INVARIANCE THROUGH INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce a general approach, called Invariance through Inference, for improving the test-time performance of an agent in deployment environments with unknown perceptual variations. Instead of producing invariant visual features through memorization, invariance through inference turns adaptation at deployment-time into an unsupervised learning problem. This is achieved in practice by deploying a straightforward algorithm that tries to match the distribution of latent features to the agent's prior experience, without relying on paired data. Although simple, we show that this idea leads to surprising improvements on a variety of adaptation scenarios without access to deployment-time rewards, including changes in camera poses and lighting conditions. Results are presented on challenging distractor control suite, a robotics environment with image-based observations.

1 INTRODUCTION

Let us consider the ability of an intelligent agent to generalize to unseen domains. To have such a discussion, we must first consider what generalization means. In much of the learning literature, we typically assume that an agent will accrue experiences of sufficient variation during training, and that these experiences will allow the agent to generalize to novel settings during deployment. Since the richness of the agent's experience is of paramount importance to the quality of its generalization, there exist a broad family of methods that expand the support distribution of the agent's training set. Domain transfer, domain randomization, and meta learning all fall into this category. While these approaches are powerful, they are often prone to failure in the case where the target domain is not known *a priori*. In this case, if the target domain falls out-of-distribution, the agent will lack the ability to self correct and will fail, even if great pains were taken to significantly widen the size of its training support. We refer to this approach, which aims to improve an agent's generalization by increasing the richness of its experiences, as in-distribution generalization.

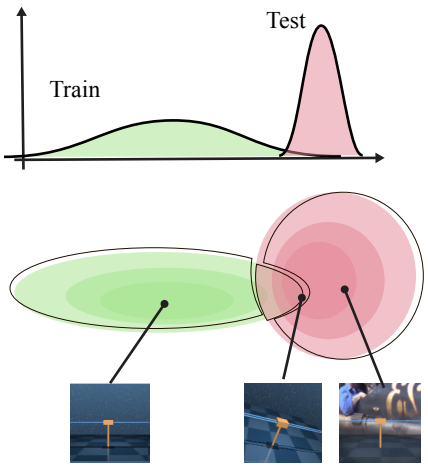


Figure 1: Methods such as data augmentation try to make the training distribution large at the expense of complexity and performance. In spite of these efforts, these methods still often fail to adequately cover the target distribution.

A more challenging problem—but also more realistic—is generalizing out-of-distribution. In this setting, we assume the specifics of the agent's deployment are not known ex-ante. Taking this even further, we assume that access to instrumented reward supervision is unavailable in the deployment environment. In this setting, the burden placed on the agent hoping to generalize is significant. Since we lack a reward function, fine-tuning via reinforcement learning is impossible. Prior experience will help, but a truly out-of-distribution environment of which the agent has no prior knowledge will make the transfer problem non-trivial. While difficult, this setting of out-of-distribution generalization is nevertheless evergreen in fields such as robotics, wherein a robot trained from pixel observations in the factory will inevitably encounter new settings with different lighting conditions, environmental objects, physical dynamics, and sensor modalities during deployment.

To make progress in this setting of out-of-distribution generalization, we consider how we can best leverage the information we do have: the agent's prior experiences during training. Specifically, if we assume the original task the agent was optimized for during training remains well-defined in the target domain, then it is also safe to assume that the agent does have some understanding of the underlying Markov decision process in the test domain. Without further assumptions or loss of generality, we may recast the out-of-distribution generalization problem into an unsupervised policy adaptation between two MDPs that share a similar latent dynamics and reward structure, but with distinct observations.

In this paper, we investigate ways to improve generalization under this challenging scenario. Rather than producing policy invariance by explicitly baking this property into training, we harness probabilistic inference to produce invariance by taking advantage of the latent structure the agent has already discovered in the environment. As we will see, this pays great dividends in producing robotic agents that can function effectively in deployment environments that exhibit different lighting, object texture, or camera poses. We also demonstrate the ability of our method to generalize across environments with different forward dynamics, so long as the distinct dynamics maintain some semblance of similarity. To distinguish our approach from prior works that attempt to produce generalization by baking invariances into the policy at training time, we refer to our method as "*Invarivance Through Inference*."

2 Related Work

Generalization in reinforcement learning is a longstanding problem. Recent work has shown that accurately quantifying generalization in this setting poses a challenge (Lee et al., 2019; Cobbe et al., 2019; Zhang et al., 2018). This challenge is further amplified in visual RL from pixels, where measuring the difference between two problems is especially difficult. One promising class of techniques for generalization in image-based RL utilize image augmentation to greatly increase the size of the training distribution. These techniques have enjoyed success in vision research (Chen et al., 2020a; He et al., 2020). However, it was only recently shown that such data augmentation leads to instability in value function estimation in RL algorithms (Hansen & Wang, 2020; Hansen et al., 2021a). This line of work showed how to stabilize data-augmented RL, which lead to a significant increase in the generalization capabilities of RL algorithms. The primary shortcoming of these methods is that it is largely impossible to effectively cover the entire potential test distribution by expanding the training distribution, and some test time adaptation is often needed. Several recent works (Hansen et al., 2020b; Wang et al., 2020) consider this setting of adaptation at test time, and are further discussed in Section 3.

Invariant Representation Learning is a broad class of techniques for learning representations that are robust across changes in the agent's environment or reward. Often, the agent will use auxiliary information related to dynamics (Hafner et al., 2020) or reward prediction (Jaderberg et al., 2016) to learn a latent representation that is invariant across changing environmental conditions. This is closely related to the idea of **self-supervised learning** (Chen et al., 2020b; Grill et al., 2020). When applied to visual space RL, it has been shown that self-supervised pre-training provides an efficient way to bootstrap useful invariant representations, even in the absence of a ground-truth reward (Schwarzer et al., 2020; Srinivas et al., 2020).

Domain Shift considers the problem of test-time shift in the underlying input data distribution (Ganin et al., 2016). One common class of techniques for addressing this problem is to learn representations that maximize domain confusion, thus requiring that the agent learn domain-invariant features. This idea has been applied successfully in imitation learning (Stadie et al., 2017). More recent advances have combined Cycle-GANs with imitation learning to produce features that are cycle consistent across domains (Smith et al., 2019). Another approach bottlenecks the information on the latent code of a VAE-based architecture to learn features that are more robust across domain shift in a variety of adversarial settings (Peng et al., 2018).

Meta Learning tries to solve generalization by aggregating learning across distributions of tasks (Finn et al., 2017; Duan et al., 2016; Rakelly et al., 2019). Often, meta learning pipelines focus heavily on fast inference at test time. This is most often accomplished through using some form of bi-level optimization (Rothfuss et al., 2018; Houthooft et al., 2018). Unfortunately, meta learning methods generally require training over entire task distributions. Worse still, this results

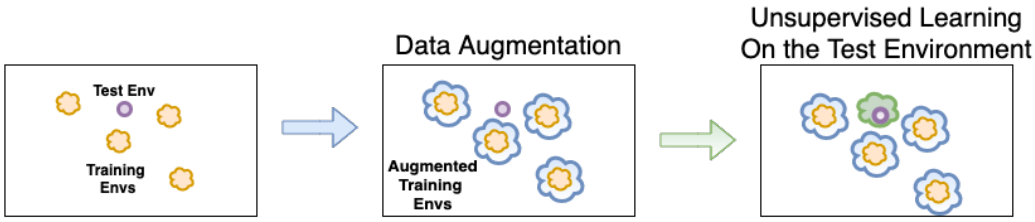


Figure 2: Expanding the support distribution over the observation space makes the resulting policy much more robust to observation perturbations at test time (Data Augmentation). However, the desired test time observation space is a comparatively small set. No matter how much we expand the support of the input training distribution, there is a chance we will miss the shift captured by the test environment. To avoid this, we consider the problem of how to best carry out additional unsupervised learning at test time.

in the need for a distribution of reward functions for each task during both training and test time. Defining even a single reward function is often quite difficult, let alone many closely related reward functions. Recent work has tried to alleviate this burden with unsupervised training (Hsu et al., 2018).

3 Problem Formulation

Consider the standard infinite horizon Markov decision process (MDP) (Puterman, 2014)) parameterized via the tuple $\mathcal{M} = \langle S, O, A, R, P, \gamma \rangle$, where S, O, and A are the state, observation, and action spaces, respectively, $P : S \times A \mapsto S$ is the transition function, $R : S \times A \mapsto \mathbb{R}$ is the scalar reward, and γ is the discount factor. Recall that the state space S usually represents ground truth information about the agents environment which we assume the agent can not access. The observations space O is the information stream the agent receives about the environment. The goal of reinforcement learning is to solve for the policy $\pi : O \times A \mapsto [0, 1]$ that maximizes the expected discounted return $\mathcal{J} = \mathbb{E} \sum_{\infty} \gamma^t R(o_t, a_t)$ over infinite horizon, represented as the $Q^\pi(o, a)$ function.

In this paper, we consider a setting wherein there are two distinct MDPs: A train MDP $\mathcal{M}_{\text{train}}$ and a test MDP $\mathcal{M}_{\text{test}}$. During training, the agent has access to $\mathcal{M}_{\text{train}}$ and can collect trajectory data $\rho_{\text{train}} = (o_i, a_i, r_{i+1}, \ldots)$, where $o_i \in O$, $a_i \in A$, and $r_{i+1} \in R$ is the corresponding reward. During testing, the agent is placed into the previously unseen $\mathcal{M}_{\text{test}}$ environment. Crucially, during the testing phase, we will assume that the agent can only observe $(o_{\text{test}}, a_{\text{test}}) \sim \mathcal{M}_{\text{test}}$ i.e., that the agent has no access to the reward function at test time. The goal of the agent is to leverage the data it collected from $\mathcal{M}_{\text{train}}$, as well as the unsupervised trajectory data $(o_1, a_1, \ldots, o_t, a_t)$ to perform fast inference in the testing environment:

$$\mathcal{J} = \mathbb{E}_{\substack{\rho_{\text{train}} \sim \pi(\mathcal{M}_{\text{train}}) \\ o_{t+1} \sim \pi(o_t, a_t; \rho_{\text{train}})}} \left[r_{\text{test}}(o_{t+1}) \right], \tag{1}$$

where $o_{t+1} \sim \pi(o_t, a_t; \rho_{\text{train}})$ makes clear the dependence of the policy π on data from the training environment ρ_{train}. Note that this objective is for evaluation purposes only, because we do not have access to r_{test} during learning. In the objective, we have used the abuse of notation $o_{t+1} \sim \pi(o_t, a_t; \rho_{\text{train}})$ instead of the correct $o_{t+1} \sim P(o_t, a_t), a_t \sim \pi(o_t; \rho_{\text{train}})$, since the next observation is sampled by following along the policy and space was limited.

Often, we are most interested in the case where shift between $\mathcal{M}_{\text{train}}$ and $\mathcal{M}_{\text{test}}$ is induced by differences in the observation spaces O_{train} and O_{test}. That is, the state and reward structures between the train and test MDPs are quite similar, but the observation spaces between the two MDPs are significantly different. Practically speaking, this can happen quite easily in the presence of distractors, corrupted or malformed inputs, or shifts in sensor readings at test time. Consider a robot that is trained from image observations in a clean environment, and then expected to perform the same task in the wild, where changing lighting conditions and environmental conditions might lead to significantly different image observations. Recent work has tried to handle this shift between training and

testing environments by greatly expanding the support of the training distribution during training (Hansen & Wang, 2020). This is most often accomplished via artificial data augmentation. Image observations are cropped, shifted, rotated, and discolored to make learned policies more robust to shifts in the input observation at test time. In RL, this type of augmentation can cause value function estimation to become unstable (Laskin et al., 2020; Raileanu et al., 2020; Kumar et al., 2021), and so care must be taken to avoid destabilizing training.

While data augmentation techniques have greatly improved the robustness of RL algorithms to observation shift at test time, potential issues linger. In Figure 2, we see a cartoon illustrating the primary worry. No matter how much the support over the input observation space is expanded, the size of the target testing observation space is very small in comparison to the overall size of all observation spaces. Thus, there is still a fairly reasonable chance that even extreme data augmentation measures will fail to adequately cover the test MDP's observation space. Indeed, the results in Section 5 show this to often be the case—even the most robust data augmentation schemes fail when the test MDP's observation space differs substantially from that of training. In the next section, we will develop a technique, Invariance Through Inference, that enables quick test-time adaptation onto the target observation space by utilizing only unsupervised transition pairs.

4 Invariance through Inference

4.1 A loss function for Invariance through Inference

Following the previous section, assume we are in a setting with two MDPs, $\mathcal{M}_{\text{train}}$ and $\mathcal{M}_{\text{test}}$. We assume these MDPs share similar states, dynamics, and reward structures. However, the observation spaces incur a significant distributional shift. Nevertheless, since the underlying state and dynamics that are relevant to the task are similar, there exists some encoding $z = G_\theta(o, a)$ s.t.

$$\|s_{\text{train}} - s_{\text{test}}\| \leq \delta \rightarrow \|G_\theta(o_{\text{train}}, a_{\text{train}}) - G_\theta(o_{\text{test}}, a_{\text{test}})\| \leq \epsilon$$

In other words, we want to learn a latent encoder that maps semantically similar observations to close latent vectors, learning to ignore irrelevant distractions in the observation space. In many real world deployment scenarios, test-time rewards R_{test} are unavailable. Thus, we have hope to train this encoder without access to test-time rewards. Given this constraint, we note that the shared underlying dynamics are the most feature-rich and informative source of mutual information between the two environments. We would therefore like our encoded latents to minimize an inverse dynamics model loss

$$\mathcal{J}_{\text{inv}} = \mathbb{E}_{\substack{o_{t+1} \sim \pi(o_t, a_t) \\ z_t = G_\theta(o_t) \\ z_{t+1} = G_\theta(o_{t+1})}} \left[\left\| P_\phi(z_t, z_{t+1}) - a_t \right\|^2 \right] \tag{2}$$

However, simply training this model with data from both MDPs will not ensure that observations with similar ground-truth states are actually aligned. Instead, the encoder might simply learn to handle each MDP independently. Critically, we do not assume access to paired trajectories, which can be difficult to achieve in practice. If we had access to such paired data, we could simply train an encoder that minimizes the distance to both paired observations. At minimum, we would like to ensure that the latent distributions over observations embeddings are similar for both MDPs, since that would hopefully force the encoder to filter out superfluous distractions that separate the two MDPs. Towards this goal, we define two objectives. First, a distribution matching objective encourages the latent distribution induced by both MDPs to match.

$$\mathcal{J}_{\text{d}} = \arg\min_{D} \left[\log P(z) + (1 - \log P(\tilde{z})) \right] \tag{3}$$

Second, we employ a GAN-style loss to help ensure that the latent code is not using distracting information that would help distinguish the train and test MDPs

$$\mathcal{J}_D = \mathbb{E}_{o_t \sim D_{\text{non-distr}}} \left[\log D(\bar{G}_\theta(o_t)) \right] + \mathbb{E}_{o_t \sim D_{\text{distr}}} \left[\log(1 - D(G_\theta(o_t))) \right] \Bigg], \tag{4}$$

where $\bar{G}$ denotes the encoder whose weights are frozen after training. The final objective can now be written as

$$\arg\min_{G_\theta} \ \arg\max_{D} \left[\mathcal{J}_D + \mathcal{J}_{\text{inv}} \right] \tag{5}$$

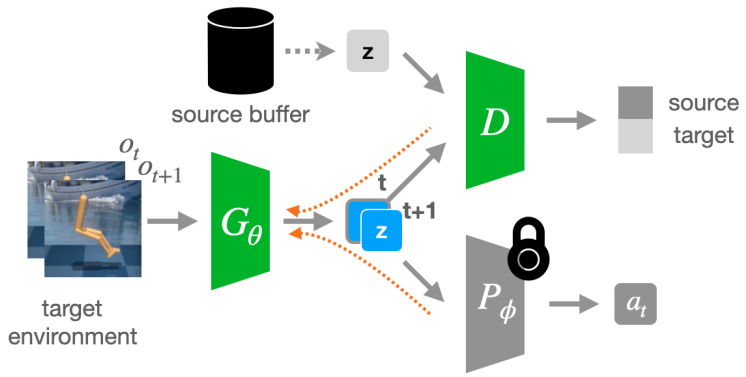


Figure 3: Our proposed Invariance through Inference architecture. The encoder G takes an observation from target environment, and learns to trick the discriminator, while the discriminator predicts whether the input comes from the encoder or source buffer. This adversarial training encourages the distribution of encoder outputs to be similar to the latent embedding sampled from the source buffer. The inverse dynamics network is pretrained and used only to guide the encoder during adaptation.

This adaptation objective is effectively maximizing the mutual information between the prototypical representation kept from during training, and the adapted representations of the new observations. In summary, in contrast to those methods that produce invariance through memorization, our adaptation objective produce invariance by making statistical inference in the latent space. For this reason we refer to the derived method as Invariance through Inference.

4.2 Algorithm Overview

Constructing a pipeline to minimize Equation 5 is fairly straightforward. The architecture is fully represented in Figure 3. In practice, the algorithm proceeds as follows. Start by collecting trajectories in the clean training environment and adding the corresponding encoded observations into a replay buffer. These initial samples are collected under an exploratory policy. In practice, we find random exploration is sufficient for many robotics environments.

Now that the buffer is nonempty, we can draw samples from it to pretrain the inverse dynamics network. With this in hand, we can start adapting the encoder towards the test environment. Since we want to ensure the observation embeddings for the train and test environment follow a similar distribution, we use the optimization framework from Generative Adversarial Networks (GANs) to constrain the encoder appropriately. Our encoder, which is analogous to a generator, is encouraged to produce similar embeddings on both train and test environment observations. The discriminator learns to classify whether a given input comes from source environment buffer or target environment.

Finally, we need a structure that can help guide the encoder to produce a meaningful embedding for each observation. To wit, we add an inverse dynamics head onto our architecture as a type of auxiliary task, under the assumption that train and test time environments share a high degree of overlap in their environmental dynamics. This head is pre-trained with transitions sampled from the source buffer, and has its weights fixed during adaptation.

5 Experiments

We want to understand the impact of test-time inference on an agent's ability to perform out-of-distribution generalization. To wit, this section will compare Invariance through Inference with two state-of-the-art methods for data-augmented reinforcement learning: SVEA (Hansen et al., 2021b) and DrQ-v2 (Yarats et al., 2021). Recall from the introduction that these methods vie for increased generalization capabilities by expanding the support of the training distribution. We expect this form of generalization to be less performant than inference at test time. In our experiments, we will also compare against Policy Adaptation during Deployment (PAD) (Hansen et al., 2020a), a baseline that, like our method, adapts the policy without access to the reward at test time.

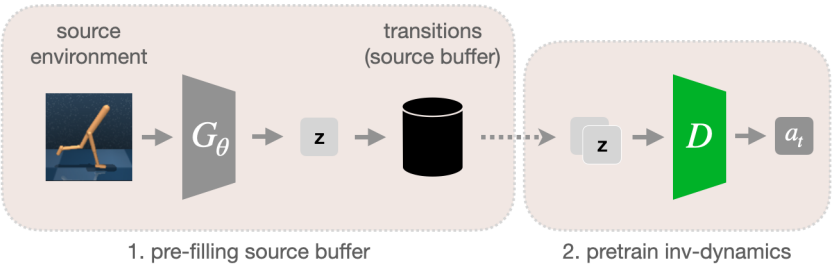


Figure 4: Preprocessing for Invariance through Inference. The first stage pre-fills the source buffer with observation embeddings. The second stage then pre-trains the inverse dynamics head using samples from the buffer.

To further probe the generalization abilities of test-time inference, we conduct an experiment wherein we vary the intensity of environmental distractions. The upshot here is that test-time inference significantly increases the robustness of our policy to distractions during deployment. We will conclude with some general discussion and remarks regarding the design tradeoffs involved in test-time inference.

Setup We conduct experiments using five domains from the DeepMind Control Suite (DMC) (Tassa et al., 2018). We use DMC as a clean training environment the Distracting Control Suite (DistractingCS) (Stone et al., 2021) for the distracted test environment. DistractingCS adds three types of distractions to the DeepMind Control Suite through deviations in background, camera pose, and color.

Modifications to DistractingCS The default configuration of DistractingCS changes distractions at the start of every episode (e.g., different background images are used at every episode). However, we are interested in measuring an agent's ability to perform inference across several episodes on the same target environment. Thus, we modify DistractingCS to sample a distraction once in the beginning of learning, and then use the same distraction across all learning epochs. This also ensures consistent evaluation across algorithms. In accordance with this change, we also modify the intensity benchmark from DistractingCS. In our experiments, intensity measures the deviation between an environment distraction and the train environment's default value. For example, intensity may measure how far the distracting color is from the default. Finally, we modify the environments to only apply a single distraction during testing (rather than all three) in order to better understand the impact of each type of distraction on overall performance.

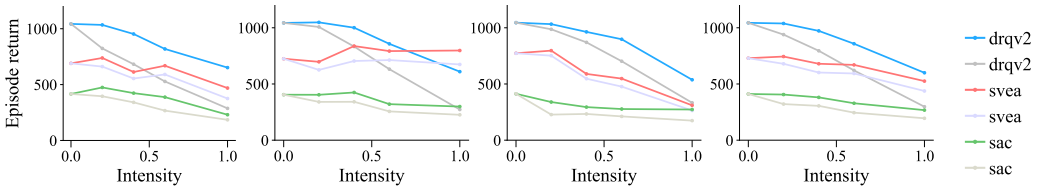
5.1 Invariance through Inference On DMControl

This section studies the impact of test-time adaptation on DMControl performance. We begin by pre-trained SAC, SVEA, and DrQ-v2 in a non-distracting training-time environment. After training, we evaluate the learned policies on distracted test environments of varying intensities. This evaluation is zero-shot, there is no additional training on the test environment.

Figure 5 visualizes the performance of the different methods as a function of the intensity of the distractions. Since the baseline methods are trained with image augmentation, they do exhibit some robustness to distraction. However, we see this robustness rapidly diminishes as the distraction intensity increases. In particular, large changes to camera pose or the image background proved challenging for standard augmentation procedures. Comparatively, Invariance through Inference saw much smoother and slower degradation of performance. This supports our hypothesis that adaptation powered by unsupervised learning can significantly widen the generalization abilities of learning algorithms.

5.2 Comparisons with PAD

Similar to our approach, PAD pre-trains the agent in a clean environment, and then adapts the agent via unsupervised transfer, without assuming access to the target environment's reward function



(a) Background (b) Colors (c) Camera Pose (d) Combined

Figure 5: The gain of applying Invariance through Inference to various distracted baselines. Lines in color denote base agent + ITI (our method), whereas gray and unsaturated lines correspond to the base agent's performance.

Hansen et al. (2020a). To evaluate the robustness of PAD to distractions, we consider DistractingCS with a fixed distraction intensity of 0.6. Table 1 compares the performance as the difference between the reward before and after adaptation along with the reward in the clean environment.

Across all environments, we see that PAD struggles to adapt to distractions at test time. We suspect this instability is caused by the large deviations in the latent variable distribution as a result of changes in the target environment. In particular, we posit that the signal from PAD's inverse dynamics head does not encourage the latent train and test distributions to match, which is a feature specifically baked into Invariance through Inference.

Table 1: Comparison against PAD

	Before Adaptation		After Adaptation	
Distraction	PAD	ITI	PAD	ITI
None	851.65 ± 102.22	873.52 ± 198.82	—	—
Camera	461.55 ± 256.83	$\mathbf{702.30 \pm 281.46}$	418.62 ± 231.91	$\mathbf{820.03 \pm 232.57}$
Background	377.53 ± 324.54	$\mathbf{528.14 \pm 374.13}$	396.90 ± 271.48	$\mathbf{736.34 \pm 354.54}$
Color	437.48 ± 344.91	$\mathbf{631.84 \pm 354.88}$	452.31 ± 337.98	$\mathbf{787.10 \pm 315.75}$

5.3 Further Discussion

Unsupervised Pipeline Ablations To verify that the adversarial training scheme actually contributes to the performance gain during adaptation, we run Invariance through Inference while setting either the adversarial training losses to 0. This corresponds to an approach that relies purely on inverse dynamics models to drive the latent code.

In these experiments, we use a pre-trained DrQ-v2 for the algorithm's base policy, and then run adaptation across all distractions with intensity 0.6. The results in Table 2 show that the adversarial training is critical for the unsupervised pipeline to learn properly. We theorize that the inverse dynamics head has a tendency to simply memorize dynamics in the training and testing domains without collapsing the commonalities, i.e. it memorizes rather than learns. The adversarial training component insures that the learning is projected onto a latent space that is agnostic of the underlying MDP.

Table 2: Impact of adversarial training

Distraction	w/o adv. training	Invariance Through Inference
Camera	292.46 ± 385.33	$\mathbf{674.20 \pm 259.02}$
Background	199.97 ± 283.93	$\mathbf{655.03 \pm 327.19}$
Color	239.05 ± 310.65	$\mathbf{682.51 \pm 342.29}$

Pre-Filling the Replay Buffer In our experiments, we pre-filled both the original latent buffer and the target observation buffer with data collected via random exploration. There is nothing particularly special about this choice, random exploration may be subbed out with any exploration strategy.

One interesting strategy is to pre-fill the buffer with an exploration strategy that relies on unsupervised pre-training. Given the recent success of these strategies on improving exploration, it seems likely this could further improve performance. This is left to future work.

6 Closing Remarks

We introduced Invariance through Inference, a new method for leveraging unsupervised data to improve the test-time adaptation of reinforcement learning systems. Empirical results demonstrate that as discrepancies between training and deployment environments become more intense, Invariance through Inference is more capable of adaptation than data augmentation techniques. The problem of test-time adaptation in visual reinforcement learning using unsupervised test-time trajectories is relatively new, but has thus far shown great promise.

Future work in this area might develop better techniques for the initial exploration phase. Many of the environments we considered were quite reversible, with a small intrinsic dimensionality. Expanding this class of methods to work on longer horizon multi-stage tasks is an intriguing possibility.

References

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 1597–1607, 2020a.

Xi Chen, Ali Ghadirzadeh, Mårten Björkman, and Patric Jensfelt. Adversarial feature training for generalizable robotic visuomotor control. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1142–1148, 2020b.

Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, pp. 1282–1289, 2019.

Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 1126–1135, 2017.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

Jean-Bastien Grill, Florian Strub, Florent Altché, C. Tallec, Pierre H. Richemond, Elena Buchatskaya, C. Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, B. Piot, K. Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. *ArXiv*, abs/2006.07733, 2020.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

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. *arXiv preprint arXiv:2007.04309*, 2020a.

Nicklas Hansen, Rishabh Jangir, Yu Sun, Guillem Alenyà, Pieter Abbeel, Alexei A Efros, Lerrel Pinto, and Xiaolong Wang. Self-supervised policy adaptation during deployment. *arXiv preprint arXiv:2007.04309*, 2020b.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Stabilizing deep q-learning with convnets and vision transformers under data augmentation. *arXiv preprint arXiv:2107.00644*, 2021a.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Stabilizing deep Q-learning with ConvNets and vision transformers under data augmentation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021b.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9726–9735, 2020.

Rein Houthooft, Richard Y Chen, Phillip Isola, Bradly C Stadie, Filip Wolski, Jonathan Ho, and Pieter Abbeel. Evolved policy gradients. *arXiv preprint arXiv:1802.04821*, 2018.

Kyle Hsu, Sergey Levine, and Chelsea Finn. Unsupervised learning via meta-learning. *arXiv preprint arXiv:1810.02334*, 2018.

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 arXiv:1611.05397*, 2016.

Aviral Kumar, Rishabh Agarwal, Aaron Courville, Tengyu Ma, George Tucker, and Sergey Levine. Value-based deep reinforcement learning requires explicit regularization. In *ICML 2021 Reinforcement Learning for Real Life Workshop*, 2021. URL https://drive.google.com/file/d/1Fg43H5oagQp-ksjpWBf_aDYEzAFMVJm6/edit.

Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. *arXiv preprint arXiv:2004.14990*, 2020.

Kimin Lee, Kibok Lee, Jinwoo Shin, and Honglak Lee. A simple randomization technique for generalization in deep reinforcement learning. *ArXiv*, abs/1910.05396, 2019.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2018.

Martin L Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, August 2014.

Roberta Raileanu, Max Goldstein, Denis Yarats, Ilya Kostrikov, and Rob Fergus. Automatic data augmentation for generalization in deep reinforcement learning. *arXiv preprint arXiv:2006.12862*, 2020.

Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 5331–5340, 2019.

Jonas Rothfuss, Dennis Lee, Ignasi Clavera, Tamim Asfour, and Pieter Abbeel. ProMP: Proximal meta-policy search. *arXiv preprint arXiv:1810.06784*, 2018.

Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with momentum predictive representations. *arXiv preprint arXiv:2007.05929*, 2020.

Laura Smith, Nikita Dhawan, Marvin Zhang, Pieter Abbeel, and Sergey Levine. AVID: Learning multi-stage tasks via pixel-level translation of human videos. *arXiv preprint arXiv:1912.04443*, 2019.

Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint arXiv:2004.04136*, 2020.

Bradly C Stadie, Pieter Abbeel, and Ilya Sutskever. Third-person imitation learning. *arXiv preprint arXiv:1703.01703*, 2017.

Austin Stone, Oscar Ramirez, Kurt Konolige, and Rico Jonschkowski. The distracting control suite—A challenging benchmark for reinforcement learning from pixels. *arXiv preprint arXiv:2101.02722*, 2021.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. DeepMind Control Suite. *arXiv preprint arXiv:1801.00690*, 2018.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. Tent: Fully test-time adaptation by entropy minimization. *arXiv preprint arXiv:2006.10726*, 2020.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021.

Amy Zhang, Nicolas Ballas, and Joelle Pineau. A dissection of overfitting and generalization in continuous reinforcement learning. *arXiv preprint arXiv:1806.07937*, 2018.