients of J_Q from updating the encoder.

6.4 ABLATION STUDY

This part provides ablation studies in multi-distraction settings. We use fixed transforms if not otherwise stated.

Hyperparameter L: The hyperparameter L control the dimension of the prediction targets $Z_\pi(o, a)$. Figure 5 shows that RSP performs better as L becomes larger no matter whether $Z_{\pi,1}$ or $Z_{\pi,2}$ is used. The reason is that high-dimensional prediction targets provide more information than low-dimensional counterparts. As large L does not cause obvious cost, we suggest set $L = 1024$.

Implementation: We prevent the gradient of RL losses from updating the encoder and let all networks share one linear encoder layer. Figure 6 shows that the two tricks are important in multi-distraction settings. A possible reason

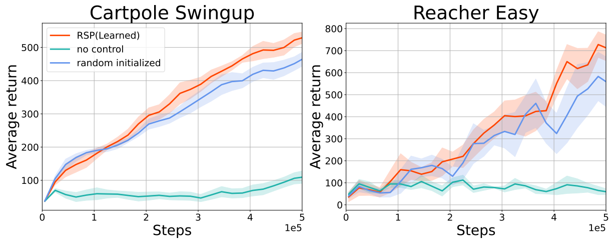
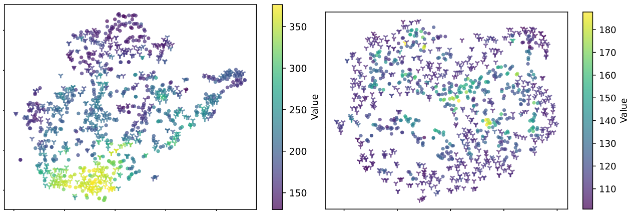


Figure 7: Ablation study for learning transforms.



(a) RSP

(b) CPC+Reward

Figure 8: T-SNE of embedding spaces after 200K-step training. The color represents the predicted state values. Different markers represent different background images. Neighboring points in the embedding space learned by RSP have similar state values, whereas no such structure is seen in the embedding learned by CPC+Reward.

is that training more encoder layers by only the gradient of J_{RSP} brings more stable representation learning.

Learning transforms: We provide an ablation study for the transform learning method. Results in Figure 7 show that the performance will significantly degrade if without controlling the infinity-norm of Γ (the "no control" line). Results also show that updating the parameters W and Γ outperforms using the randomly initialized ones (the "random initialized" line), demonstrating the effectiveness of learning transforms by maximizing information.

6.5 VISUALIZATION

This part visualize the embeddings with t-SNE. We use the data that used in Section 4.1. Figure 8 shows that RSP maps observations with similar values to neighboring regions while CPC+Reward does not. This means that RSP can better extract task-relevant information from raw observations compared with CPC+Reward. Moreover, CPC+Reward tends to map observations with different background images to different regions while RSP does not. This means that representations learned by CPC+Reward encode more task-irrelevant information than those learned by RSP.

7 CONCLUSION

Learning behaviors with distractions have been a long-standing challenge. To address this challenge, we introduce RSP, a novel method that learns robust presentations by predicting reward sequences. We compare different methods by estimating the information in representations. Compared with prior methods, representations learned by RSP encode more task-relevant information while containing less task-irrelevant information. We evaluate RSP in both multi-distraction and video-background settings. Experiments demonstrate that RSP can achieve state-of-the-art sample efficiency and generalization. A promising future research is combining state abstraction with RSP to filter task-irrelevant information further.

Acknowledgements

We would like to thank all the anonymous reviewers for their insightful comments. This work was supported in part by National Science Foundations of China grants U19B2026, U19B2044, 61836006, 61836011, and 62021001, and the Scientific and Technological Innovation 2030—“New Generation Artificial Intelligence” Major Project 2022ZD0119801.

References

Rishabh Agarwal, Marlos C Machado, Pablo Samuel Castro, and Marc G Bellemare. Contrastive behavioral similarity embeddings for generalization in reinforcement learning. *International Conference on Learning Representations*, 2021.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint*, 2019.

Pablo Samuel Castro. Scalable methods for computing state similarity in deterministic markov decision processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Jianda Chen and Sinno Jialin Pan. Learning generalizable representations for reinforcement learning via adaptive meta-learner of behavioral similarities. *arXiv preprint arXiv:2212.13088*, 2022.

Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, 2019.

Fei Deng, Ingook Jang, and Sungjin Ahn. Dreamerpro: Reconstruction-free model-based reinforcement learning with prototypical representations. *arXiv preprint arXiv:2110.14565*, 2021.

Ben Eysenbach, Russ R Salakhutdinov, and Sergey Levine. Robust predictable control. *Advances in Neural Information Processing Systems*, 2021.

Jiameng Fan and Wenchao Li. Robust deep reinforcement learning via multi-view information bottleneck. *arXiv preprint*, 2021.

Linxi Fan, Guanzhi Wang, De-An Huang, Zhiding Yu, Li Fei-Fei, Yuke Zhu, and Anima Anandkumar. Secant: Self-expert cloning for zero-shot generalization of visual policies. *International Conference on Machine Learning*, 2021.

Jesse Farebrother, Marlos C Machado, and Michael Bowling. Generalization and regularization in dqn. *arXiv preprint*, 2018.

William Fedus, Carles Gelada, Yoshua Bengio, Marc G Bellemare, and Hugo Larochelle. Hyperbolic discounting and learning over multiple horizons. *arXiv preprint*, 2019.

Norm Ferns, Prakash Panangaden, and Doina Precup. Bisimulation metrics for continuous markov decision processes. *SIAM Journal on Computing*, 2011.

Norman Ferns and Doina Precup. Bisimulation metrics are optimal value functions. In *UAI*, 2014.

Xiang Fu, Ge Yang, Pulkit Agrawal, and Tommi Jaakkola. Learning task informed abstractions. In *International Conference on Machine Learning*, 2021.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, 2018.

Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G Bellemare. Deepmdp: Learning continuous latent space models for representation learning. In *International Conference on Machine Learning*, 2019.

Dibya Ghosh, Jad Rahme, Aviral Kumar, Amy Zhang, Ryan P Adams, and Sergey Levine. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability. *Advances in Neural Information Processing Systems*, 2021.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint*, 2019a.

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

Nicklas Hansen and Xiaolong Wang. Generalization in reinforcement learning by soft data augmentation. *arXiv preprint arXiv:2011.13389*, 2020.

Nicklas Hansen, Rishabh Jangir, Yu Sun, Guillem Alenyà, Pieter Abbeel, Alexei A Efros, Lerrel Pinto, and Xiaolong Wang. Self-supervised policy adaptation during deployment. *International Conference on Learning Representations*, 2021a.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Stabilizing deep q-learning with convnets and vision transformers under data augmentation. *Advances in Neural Information Processing Systems*, 34, 2021b.

Maximilian Igl, Kamil Ciosek, Yingzhen Li, Sebastian Tschiatschek, Cheng Zhang, Sam Devlin, and Katja Hofmann. Generalization in reinforcement learning with selective noise injection and information bottleneck. *Advances in Neural Information Processing Systems*, 2019.

Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint*, 2016.

Nan Rosemary Ke, Amanpreet Singh, Ahmed Touati, Anirudh Goyal, Yoshua Bengio, Devi Parikh, and Dhruv Batra. Learning dynamics model in reinforcement learning by incorporating the long term future. *arXiv preprint*, 2019.

Robert Kirk, Amy Zhang, Edward Grefenstette, and Tim Rocktäschel. A survey of generalisation in deep reinforcement learning. *arXiv preprint*, 2021.

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

Alex X Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *arXiv preprint*, 2019a.

Kimin Lee, Kibok Lee, Jinwoo Shin, and Honglak Lee. Network randomization: A simple technique for generalization in deep reinforcement learning. *arXiv preprint*, 2019b.

Lihong Li, Thomas J Walsh, and Michael L Littman. Towards a unified theory of state abstraction for mdps. *ISAIM*, 2006.

Guoqing Liu, Chuheng Zhang, Li Zhao, Tao Qin, Jinhua Zhu, Jian Li, Nenghai Yu, and Tie-Yan Liu. Return-based contrastive representation learning for reinforcement learning. *International Conference on Learning Representations*, 2021.

Xingyu Lu, Kimin Lee, Pieter Abbeel, and Stas Tiomkin. Dynamics generalization via information bottleneck in deep reinforcement learning. *arXiv preprint*, 2020.

Dhruv Malik, Yuanzhi Li, and Pradeep Ravikumar. When is generalizable reinforcement learning tractable? *arXiv preprint*, 2021.

Tung D Nguyen, Rui Shu, Tuan Pham, Hung Bui, and Stefano Ermon. Temporal predictive coding for model-based planning in latent space. In *International Conference on Machine Learning*, 2021.

Masashi Okada and Tadahiro Taniguchi. Dreaming: Model-based reinforcement learning by latent imagination without reconstruction. In *International Conference on Robotics and Automation*, 2021.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint*, 2018.

Alan V Oppenheim. *Discrete-time signal processing*. Pearson Education India, 1999.

Roberta Raileanu and Rob Fergus. Decoupling value and policy for generalization in reinforcement learning. *International Conference on Machine Learning*, 2021.

Roberta Raileanu, Max Goldstein, Denis Yarats, Ilya Kostrikov, and Rob Fergus. Automatic data augmentation for generalization in deep reinforcement learning. *arXiv preprint*, 2020.

Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. *arXiv preprint*, 2020.

Anoopkumar Sonar, Vincent Pacelli, and Anirudha Majumdar. Invariant policy optimization: Towards stronger generalization in reinforcement learning. In *Learning for Dynamics and Control*, 2021.

Xingyou Song, Yiding Jiang, Stephen Tu, Yilun Du, and Behnam Neyshabur. Observational overfitting in reinforcement learning. *The International Conference on Learning Representations*, 2019.

Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint*, 2020.

Austin Stone, Oscar Ramirez, Kurt Konolige, and Rico Jonschkowski. The distracting control suite–a challenging benchmark for reinforcement learning from pixels. *arXiv preprint*, 2021.

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

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *IEEE Information Theory Workshop*, 2015.

Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Matias Vera, Pablo Piantanida, and Leonardo Rey Vega. The role of the information bottleneck in representation learning. In *International Symposium on Information Theory*, 2018.

Rui Yang, Jie Wang, Zijie Geng, Mingxuan Ye, Shuiwang Ji, Bin Li, and Feng Wu. Learning task-relevant representations for generalization via characteristic functions of reward sequence distributions. *SIGKDD*, 2022.

Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2020.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint*, 2021a.

Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021b.

Tao Yu, Cuiling Lan, Wenjun Zeng, Mingxiao Feng, and Zhibo Chen. Playvirtual: Augmenting cycle-consistent virtual trajectories for reinforcement learning. *Advances in Neural Information Processing Systems*, 2021.

Amy Zhang, Zachary C Lipton, Luis Pineda, Kamyar Azizzadenesheli, Anima Anandkumar, Laurent Itti, Joelle Pineau, and Tommaso Furlanello. Learning causal state representations of partially observable environments. *arXiv preprint*, 2019.

Amy Zhang, Clare Lyle, Shagun Sodhani, Angelos Filos, Marta Kwiatkowska, Joelle Pineau, Yarin Gal, and Doina Precup. Invariant causal prediction for block mdps. In *International Conference on Machine Learning*, 2020.

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

Hanping Zhang and Yuhong Guo. Generalization of reinforcement learning with policy-aware adversarial data augmentation. *arXiv preprint*, 2021.

Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. *arXiv preprint*, 2021.

Jinhua Zhu, Yingce Xia, Lijun Wu, Jiajun Deng, Wengang Zhou, Tao Qin, and Houqiang Li. Masked contrastive representation learning for reinforcement learning. *arXiv preprint*, 2020.