Stabilizing Off-Policy Deep Reinforcement Learning from Pixels

Edoardo Cetin [*1] **Philip J. Ball** [*2] **Steve Roberts** [2] **Oya Celiktutan** [1]

Abstract

Off-policy reinforcement learning (RL) from pixel observations is notoriously unstable. As a result, many successful algorithms must combine different domain-specific practices and auxiliary losses to learn meaningful behaviors in complex environments. In this work, we provide novel analysis demonstrating that these instabilities arise from performing temporal-difference learning with a convolutional encoder and low-magnitude rewards. We show that this new *visual deadly triad* causes unstable training and premature convergence to degenerate solutions, a phenomenon we name *catastrophic self-overfitting*. Based on our analysis, we propose A-LIX, a method providing adaptive regularization to the encoder's gradients that explicitly prevents the occurrence of *catastrophic self-overfitting* using a dual objective. By applying A-LIX, we significantly outperform the prior state-of-the-art on the DeepMind Control and Atari 100k benchmarks without any data augmentation or auxiliary losses.

1. Introduction

One of the core challenges in real world Reinforcement Learning (RL) is achieving stable training with sample-efficient algorithms (Dulac-Arnold et al., 2019). Combining these properties with the ability to reason from visual observations has great implications for the application of RL to the real world (Kalashnikov et al., 2018; Zhu et al., 2020). Recent works utilizing *temporal-difference* (TD-) learning have made great progress advancing sample-efficiency (Lillicrap et al., 2015; Fujimoto et al., 2018; Haarnoja et al., 2018a; Cetin & Celiktutan, 2021). However, stability has remained a key issue for these off-policy algorithms (Sutton, 1988; Duan et al., 2016; Van Hasselt et al., 2018; Buşoniu et al., 2018), making their general applicability limited as compared to their on-policy counterparts (Schulman et al., 2017; Cobbe et al., 2021). At the same time, using pixel observations has been another orthogonal source of instabilities, with several successful approaches relying on pre-training instead of end-to-end learning (Finn et al., 2015; Dwibedi et al., 2018). In fact, alternative optimization objectives, large amounts of simulation data, and symbolic observations have been common factors in most contemporary large-scale RL milestones (Silver et al., 2017; Vinyals et al., 2019; Berner et al., 2019).

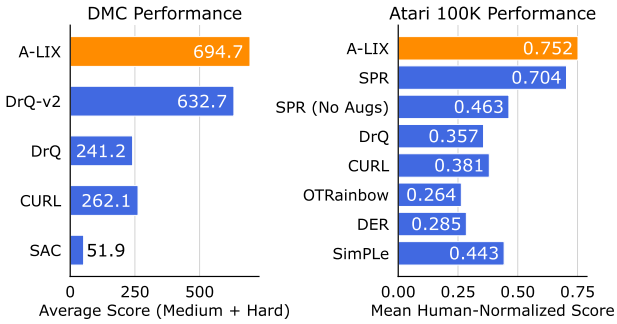


Figure 1. Performance of agents in DMC (**left**) and Atari 100k (**right**) benchmarks from 10 seeds. A-LIX outperforms previous methods without using image augmentations or auxiliary losses.

In this work, we provide novel insights behind why applying successful off-policy RL algorithms designed for proprioceptive tasks to pixel-based environments is generally underwhelming (Lee et al., 2019; Yarats et al., 2021). In particular, we provide evidence that three key elements strongly correlate with the occurrence of detrimental instabilities: *i) Exclusive* reliance on the *TD-loss*. *ii)* Unregularized end-to-end learning with a *convolutional encoder*. *iii) Low-magnitude sparse rewards*. Using this framework, we are able to motivate the effectiveness of auxiliary losses (Laskin et al., 2020b; Schwarzer et al., 2020; Yarats et al., 2021) and many domain-specific practices (Hessel et al., 2018b; Laskin et al., 2020a) by explaining how they address elements of this new *visual deadly triad*.

We focus our analysis on the popular DeepMind Control suite (DMC) (Tassa et al., 2018), where the introduction of random shift augmentations has played a key role in recent advances (Laskin et al., 2020a; Kostrikov et al., 2021;

[*]Equal contribution [1]Centre for Robotics Research, Department of Engineering, King's College London [2]Department of Engineering Science, University of Oxford. Correspondence to: Edoardo Cetin <edoardo.cetin@kcl.ac.uk>, Philip J. Ball <ball@robots.ox.ac.uk>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

Table 1. Practices from recent pixel-based TD-learning methods to mitigate elements of the visual deadly triad. †DrQ uses 10-step returns on Atari. *CURL uses 20-step returns on Atari.

Algorithm	Visual Deadly Triad Mitigation		
	TD-Loss	CNN Overfit	Low-Density Reward
DrQ/RAD	-	Shift/Jitter Augmentations	10-step returns†
DrQ-v2	-	Shift Augmentations	3-step returns
SAC-AE	VAE Loss	-	-
SPR	Model-Based Loss	Shift/Jitter Augmentations	10-step returns
DER	-	Non-Overlapping Strides	20-step returns
CURL	Contrastive Loss	Shift Augmentations	20-step returns*

Yarats et al., 2022). In this domain, we observe that the presence of the *visual deadly triad* results in the TD-loss gradients through the convolutional encoder's feature maps having high spatial frequencies. We find these gradients are *spatially inconsistent* and result in degenerate optimization landscapes when backpropagated to the encoder's parameters. Furthermore, repeatedly updating the convolutional encoder with these gradients consistently leads to early convergence to degenerate feature representations causing the critic to fit high-variance erroneous targets, a phenomenon we name *catastrophic self-overfitting*. As a way of identifying the direct implications of the visual deadly triad in the gradient signal, we propose a new measure called the *Normalized Discontinuity (ND)* score and show how its value precisely correlates with agent performance. Thus, we explain the effectiveness of shift augmentations by recognizing that they regularize the gradient signal by providing an *implicit* spatial smoothing effect.

Based on our analysis, we propose **A**daptive **L**ocal **SI**gnal Mi**X**ing (A-LIX) a novel method to prevent catastrophic self-overfitting with two key components: *i)* A new parameterized layer (LIX) that *explicitly* enforces smooth feature map gradients. *ii)* A dual objective that ensures learning stability by adapting the LIX parameters based on the estimated *ND* scores. We show that integrating A-LIX with existing off-policy algorithms achieves state-of-the-art performance in both DeepMind Control and Atari 100k benchmarks without requiring image augmentations or auxiliary losses and significantly fewer heuristics. We open-source our code to facilitate reproducibility and future extensions[1].

Our main contribution can be summarized as follows:

- We conjecture the existence of a ***visual deadly triad*** as a principal source of instability in reinforcement learning from pixel observations and provide clear empirical evidence validating our hypothesis.
- We show these instabilities affect the gradient signal causing ***catastrophic self-overfitting***, a phenomenon that can severely harm TD-learning. As a result, we design the ***normalized discontinuity score*** to explicitly anticipate its occurrence.
- We propose ***A-LIX***, a new method that adaptively regularizes convolutional features to prevent catastrophic self-overfitting, achieving state-of-the-art results on two popular pixel-based RL benchmarks.

[1]https://github.com/Aladoro/Stabilizing-Off-Policy-RL

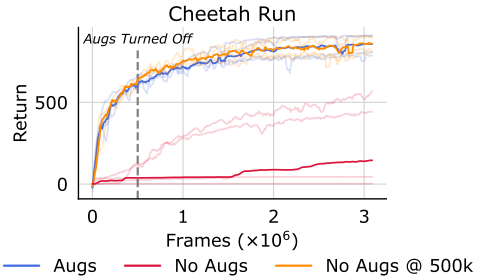


Figure 2. Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs. The vertical dashed line shows when augmentations are turned off.

2. Background

We consider problem settings described by Markov Decision Processes (MDPs) (Bellman, 1957), defined as the tuple $(S, A, P, p_0, r, \gamma)$. This comprises a state space S, an action space A, transitions dynamics given by P and p_0, and a reward function r. The RL objective is then for an agent to recover an optimal policy π^*, yielding a distribution of trajectories $p_\pi(\tau)$ that maximizes its expected sum of discounted future rewards, $\pi^* = \arg\max_\pi \mathbb{E}_{p_\pi(\tau)} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$. In off-policy RL, this objective is usually approached by learning a *critic* function to evaluate the effectiveness of the agent's behavior. A common choice for the critic is to parameterize the policy's Q-function $Q^\pi : S \times A \to \mathbb{R}$, that quantifies the agent's performance after performing a particular action: $Q^\pi(s,a) = \mathbb{E}_{p_\pi(\tau|s_0=s,a_0=a)} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$. Most off-policy algorithms entail storing trajectories in a buffer D, and learning parameterized Q-functions by iteratively minimizing a squared temporal difference (TD-) loss:

$$J_Q(\phi) = \mathbb{E}_{(s,a,s',r)\sim D}\left[(Q^\pi_\phi(s,a) - y)^2\right], \\ y = r + \gamma \mathbb{E}_{a\sim\pi(s')}\left[\hat{Q}^\pi_{\phi'}(s',a)\right]. \tag{1}$$

Here, the TD-targets y are computed from a 1-step bootstrap operation with a slowly-changing *target Q-function* $\hat{Q}^\pi_{\phi'}$. In continuous action spaces, we also learn a separate parameterized policy to exploit the information in the critic. This practically results in alternating TD-learning with maximizing the Q-function's expected return predictions, following the policy gradient theorem (Sutton et al., 2000).

3. Instabilities in TD-Learning from Pixels

Unlike proprioceptive observations, off-policy RL from pixel observations commonly requires additional domain-

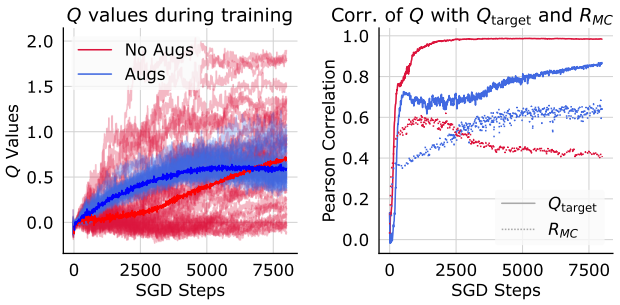


Figure 3. Evidence of overfitting when augmentations are not used. On the **left**, shaded lines are individual estimates, the solid line represents the median Q-value. On the **right**, the Q-values Pearson correlation with target values and Monte-Carlo returns (R_{MC}).

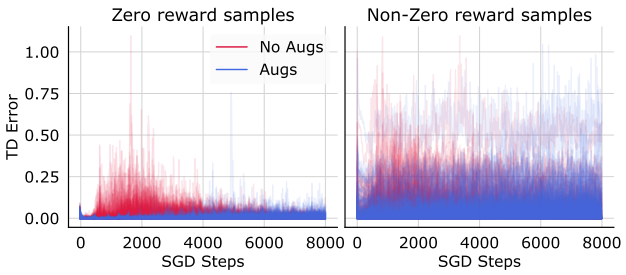


Figure 4. TD-loss of offline fixed transitions during training, separated based on having non-zero reward.

specific practices to ensure stability. In this section, we provide a novel analysis of this phenomenon by focusing on the DeepMind Control Suite (Tassa et al., 2018). In this benchmark, the introduction of random shift data augmentations has been a core component of recent advances in pixel-based off-policy RL (Laskin et al., 2020a; Yarats et al., 2022), allowing us to isolate and reproduce stable and unstable training regimes. Our analysis suggests the existence of specific elements that cause instabilities and strives to explain their implications on learning dynamics. We validate our findings via thorough empirical experimentation showing numerous results corroborating our hypotheses. Based on our discoveries, in Section 4 we provide a new interpretation of random shifts and propose a new improved method to isolate and counteract instabilities.

3.1. Why Do Augmentations Help?

The underlying mechanism behind the effectiveness of random shifts is not immediately clear. While this augmentation may appear to assist generalization by encoding an invariance (Shorten & Khoshgoftaar, 2019), we note that all the environments from DMC employ a camera that is fixed relative to the agent's position. Hence, robustness to shifts does not appear to introduce any useful inductive bias about the underlying tasks. Moreover, prior work successfully learned effective controllers without augmentations (Hafner et al., 2020; Yarats et al., 2021), suggesting that shift generalization might not be the primary benefit of this method. We analyze the effect of random shifts by training a DrQ-v2 agent (Yarats et al., 2022) on Cheetah Run but turning off augmentations after an initial 500,000 time-steps learning phase. As shown in Fig. 2, while training without any shift augmentation fails to make consistent progress, turning off augmentations after the initial learning phase actually appears to slightly improve the performance of DrQ-v2. This result is a clear indication that augmentations are not needed for asymptotic performance, and are most helpful to counteract instabilities present in the *earlier stages of learning*, which we now focus on analyzing (see App. F.1-F.2 for complementary experiments validating these claims).

3.2. Identifying a New Deadly Triad

To reduce confounding factors and to disentangle the origin of these instabilities, we design an offline RL experiment (Levine et al., 2020). This experiment isolates three distinct elements affecting off-policy RL: exploration, policy evaluation, and policy improvement. First, we gather a set of 15,000 transitions with pixel observations using a random policy in Cheetah Run. This allows us to ground exploration and analyze learning from fixed data resembling the early stages of online training (when augmentations appear most helpful). We then isolate policy evaluation by training both critic and encoder using SARSA (Rummery & Niranjan, 1994) until convergence on this fixed data. Finally, we run policy improvement, training an actor to maximize the expected discounted return as predicted by the converged critic (see App. B.1 for details). Interestingly, we find that *turning on* augmentations exclusively during exploration or policy improvement has no apparent effect on stability and final performance. Hence, we focus on the effects that augmentations have on TD-learning and analyze applying augmentations only during policy evaluation.

Table 2. Performance and training statistics of different agent types in the offline experiments from 15,000 random transitions.

Agent	Final TD-Loss	Final Policy Loss	Return
Augmented	0.021	-0.99	86.5 ± 11.3
Non-Augmented	0.002	-1.05	9.2 ± 12.1
Proprioceptive	0.012	-1.14	79.1 ± 7.7
Frozen CNN (random)	0.023	-0.95	43.6 ± 20.2
Frozen CNN (pre-trained)	0.012	-0.99	77.6 ± 18.5
Non-Augmented (norm r)	18.616	3.86	38.6 ± 16.5
Non-Augmented (10-step returns)	0.003	-1.24	36.5 ± 20.3

As shown in Table 2, applying augmentations during policy evaluation enables us to learn policies that achieve a return of 86.5, despite the best trajectory in the offline data achieving only 10.8. In contrast, without augmentations we consistently recover near 0 returns, resembling the failures observed in the online experiments. On the left of Fig. 3 we show the evolution of the predicted Q-values for both

agents on a fixed batch of offline data. In particular, when performing policy evaluation without augmentations, these predictions display extremely high variance across different state-action pairs. In Table 2 we further show that the non-augmented agent displays significantly lower loss, despite having higher average Q-values than the augmented agent (Schaul et al., 2021). We argue this is a clear indication of the occurrence of *overfitting*. We corroborate our claim by analyzing the evolution of the Pearson bi-variate correlation between the estimated Q-values and target Q-values on the right of Fig. 3. These results show that the non-augmented agent displays near-perfect correlation with its own target Q-values throughout training, indicating that it immediately *learns to fit its own noisy, randomly-initialized predictions*. We also record the correlation with the actual discounted Monte-Carlo returns, which represent the *true targets* the Q-values should ideally approximate during policy evaluation. For these results, we observe that the relationship between applying augmentations and the recorded correlation is reversed, with the non-augmented agent displaying significantly lower correlation. This dichotomy appears to indicate that fitting the noisy targets severely affects learning the *useful* training signal from the collected transitions regarding the experienced rewards. We confirm this phenomenon by splitting the data into non-zero and zero reward transitions, where the only learning signal propagated in the TD-loss is from the initially random target values. In Fig. 4 we illustrate that the non-augmented agents initially experience much higher TD-errors on zero reward transitions, confirming that they focus on fitting uninformative components of the TD-objective.

In Table 2 we provide the results of additional experiments that indicate that TD-learning is not the *only* cause for the observed instabilities. First, we confirm that the observed overfitting appears to be exclusive to performing *end-to-end* TD-learning with convolutional neural network (CNN) encoders. Concretely, we run the same offline experiment without *training an encoder* in three different settings. First, we consider performing policy evaluation directly from non-augmented *proprioceptive* observations with a fully-connected critic network. Moreover, we also consider *freezing* the encoder weights either to their initial *random values* or to *pre-trained* values from the augmented agent experiments. In all three cases, we attain largely superior performance, almost matching the augmented agent's performance for both the proprioceptive and pre-trained experiments. In addition, we also find that the observed overfitting phenomenon is diminished when simply increasing the magnitude of the reward signal in the TD-loss. We test this through two additional experiments which consider *normalizing* the collected rewards before policy evaluation and incorporating *large n-step returns* (Sutton, 1988). As reported, both modifications considerably improve the non-augmented agent's performance. However, we note that both practices introduce further unwanted variance in the optimization, failing to yield the same improvements as augmentations (see App. C.2).

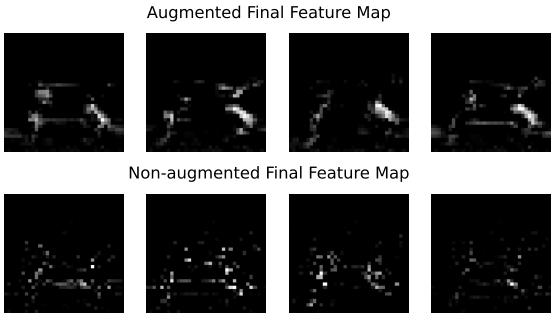


Figure 5. Feature maps in the final layer of both augmented (**top**) and non-augmented (**bottom**) agent encoders. Non-augmented agents manifest inconsistent, high-frequency feature maps.

Taken together, our results appear to strongly indicate that instabilities in off-policy RL from pixel observation come from three key conditions, which we refer to as the *visual deadly triad*: i) Exclusive reliance on the TD-loss; ii) Unregularized learning with an expressive convolutional encoder; iii) Initial low-magnitude sparse rewards. Further evidence arises when considering the ubiquity of particular practices employed in pixel-based off-policy RL. In particular, as summarized in Table 1, most popular prior algorithms feature design choices that appear to counteract at least two elements of this triad, either directly or implicitly. Furthermore, we show these instabilities result in the non-augmented critics focusing on learning their own noisy predictions, rather than the actual experienced returns. We observe this ultimately leads to convergence to erroneous and high-variance Q-value predictions, a phenomenon we name *catastrophic self-overfitting*.

3.3. Anticipating Catastrophic Self-Overfitting

We now attempt to unravel the links that connect the *visual deadly triad* with *catastrophic self-overfitting*. We start by observing that catastrophic self-overfitting comes with a significant reduction of the critic's sensitivity to changes in action inputs, implying that the erroneous high-variance Q-value predictions arise primarily due to changes in the observations (see App. F.3 for action-value surface plots). Hence, we focus on analyzing the feature representations of the pixel observations, computed by the convolutional encoder, $z \in \mathbb{R}^{C \times H \times W}$. In particular, we wish to quantify the *sensitivity* of feature representations to small perturbations in the input observations. To measure this, we evaluate the Jacobians of the encoder across a fixed batch of offline data for the augmented and non-augmented agents. We then calculate the Frobenius norm of each agent's Jacobians, giving us a measure of how quickly the encoder feature represen-

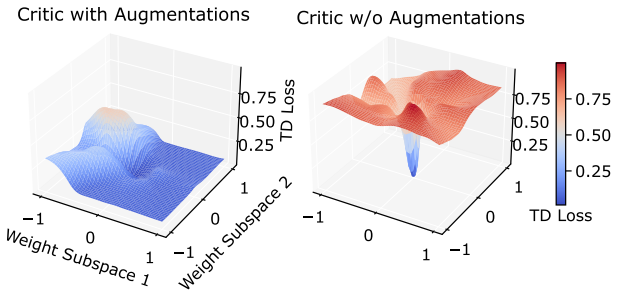


Figure 6. Critic loss surface plots of augmented (**left**) and non-augmented (**right**) agents after 5,000 steps of offline training.

tations are changing locally around an input (see App. B.2 for details). Our results show a stark difference, with the feature representations of the non-augmented agents being on average **2.85** times more sensitive. This suggests that overfitting is driven by the CNN encoder's *representations* learning *high-frequency* information about the input observations and, thus, breaking useful inductive biases about this class of models (Rahaman et al., 2019).

In App. E.1 we demonstrate that lower sensitivity to random noise, while desirable for optimization (Rosca et al., 2020), is actually a *byproduct* of a stable feature representations, and not its defining factor. Furthermore, observing the actual feature maps of different observations in Fig. 5, we see that augmentations make the encoder produce features that are *spatially consistent*, aligned with common understandings of how natural representations should appear (Alsallakh et al., 2021; Allen-Zhu & Li, 2021). In contrast, the non-augmented agents display high-frequency and *discontinuous* feature maps that do not reflect the spatial properties of their inputs. Hence, our evidence suggests that catastrophic self-overfitting specifically follows from the same learning process that produces highly-sensitive and discontinuous encoder feature maps. Therefore, we turn our focus to analyzing the *gradients* backpropagated to the encoder's features maps and observe one key property: the gradients of the output feature maps *consistently reflect the same spatial properties* of their resulting features. In particular, the *gradients of the feature maps* appear *spatially consistent* for the augmented agent, and *discontinuous* for the non-augmented agent. This optimization property reflects intuitive understandings of backpropagation since discontinuous gradients should push the encoder's weights to encode discontinuous representations. To provide further complementary evidence that discontinuous gradients are the direct cause of catastrophic self-overfitting, we analyze the normalized loss surfaces *when backpropagating these discontinuous gradients* to the encoder's parameters (following Li et al. (2018)). In Fig. 6, we see that gradient discontinuities in the non-augmented agent yield extreme peaks in its encoder's loss surface, clearly suggestive of overfitting (Keskar et al., 2017) [2].

To quantify the level of discontinuity in the features and their gradients, we propose a new metric that encodes the aggregated immediate spatial 'unevenness' of each feature location within its relative feature map. In particular, we define $D(z) \in \mathbb{R}^{C\times H\times W}$ as the expected squared local discontinuity of z in any spatial direction, i.e.:

$$D(z)_{ijc} \approx \mathbb{E}_{v\sim S^1}\left[\left(\frac{\partial z_{ijc}}{\partial v}\right)^2\right], \tag{2}$$

practically estimated via sampling. We then normalize each value in $D(z)$ by its squared input and average over all the feature positions. We name this metric the *normalized discontinuity (ND)* score:

$$ND(z) = \frac{1}{C\times H\times W}\sum_{c=1}^{C}\sum_{j=1}^{H}\sum_{i=1}^{W}\frac{D(z)_{ijc}}{z_{ijc}^2}. \tag{3}$$

Intuitively, this score reflects how locally discontinuous z is expected to be at any spatial location. In Fig. 7, we show how the ND score of ∇z evolves during training in the offline and an online setting for both augmented and non-augmented agents. We see that augmented agents experience considerably less discontinuous gradients through their features, and that recordings of lower ND scores also appear to be highly correlated with performance improvements. We additionally show an accumulated ND score, using an exponential moving average of ∇z in each spatial position to calculate this metric. Interestingly, we observe that the ND score over accumulated gradients is almost identical to the instantaneous ND score, showing that similar gradient discontinuities are propagated *persistently* through training in each position of the feature map. This property confirms that the discontinuities are not smoothed by the stochastic sampling of different consecutive training batches, in which case we would expect to observe lower accumulated ND scores. Thus, it suggests that self-overfitting emerges in the non-augmented agents due to repeated gradient steps towards *persistent* feature map discontinuities.

4. Counteracting Gradient Discontinuities

4.1. Gradient Smoothing and Random Shifts

As analyzed in Section 3, catastrophic self-overfitting occurs when the gradients in the convolution layers are locally discontinuous. As a result, we argue that the efficacy of random shifts arises from their downstream effect on feature gradient computation, which counteracts these discontinuities during backpropagation. In particular, while random shifts do not act directly on the latent representation or their

[2] Instead, the loss surface with respect to the fully-connected weights is smoother (App. F.5).

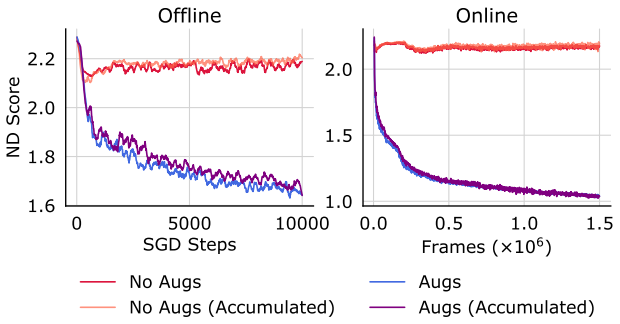


Figure 7. Instantaneous (red and blue) and accumulated (orange and purple) ND scores for the features gradients from offline (**left**) and online (**right**) training in Cheetah Run.

respective gradients, they do affect *how* the latent representations are computed. This has an impact on how *persistent* discontinuous components of the gradient are backpropagated to the encoder's parameters during learning. From the approximate shift invariance of convolutional layers, we can view a convolutional encoder as computing each of the feature vectors $[z_{1ij}, ..., z_{Cij}]^T$ with the same parameterized function, V_ϕ, that takes as input a subset of each observation $O \in \mathbb{R}^{C' \times H' \times W'}$. This subset corresponds to a local neighborhood around some reference input coordinates i', j'. Thus, the only factor differentiating features in the same feature map (e.g., z_{cij} and z_{ckl}) is some implicit function $f(i, j) = i', j'$ translating each of the output features coordinate into the relative reference input coordinate, i.e. $z_{cij} = V_\phi(O, i', j')_c$ (determined by kernel sizes, strides...). Therefore, random shifts are approximately equivalent to further translating each reference coordinate by adding some uniform random variables δ'_x, δ'_y:

$$z_{cij} \approx V_\phi(O, i' + \delta'_x, j' + \delta'_y)_c,$$
$$\text{where} \quad \delta'_x, \delta'_y \sim U[-s', s'], \quad f(i, j) = i', j'.$$

Due to the employed strides from the convolutional architectures used in DrQ-v2 (Yarats et al., 2022), the difference in reference coordinates of adjacent features in a feature map is less than the maximum allowable shift employed in the augmentations, i.e., $(i + 1)' - i', (j + 1)' - j' < s'$ (where s' is the maximum allowable shift). Consequently, shift augmentations effectively turn the deterministic computation graph of each feature z_{cij} into a random variable, whose sample space comprises the computation graphs of all nearby features within its feature map. Hence, applying different random shifts to samples in a minibatch makes the gradient of each feature ∇z_{cij} backpropagate to a random computation graph, sampled from a set that extends the set of non-augmented computation graphs for all features in a local neighborhood of coordinates i, j. Therefore, aggregating the parameter gradients produced with different δ'_x, δ'_y, provides a *smoothing effect* on how each discontinuous component of ∇z affects learning, and prevents persistent discontinuities from accumulating. Hence, random shifts break the second condition of the visual deadly triad, by providing effective *implicit* regularization of the convolutional encoder's learning process. At the same time, this minimally disrupts the information content of the resultant features, since discarded observation borders almost exclusively comprise background textures that are irrelevant for performing the task. This interpretation of random shifts aligns with the analysis in Section 3, showing that implicitly smoothing over the backpropagated gradient maps consistently prevents catastrophic self-overfitting.

4.2. Local Signal Mixing

We extrapolate our hypotheses about catastrophic self-overfitting and random shifts by proposing a technique that aims to enforce gradient smoothing regularization *explicitly*. We propose **L**ocal S**I**gnal Mi**X**ing, or **LIX**, a new layer specifically designed to prevent catastrophic self-overfitting in convolutional reinforcement learning architectures. LIX acts on the features produced by the convolutional encoder, $z \in \mathbb{R}^{C \times H \times W}$, by randomly mixing each component z_{cij} with its neighbors belonging to the same feature map. Hence, LIX outputs a new latent representation with the same dimensionality $\hat{z} \in \mathbb{R}^{C \times H \times W}$, whose computation graph minimally disrupts the information content of each feature z_{cij} while smoothing discontinuous components of the gradient signal during backpropagation.

LIX is a regularization layer that acts as a simple random smoothing operation, reducing the expected magnitude of gradient discontinuities by preventing higher frequency signals to persist. In the forward pass, LIX produces a new latent representation where for each of the C feature maps, $\hat{z}_{cij}$ is computed as a randomly weighted average of its spatial neighbors around coordinates i, j. We further *parameterize* this stochastic operation using some maximum range radius S, and consequently sample two uniform *continuous* random variables $\delta_x, \delta_y \sim U[-S, S]$, representing shifts in the x and y coordinates respectively. Correspondingly, we define $\tilde{i} = i + \delta_x$ and $\tilde{j} = j + \delta_y$ and perform the weighted averaging as a bilinear interpolation with weights determined by the random shifts:

$$\begin{aligned}\hat{z}_{cij} =& z_{c\lfloor\tilde{i}\rfloor\lfloor\tilde{j}\rfloor}(\lceil\tilde{i}\rceil - \tilde{i})(\lceil\tilde{j}\rceil - \tilde{j}) + z_{c\lfloor\tilde{i}\rfloor\lceil\tilde{j}\rceil}(\lceil\tilde{i}\rceil - \tilde{i})(\tilde{j} - \lfloor\tilde{j}\rfloor) \\ &+ z_{c\lceil i\rceil\lfloor\tilde{j}\rfloor}(\tilde{i} - \lfloor\tilde{i}\rfloor)(\lceil\tilde{j}\rceil - \tilde{j}) + z_{c\lceil i\rceil\lceil\tilde{j}\rceil}(\tilde{i} - \lfloor\tilde{i}\rfloor)(\tilde{j} - \lfloor\tilde{j}\rfloor).\end{aligned}$$

Since nearby features in a convolutional feature map are computed with very similar receptive fields, the mixing effect of LIX should have a trivial effect on the information the encoder can convey in its latent representations. In addition, LIX should have a direct regularization effect on the gradients by acting on the feature maps themselves. In particular, since LIX computes each output feature from a weighted average of its neighbors, back-

propagation will split each gradient $\nabla \hat{z}_{cij}$, to a random local combination of features within the same feature map, $\{\nabla z_{c\lfloor \tilde{i} \rfloor \lfloor \tilde{j} \rfloor}, \nabla z_{c\lfloor \tilde{i} \rfloor \lceil \tilde{j} \rceil}, \nabla z_{c\lceil \tilde{i} \rceil \lfloor \tilde{j} \rfloor}, \nabla z_{c\lceil \tilde{i} \rceil \lceil \tilde{j} \rceil}\}$. Thus, LIX should mostly preserve the consistent component of ∇z, while randomly smoothing its discontinuous component.

There are multiple key differences between the regularization from LIX and random shifts. LIX provides a local smoothing effect over the gradients explicitly and exactly, without having to deal with the implications of padding and strided convolutions breaking shift-invariance assumptions. Moreover, LIX smooths the gradient signal not only across different inputs but also within each feature map. In addition, by applying its operation solely at the feature level, the encoder can still learn to entirely circumvent LIX's smoothing effect on the information encoded in the latent representations, given enough capacity. This means that LIX does not forcibly preclude any input information from affecting the computation. Consequently, LIX also does not have to enforce learning invariances which might not necessarily reflect useful inductive biases about the distribution of observations. In contrast, random shifts need to exploit the particular uninformativeness of the observations borders to avoid disrupting the features' information content.

4.3. A-LIX

LIX introduces a single key parameter: the range radius S used for sampling δ_x and δ_y. Intuitively, this value should reflect how much we expect gradients to be locally consistent for a given architecture and task. Therefore, we argue that the value of S should ideally decrease throughout training as the useful learning signal from the TD-loss becomes stronger. This is consistent with the results illustrated in Figure 2, showing that turning off random shift augmentations after the TD-targets become informative can improve learning. Hence, we propose an adaptive strategy to learn S throughout training. Utilizing the *normalized discontinuity (ND)* score in Section 3.3, we set up a dual optimization objective to ensure a minimum value of local smoothness in the representation gradients, $\overline{ND}$. However, computing the ND score of the gradient signal involves a ratio between potentially very small values. As a result, estimation of these values from a batch of gradient samples can lead to outliers having an extreme impact on this average measure, translating into large erroneous updates of S. To overcome this, we propose using a slightly modified version of the ND score with increased robustness to outliers (see App. C.1 for further details):

$$\widetilde{ND}(\nabla \hat{z}) = \sum_{c=1}^{C} \sum_{j=1}^{H} \sum_{i=1}^{W} \log \left(1 + \frac{D(\nabla \hat{z})_{cij}}{\nabla \hat{z}_{ijc}^2} \right). \quad (4)$$

In practice, we set up a dual optimization objective similar to the automatic temperature adjustment from Haarnoja et al. (2018b). This entails alternating the optimization of the TD-learning objectives described in Section 2 with minimizing a dual objective loss:

$$\arg\min_{S} -S \times \mathbb{E}_{\hat{z}} \left[\widetilde{ND}(\nabla \hat{z}) - \overline{ND} \right], \quad (5)$$

approximating dual gradient descent (Boyd et al., 2004). Hence, we call this new layer **A**daptive **LIX** (A-LIX). In Fig. 8 we show that A-LIX effectively anneals S as the agent escapes its unstable regimes, in line with our intuition.

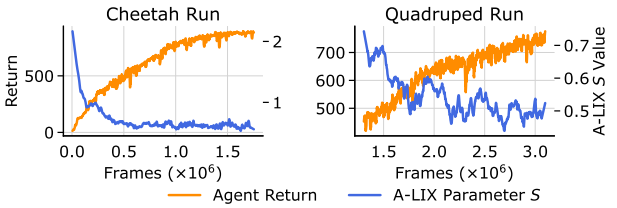


Figure 8. A-LIX's S parameter evolution during training in Cheetah Run (**left**) and Quadruped Run (**right**). As the critic targets become more informative, S falls, improving data efficiency and asymptotic performance.

5. Performance Evaluation

We evaluate the effectiveness of A-LIX in pixel-based reinforcement learning tasks in two popular and distinct domains featuring a diverse set of continuous and discrete control problems. We integrate A-LIX with existing popular algorithms and compare against current state-of-the-art model-free baselines. We provide further details of our integration and full hyperparameters in App. D. We also extend this section by providing more granular evaluation metrics in App. A. Furthermore, we provide ablation studies analyzing different components of A-LIX in App. E.

5.1. DeepMind Control Evaluation

We first evaluate the effectiveness of A-LIX for pixel-based RL on continuous control tasks from the DeepMind Control Suite (DMC) (Tassa et al., 2018). Concretely, we integrate A-LIX with the training procedure and network architecture from DrQ-v2 (Yarats et al., 2022), but *without using image augmentations*. To show the generality of our method we *do not modify* any of the environment-specific hyperparameters from DrQ-v2 and simply add our A-LIX layer after each encoder nonlinearity. For simplicity, we optimize a shared S for all the A-LIX layers with the dual objective in Eq. 5. Hence, this introduces a single additional parameter and negligible computational overhead. We compare A-LIX to DrQ-v2, which represents the current state-of-the-art on this benchmark. We also compare against three further baselines: the original DrQ (Kostrikov et al., 2021), which foregoes n-step returns and includes an entropy bonus; CURL (Laskin et al., 2020b), which includes an auxiliary contrastive ob-

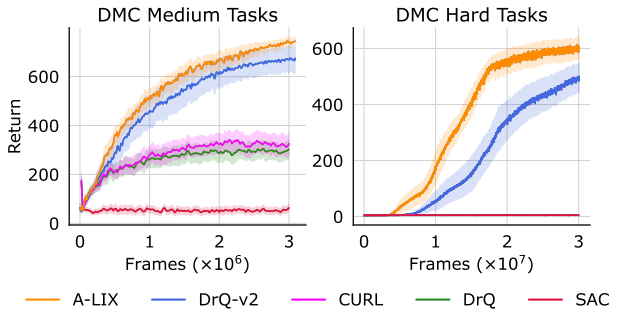


Figure 9. Average performance in 10 seeds for DMC Medium (**left**) and Hard tasks (**right**). Shaded regions represent ± 1 SE.

jective; an extension of SAC (Haarnoja et al., 2018b) with the encoder from Yarats et al. (2021). These last three baselines have been performant on a prior DMC benchmark that considers fewer tasks with high action repeats, as described by Hafner et al. (2019). Instead, we evaluate on the more challenging 'Medium' and 'Hard' benchmarks from Yarats et al. (2022), comprising 15 tasks with low action repeats.

Results. We summarize the results in Figure 9, showing the mean performance curves for both medium and hard benchmark tasks. We provide further details and the full list of results across all 15 environments in App. A.1. Overall, A-LIX surpasses all prior methods with clear margins, both in terms of efficiency and final performanc. This is particularly notable in the more complex 'Hard' tasks. As highlighted in prior work (Cetin & Celiktutan, 2021), DrQ-v2 appears to yield inconsistent results on some of the harder exploration tasks with sparse rewards. This likely indicates that the gradient regularization induced by random shifts (described in Section 4.1) is unable to consistently prevent catastrophic self-overfitting in scenarios where the initial learning signal from TD-learning is particularly low. Finally, DrQ, CURL, and SAC fail to make consistent meaningful progress on this harder benchmark. This performance gap corroborates the third component of the visual deadly triad, showing how lower magnitude rewards due to harder exploration and lower action-repeats further destabilize TD-learning based algorithms, and explains the gains seen in DrQ-v2 when incorporating n-step returns. We believe these results emphasize the challenge of overcoming the visual deadly triad in continuous control problems and the particular effectiveness of A-LIX to counteract its direct implications.

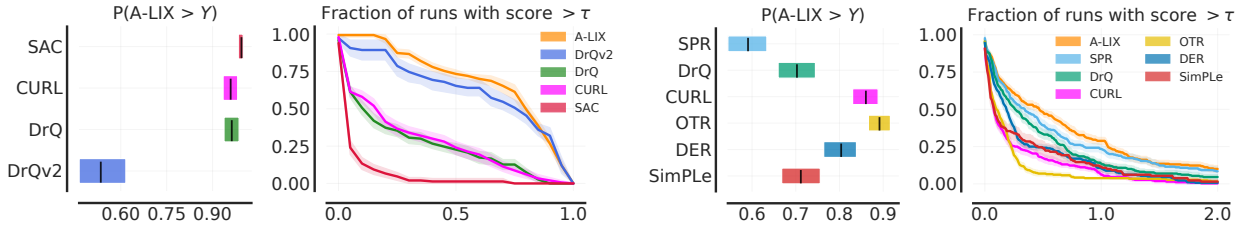
5.2. Atari 100k Evaluation

We perform a second set of experiments in an entirely different setting, discrete control. We make use of the popular Atari Learning Environment (ALE) (Bellemare et al., 2013) and consider the 100k evaluation benchmark from Kaiser et al. (2020). In particular, this benchmark comprises evaluating performance for 26 tasks after only two hours of play-time (100k interaction steps), following the evaluation protocol in Machado et al. (2018). We integrate A-LIX with Data-Efficient Rainbow (DER) (van Hasselt et al., 2019), a simple extension to Rainbow (Hessel et al., 2018a) with improved data-efficiency. We would like to note that our integration has key differences to DER, designed to highlight the generality of our method in tackling the visual deadly triad. In particular, we reduce the n-step returns to 3 (from 20), and we maintain the *same* encoder architecture as in DrQ-v2. To speak to the latter point, this means we do not require the highly regularized encoders with large convolutional filters and strides, used ubiquitously in off-policy learning for Atari environments. Instead, to stabilize learning we simply apply our A-LIX layer after the final encoder nonlinearity. We compare against three algorithms that, like A-LIX, do not employ data-augmentation: Data-Efficient Rainbow (DER); Overtrained Rainbow (OTRainbow) (Kielak, 2019); and Simulated Policy Learning (SimPLe) (Kaiser et al., 2020) (model-based). Moreover, we also compare with additional state-of-the-art off-policy baselines that make use of data augmentations: the aforementioned CURL and DrQ; and Self-Predictive Representations (SPR) (Schwarzer et al., 2020), the current state-of-the-art TD-learning based algorithm on this benchmark. SPR combines data augmentation with numerous additional algorithmic design choices, such as an auxiliary self-supervised loss for learning a latent dynamics model.

Table 3. Results summary for the Atari 100k benchmark. The reported performance of A-LIX is from 10 seeds.

Metrics	SimPLe	DER	OTRainbow	CURL	DrQ	SPR	**A-LIX**
Norm. Mean	0.443	0.285	0.264	0.381	0.357	0.704	**0.753**
Norm. Median	0.144	0.161	0.204	0.175	0.268	**0.415**	0.411
# SOTA	7	1	1	1	1	4	**11**
# Super	2	2	1	2	2	**7**	**7**
Average Rank	3.92	5.00	5.21	3.92	4.85	2.88	**2.21**

Results. We summarize the results in Table 3, showing the mean and median human-normalized scores together with the number of environments where each algorithm either achieves state-of-the-art or super-human performance. We include the full per-environment results in App. A.2. Remarkably, A-LIX obtains a substantially higher human-normalized mean performance than all other considered algorithms. While the recorded normalized median performance is slightly inferior to SPR, we argue that such difference is not particularly significant since this metric depends on the performance obtained in just two environments. Moreover, A-LIX achieves super-human performance in 7 games (the same as SPR), and state-of-the-art performance in 11 games, considerably more than all other algorithms. These results corroborate how tuned architectures, data augmentation, and auxiliary losses used on ALE mostly serve the purpose of counteracting the direct implications of the visual deadly triad and show that A-LIX enables us to learn



(a) DeepMind Control: Medium and Hard Tasks (b) Atari 100k

Figure 10. Probability of improvement and *performance profiles* obtained from the recorded results in DMC (**left**) and Atari 100k (**right**). A-LIX displays statistically significantly improvements and stochastically dominates most prior algorithms.

powerful models without relying on these design choices.

5.3. Statistical Significance

To validate the significance of our improvements, we statistically analyze our results using the *Rliable* tools and practices from Agarwal et al. (2021). We summarize some of our key findings in Fig. 10, showing the *probability of improvements* of A-LIX over prior methods (computed with the Mann-Whitney U statistic (Mann & Whitney, 1947)) and the relative normalized *performance profiles* (Dolan & Moré, 2002). The ranges correspond to 95% *stratified bootstrap confidence intervals* (Efron, 1992). In both DMC and Atari benchmarks, we find that our improvements are statistically significant (lower confidence intervals >0.5) and observe 'stochastic dominance' of our algorithm against almost all considered baselines (Dror et al., 2019). We provide further results and details of the employed statistical analysis in App. A.1 and App. A.3 respectively.

6. Related Work

Previous works have characterized several optimization issues related to performing RL via TD-learning (Baird & Moore, 1998; Baird, 1999). In this work, we instead focus on the *empirical analysis* of modern TD-learning algorithms, specific to the pixel-based RL setting. We also observe links with recent work studying *observational overfitting* (Song et al., 2020). Our work differs by focusing on memorization effects particular to the combination of CNNs and TD-learning. There are also connections with existing feature-level augmentation work, such as Dropout (Srivastava et al., 2014) and DropBlock (Ghiasi et al., 2018). In particular, the latter also applies structured transformations directly to the feature maps and introduces a heuristic to adjust this transformation over training, validating our findings on the utility of adaptivity. Outside RL, there is a rich body of work on implicit regularization and memorization in CNNs (Keskar et al., 2017; Neyshabur et al., 2017; Arpit et al., 2017; Liu et al., 2020; Maennel et al., 2020). Rahaman et al. (2019) show that higher frequency data manifolds cause CNNs to learn higher spectral frequency terms, aligning with our analysis of higher frequency representations. Chatterjee (2020) show generalization arises when similar examples induce similar gradients during learning (i.e., coherence). Their work supports our findings since inconsistent feature gradients are a manifestation of non-coherence, explaining their poor generalization. Finally, our dual objective falls under automatic tuning methods in RL (AutoRL) (Parker-Holder et al., 2022). These approaches have been applied very successfully to manage non-stationary trade-offs, such as exploration and exploitation (Ball et al., 2020) and optimism (Moskovitz et al., 2021; Cetin & Celiktutan, 2021). Finally, we note links with recent work concerning implicit regularization in TD-learning (Kumar et al., 2021). However, while Kumar et al. (2021) observe an implicit 'underfitting' phenomenon in later training stages, we analyze an opposed 'overfitting' phenomenon occurring during the first training steps, which we find to be specific to learning from visual inputs.

7. Conclusion

In this work, we provide a novel analysis demonstrating that instabilities in pixel-based off-policy RL come specifically from performing TD-learning with a convolutional encoder in the presence of a sparse reward signal. We show this *visual deadly triad* affects the encoder's gradients, causing the critic to *catastrophically self-overfit* to its own noisy predictions. Therefore, we propose **A**daptive **L**ocal **SI**gnal Mi**X**ing (A-LIX), a powerful regularization layer to explicitly counteract this phenomenon. Applying A-LIX enables us to outperform prior state-of-the-art algorithms on popular benchmarks without relying on image augmentations, auxiliary losses, or other notable design choices.

Acknowledgments

Edoardo Cetin and Oya Celiktutan would like to acknowledge the support from the Engineering and Physical Sciences Research Council [EP/R513064/1] and LISI Project [EP/V010875/1]. Philip J. Ball would like to thank the Willowgrove Foundation for support and funding. Furthermore, support from Toyota Motor Corporation contributed towards funding the utilized computational resources.

References

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A., and Bellemare, M. G. Deep reinforcement learning at the edge of the statistical precipice, 2021.

Allen-Zhu, Z. and Li, Y. Feature purification: How adversarial training performs robust deep learning, 2021.

Alsallakh, B., Kokhlikyan, N., Miglani, V., Yuan, J., and Reblitz-Richardson, O. Mind the pad – {cnn}s can develop blind spots. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=m1CD7tPubNy.

Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., and Lacoste-Julien, S. A closer look at memorization in deep networks. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 233–242. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/arpit17a.html.

Baird, L. Reinforcement learning through gradient descent. Technical report, Carnegie-Mellon University, Department of Computer Science, 1999.

Baird, L. and Moore, A. Gradient descent for general reinforcement learning. *Advances in neural information processing systems*, 11, 1998.

Ball, P., Parker-Holder, J., Pacchiano, A., Choromanski, K., and Roberts, S. Ready policy one: World building through active learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML*. 2020.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Bellman, R. A markovian decision process. *Indiana Univ. Math. J.*, 6:679–684, 1957. ISSN 0022-2518.

Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Boyd, S., Boyd, S. P., and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Brandfonbrener, D., Whitney, W. F., Ranganath, R., and Bruna, J. Offline RL without off-policy evaluation. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=LU687itn08w.

Buşoniu, L., de Bruin, T., Tolić, D., Kober, J., and Palunko, I. Reinforcement learning for control: Performance, stability, and deep approximators. *Annual Reviews in Control*, 46:8–28, 2018.

Cetin, E. and Celiktutan, O. Learning pessimism for robust and efficient off-policy reinforcement learning. *arXiv preprint arXiv:2110.03375*, 2021.

Chatterjee, S. Coherent gradients: An approach to understanding generalization in gradient descent-based optimization. *arXiv preprint arXiv:2002.10657*, 2020.

Cobbe, K. W., Hilton, J., Klimov, O., and Schulman, J. Phasic policy gradient. In *International Conference on Machine Learning*, pp. 2020–2027. PMLR, 2021.

Dolan, E. D. and Moré, J. J. Benchmarking optimization software with performance profiles. *Mathematical programming*, 91(2):201–213, 2002.

Dror, R., Shlomov, S., and Reichart, R. Deep dominance-how to properly compare deep neural models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2773–2785, 2019.

Duan, Y., Chen, X., Houthooft, R., Schulman, J., and Abbeel, P. Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning*, pp. 1329–1338. PMLR, 2016.

Dulac-Arnold, G., Mankowitz, D., and Hester, T. Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*, 2019.

Dwibedi, D., Tompson, J., Lynch, C., and Sermanet, P. Learning actionable representations from visual observations. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1577–1584. IEEE, 2018.

Efron, B. Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics*, pp. 569–593. Springer, 1992.

Finn, C., Tan, X. Y., Duan, Y., Darrell, T., Levine, S., and Abbeel, P. Learning visual feature spaces for robotic manipulation with deep spatial autoencoders. *arXiv preprint arXiv:1509.06113*, 25, 2015.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4{rl}: Datasets for deep data-driven reinforcement learning, 2021.

Fujimoto, S., van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *ICML*, pp. 1582–1591, 2018. URL http://proceedings.mlr.press/v80/fujimoto18a.html.

Ghiasi, G., Lin, T.-Y., and Le, Q. V. Dropblock: A regularization method for convolutional networks. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/7edcfb2d8f6a659ef4cd1e6c9b6d7079-Paper.pdf.

Gogianu, F., Berariu, T., Rosca, M. C., Clopath, C., Busoniu, L., and Pascanu, R. Spectral normalisation for deep reinforcement learning: An optimisation perspective. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 3734–3744. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/gogianu21a.html.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 10–15 Jul 2018a. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b.

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

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Hernandez-Garcia, J. F. and Sutton, R. S. Understanding multi-step deep reinforcement learning: A systematic study of the dqn target, 2019.

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018a.

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-second AAAI conference on artificial intelligence*, 2018b.

Kaiser, L., Babaeizadeh, M., Milos, P., Osiński, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., Mohiuddin, A., Sepassi, R., Tucker, G., and Michalewski, H. Model based reinforcement learning for Atari. In *International Conference on Learning Representations*, 2020.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

Kearns, M. J. and Singh, S. P. Bias-variance error bounds for temporal difference updates. In *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*, COLT '00, pp. 142–147, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 155860703X.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. On large-batch training for deep learning: Generalization gap and sharp minima. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=H1oyRlYgg.

Kielak, K. P. Do recent advancements in model-based deep reinforcement learning really improve data efficiency? 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kostrikov, I., Yarats, D., and Fergus, R. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*. 2021.

Kumar, A., Agarwal, R., Ghosh, D., and Levine, S. Implicit under-parameterization inhibits data-efficient deep reinforcement learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=O9bnihsFfXU.

Laskin, M., Lee, K., Stooke, A., Pinto, L., Abbeel, P., and Srinivas, A. Reinforcement learning with augmented data. In *Advances in Neural Information Processing Systems 33*. 2020a.

Laskin, M., Srinivas, A., and Abbeel, P. CURL: Contrastive unsupervised representations for reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning*, 2020b.

Lee, A. X., Nagabandi, A., Abbeel, P., and Levine, S. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *arXiv preprint arXiv:1907.00953*, 2019.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020.

Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. Visualizing the loss landscape of neural nets. In *Neural Information Processing Systems*, 2018.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Liu, S., Papailiopoulos, D., and Achlioptas, D. Bad global minima exist and sgd can reach them. *Advances in Neural Information Processing Systems*, 33, 2020.

Machado, M. C., Bellemare, M. G., Talvitie, E., Veness, J., Hausknecht, M., and Bowling, M. Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. *Journal of Artificial Intelligence Research*, 61:523–562, 2018.

Maennel, H., Alabdulmohsin, I. M., Tolstikhin, I. O., Baldock, R., Bousquet, O., Gelly, S., and Keysers, D. What do neural networks learn when trained with random labels? In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 19693–19704. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/e4191d610537305de1d294adb121b513-Paper.pdf.

Mann, H. B. and Whitney, D. R. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics*, 18(1):50 – 60, 1947. doi: 10.1214/aoms/1177730491. URL https://doi.org/10.1214/aoms/1177730491.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1QRgziT-.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Moskovitz, T., Parker-Holder, J., Pacchiano, A., Arbel, M., and Jordan, M. Tactical optimism and pessimism for deep reinforcement learning. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=a4WgjcLeZIn.

Neyshabur, B., Bhojanapalli, S., McAllester, D., and Srebro, N. Exploring generalization in deep learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 5949–5958, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Parker-Holder, J., Rajan, R., Song, X., Biedenkapp, A., Miao, Y., Eimer, T., Zhang, B., Nguyen, V., Calandra, R., Faust, A., Hutter, F., and Lindauer, M. Automated reinforcement learning (autorl): A survey and open problems, 2022.

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., and Courville, A. On the spectral bias of neural networks. In *International Conference on Machine Learning*, pp. 5301–5310. PMLR, 2019.

Rosca, M., Weber, T., Gretton, A., and Mohamed, S. A case for new neural networks smoothness constraints. In *"I Can't Believe It's Not Better!" NeurIPS 2020 workshop*, 2020. URL https://openreview.net/forum?id=_b-uT9wCI-7.

Rummery, G. A. and Niranjan, M. On-line Q-learning using connectionist systems. Technical Report TR 166, Cambridge University Engineering Department, Cambridge, England, 1994.

Schaul, T., Ostrovski, G., Kemaev, I., and Borsa, D. Return-based scaling: Yet another normalisation trick for deep rl, 2021.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

Schwarzer, M., Anand, A., Goel, R., Hjelm, R. D., Courville, A., and Bachman, P. Data-efficient reinforcement learning with self-predictive representations. *arXiv preprint arXiv:2007.05929*, 2020.

Shorten, C. and Khoshgoftaar, T. M. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. Deterministic policy gradient algorithms. 2014.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017.

Song, X., Jiang, Y., Tu, S., Du, Y., and Neyshabur, B. Observational overfitting in reinforcement learning. In *International Conference on Learning Representations*, 2020.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html.

Student. The probable error of a mean. *Biometrika*, 6 (1):1–25, 1908. ISSN 00063444. URL http://www.jstor.org/stable/2331554.

Sutton, R. Learning to predict by the method of temporal differences. *Machine Learning*, 3:9–44, 08 1988. doi: 10.1007/BF00115009.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pp. 1057–1063, 2000.

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

Van Hasselt, H., Doron, Y., Strub, F., Hessel, M., Sonnerat, N., and Modayil, J. Deep reinforcement learning and the deadly triad. *arXiv preprint arXiv:1812.02648*, 2018.

van Hasselt, H. P., Hessel, M., and Aslanides, J. When to use parametric models in reinforcement learning? *Advances in Neural Information Processing Systems*, 32:14322–14333, 2019.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Wilcoxon, F. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6):80–83, 1945. ISSN 00994987. URL http://www.jstor.org/stable/3001968.

Yarats, D., Zhang, A., Kostrikov, I., Amos, B., Pineau, J., and Fergus, R. Improving sample efficiency in model-free reinforcement learning from images. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12): 10674–10681, May 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/17276.

Yarats, D., Fergus, R., Lazaric, A., and Pinto, L. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=_SJ-_yyes8.

Zhu, H., Yu, J., Gupta, A., Shah, D., Hartikainen, K., Singh, A., Kumar, V., and Levine, S. The ingredients of real-world robotic reinforcement learning. *arXiv preprint arXiv:2004.12570*, 2020.

A. Detailed Results

A.1. DMC Medium and Hard Tasks

In Table 4, we show the performance in each of the evaluated 15 DMC environments by reporting the mean and standard deviations over the cumulative returns obtained midway and at the end of training for the medium and hard benchmark tasks, respectively. A-LIX attains state-of-the-art performance in the majority of the tasks at *both* reported checkpoints, while still closely matching DrQ-v2's performance on the remaining tasks. On the other hand, DrQ-v2 struggles to consistently solve some of the harder exploration tasks such as Cartpole Swingup Sparse and Humanoid Run, as shown by the high standard deviations. Interestingly, unlike in the simpler DMC benchmark from Hafner et al. (2019) with higher action repeat, CURL appears have a slight edge over DrQ. In particular, the self-supervised signal from CURL appears to aid precisely in the sparse reward environments where DrQ-v2 struggles. Hence, this appears to suggest that including an additional self-supervised signal to the TD-loss, lessens the hindering effects of a lower-magnitude reward signal. We interpret this result as additional evidence showing how addressing any individual component of the deadly triad helps counteracting the catastrophic self-overfitting phenomenon.

We also test the significance of our results by performing a Wilcoxon signed-rank test (Wilcoxon, 1945) between A-LIX and DrQ-v2. We perform a paired rank test across both seeds and tasks, allowing us to obtain an p-value that takes into account both population size and relative performance gains across all tasks. The choice of Wilcoxon signed-rank test also does not presume normality in the distributions of performance which we believe is a more appropriate assumption than for instance a paired t-test (Student, 1908), despite a potential loss of statistical power. To ensure correct population pairing, A-LIX and DrQ-v2 seeds were identical, resulting in the same initially collected data and network initialization. Performing this test over all 15 tasks and 5 seeds, we achieve a p-value of $\mathbf{0.0057}$ at 50% total frames (1.5M and 15M for Medium and Hard respectively) and $\mathbf{0.0053}$ at 100% total frames (3.0M and 30M for Medium and Hard Respectively), much lower than the typical rejection criteria of $p > 0.05$. We therefore believe this shows clear evidence that our results in DMC are *strongly statistically significant*.

Table 4. Full results for the DeepMind Control Suite benchmark. Each displayed return is averaged over 10 random seeds and from 10 evaluation runs collected at each experience checkpoint.

	1.5M frames					3.0M frames				
Medium tasks	SAC	CURL	DrQ	DrQv2	**A-LIX (Ours)**	SAC	CURL	DrQ	DrQv2	**A-LIX (Ours)**
Acrobot Swingup	8±9	6±5	24±27	256±47	**270±99**	12±11	6±5	28±25	**442±64**	402±100
Cartpole Swingup Sparse	118±233	479±329	318±389	485±396	**718±250**	185±295	499±349	316±389	505±412	**742±250**
Cheetah Run	9±8	507±114	788±59	792±29	**806±78**	7±8	590±95	835±45	**873±60**	864±78
Finger Turn Easy	190±137	297±150	199±132	**854±73**	546±101	200±155	309±176	216±158	**934±54**	901±109
Finger Turn Hard	79±73	174±106	100±63	491±182	**587±109**	100±78	146±95	86±70	902±77	**906±101**
Hopper Hop	0±0	184±127	268±91	198±102	**287±48**	0±0	224±135	285±96	240±123	**372±48**
Quadruped Run	68±72	164±91	129±97	419±204	**528±107**	63±45	175±104	130±59	523±271	**759±107**
Quadruped Walk	75±65	134±53	144±149	591±256	**776±37**	48±32	168±49	142±67	920±36	**900±37**
Reach Duplo	1±1	8±10	8±12	**220±7**	212±3	2±3	7±10	9±9	**228±2**	221±3
Reacher Easy	52±64	707±142	600±201	**971±4**	887±19	115±98	667±182	612±181	940±50	**966±19**
Reacher Hard	3±2	463±196	320±233	**727±172**	720±83	10±23	678±350	397±273	**935±49**	855±83
Walker Run	26±4	379±234	474±148	571±276	**691±10**	25±3	447±224	547±143	616±297	**756±10**
Average score	52.28	291.73	281.03	547.96	**585.67**	63.80	326.45	300.27	671.40	**720.30**
	15.0M frames					30.0M frames				
Hard tasks	SAC	CURL	DrQ	DrQv2	**A-LIX (Ours)**	SAC	CURL	DrQ	DrQv2	**A-LIX (Ours)**
Humanoid Walk	7±3	5±3	3±2	243±162	**476±79**	4±3	4±3	5±3	675±86	**754±79**
Humanoid Stand	5±3	6±3	4±3	167±159	**519±94**	6±3	6±2	6±2	588±63	**781±94**
Humanoid Run	5±3	6±2	5±3	22±30	**122±59**	3±3	4±3	4±2	170±122	**242±59**
Average score	5.64	5.74	4.02	144.16	**372.78**	4.30	4.89	4.90	477.74	**592.48**

We now compare our results using the *Rliable* framework introduced in Agarwal et al. (2021) (see App. A.3 for a detailed explanation about the metrics introduced).

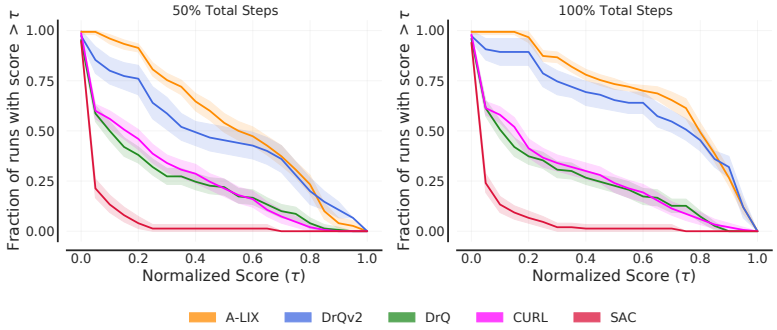
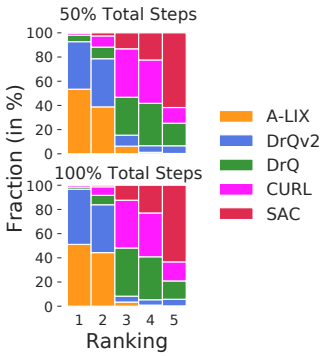
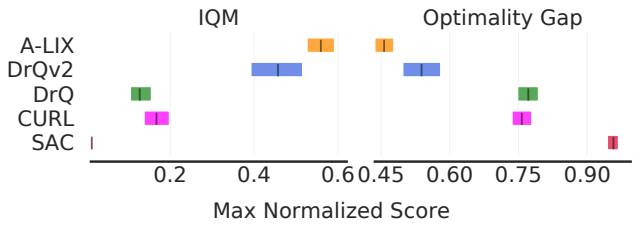


Figure 11. Performance profiles at 50% (**left**) and 100% (**right**) of the total steps in Medium and Hard DMC Tasks.

We plot performance profiles in Fig. 11 at both 50% and 100% the total training steps in DMC, which aim to represent sample efficiency and asymptotic performance respectively. We see that in almost all cases, A-LIX improves upon DrQ-v2.

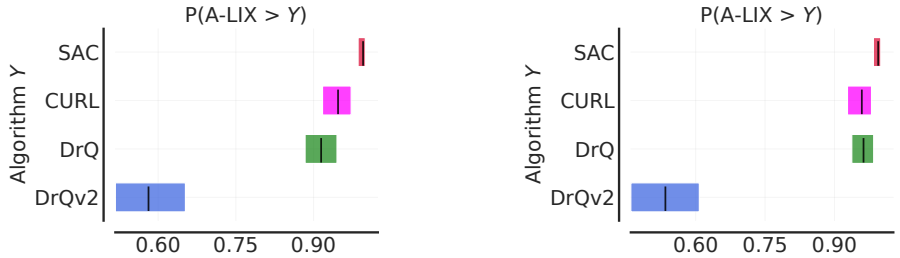


(a) Overall ranking statistics at 50% (**top**) and 100% (**bottom**) of the total steps in Medium and Hard DMC Tasks.



(b) Aggregate IQM (**left**) and Optimality Gap (**right**) metrics at 50% of the total steps in Medium and Hard DMC Tasks.

We plot ranking statistics in Fig. 11 at both 50% and 100% the total training steps in DMC. We see that A-LIX clearly appears most in the 1st ranked column, and rarely appears in lower ranked (i.e., > 3), suggesting strong performance across all environments in DMC Medium and Hard. We also provide a further aggregated statistics plot in Fig. 12b (this time at 50% the total steps), which shows A-LIX is particularly sample-efficient and consistent (i.e., low error bars) across all environments.



(a) 50% total steps. (b) 100% total steps.

Figure 13. Probability of Improvement statitistics at both 50% (**left**) and 100% (**right**) of the total timesteps in Medium and Hard DMC Tasks.

In Fig. 13 we observe that A-LIX likely improves over prior work, and note that whilst the improvement probability over DrQ-v2 may seem slightly low at ∼60%, we note that this value is in line with statistics in prior works that achieve significant gains (as seen in Agarwal et al. (2021)), and furthermore it does not take into account absolute performance values, and instead only compares relative values, which explains why the gains of A-LIX appear larger when evaluated under IQM and OG. Furthermore, the lower CI for 50% total steps does not fall below 0.5, which means improvements are indeed statistically significant.

A.2. Atari 100k

In Table 5, we show the final average performance for all the evaluated algorithms in each of the twenty-six tasks in the Atari 100k benchmark. A-LIX outperforms SPR, the previous state-of-the-art off-policy algorithm on this benchmark, on 16 out of 26 tasks. Moreover, it attains comparatively similar performance on most of the remaining tasks despite using no augmentation, auxiliary losses, or model-based elements.

Table 5. Full results for the Atari 100k benchmark, following the evaluation protocol from Machado et al. (2018). We report the results collected from 10 random seeds.

Tasks	Random	Human	SimPLe	DER	OTRainbow	CURL	DrQ	SPR	**A-LIX (Ours)**
Alien	227.80	7127.70	616.9	739.9	824.7	558.2	771.2	801.5	**902**
Amidar	5.80	1719.50	88	**188.6**	82.8	142.1	102.8	176.3	174.27
Assault	222.40	742.00	527.2	431.2	351.9	600.6	452.4	571	**660.53**
Asterix	210.00	8503.30	**1128.3**	470.8	628.5	734.5	603.5	977.8	809.5
Bank Heist	14.20	753.10	34.2	51	182.1	131.6	168.9	380.9	**639.4**
Battle Zone	2360.00	37187.50	5184.4	10124.6	4060.6	14870	12954	**16651**	14470
Boxing	0.10	12.10	9.1	0.2	2.5	1.2	6	**35.8**	21.5
Breakout	1.70	30.50	16.4	1.9	9.8	4.9	16.1	17.1	**23.52**
Chopper Command	811.00	7387.80	**1246.9**	861.8	1033.3	1058.5	780.3	974.8	747
Crazy Climber	10780.50	35829.40	**62583.6**	16185.3	21327.8	12146.5	20516.5	42923.6	53166
Demon Attack	152.10	1971.00	208.1	508	711.8	817.6	**1113.4**	545.2	888.15
Freeway	0.00	29.60	20.3	27.9	25	26.7	9.8	24.4	**31.04**
Frostbite	65.20	4334.70	254.7	866.8	231.6	1181.3	331.1	1821.5	**1845.7**
Gopher	257.60	2412.50	771	349.5	**778**	669.3	636.3	715.2	500.6
Hero	1027.00	30826.40	2656.6	6857	6458.8	6279.3	3736.3	7019.2	**7185.85**
Jamesbond	29.00	302.80	125.3	301.6	112.3	471	236	**365.4**	341.5
Kangaroo	52.00	3035.00	323.1	779.3	605.4	872.5	940.6	3276.4	**6507**
Krull	1598.00	2665.50	4539.9	2851.5	3277.9	4229.6	4018.1	3688.9	**4884.04**
Kung Fu Master	258.50	22736.30	**17257.2**	14346.1	5722.2	14307.8	9111	13192.7	16316
Ms Pacman	307.30	6951.60	**1480**	1204.1	941.9	1465.5	960.5	1313.2	1258.4
Pong	-20.70	14.60	**12.8**	-19.3	1.3	-16.5	-8.5	-5.9	6.03
Private Eye	24.90	69571.30	58.3	97.8	100	**218.4**	-13.6	124	100
Qbert	163.90	13455.00	1288.8	1152.9	509.3	1042.4	854.4	669.1	**2974**
Road Runner	11.50	7845.00	5640.6	9600	2696.7	5661	8895.1	14220.5	**17471**
Seaquest	68.40	42054.70	**683.3**	354.1	286.9	384.5	301.2	583.1	654.6
Up N Down	533.40	11693.20	3350.3	2877.4	2847.6	2955.2	3180.8	**28138.5**	5011.7
Human Norm. Mean	0.000	1.000	0.443	0.285	0.264	0.381	0.357	0.704	**0.753**
Human Norm. Median	0.000	1.000	0.144	0.161	0.204	0.175	0.268	**0.415**	0.411
# SOTA	N/A	N/A	7	1	1	1	1	4	**11**
# Super	N/A	N/A	2	2	1	2	2	**7**	**7**
Average Rank	N/A	N/A	3.92	5.00	5.21	3.92	4.85	2.88	**2.21**

We now present additional evaluations under the *Rliable* framework, continuing on from the analysis in Fig. 10b.

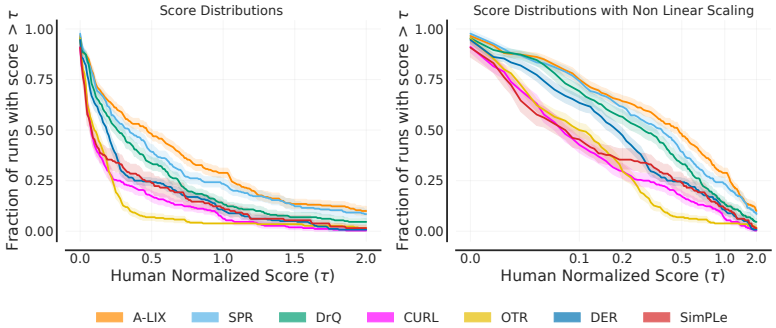
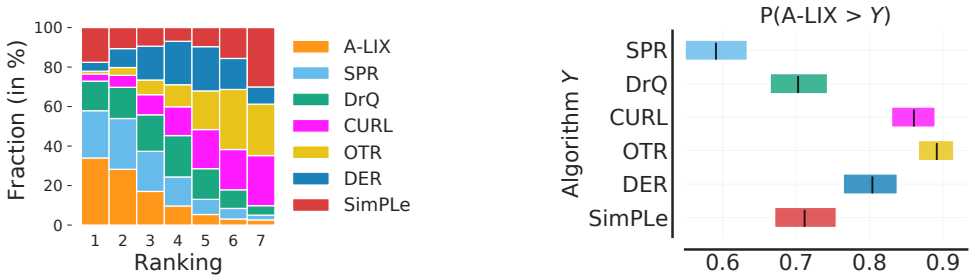


Figure 14. Performance profiles with linear (**left**) and logarithmic (**right**) scaling in Atari 100k.

In Fig. 14 A-LIX performs noticeably better than previous work, and performs at least as well as SPR over all settings of

normalized scores.



(a) Ranking statistics.

(b) Probability of improvement statistics.

Figure 15. Bootstrapped ranking statistics (**left**) and probability of improvement plots (**right**) on Atari 100k.

In Fig. 15a A-LIX constitutes the majority of the algorithms ranked in 1st, and shows far fewer instances of being ranked in lower positions (i.e., > 4). In Fig. 15b we observe A-LIX likely improves upon prior work. Similar to Fig. 13, while the $\sim$60% improvement value over SPR may seem low, this is justified due to shortcomings in this metric, such as not taking into account actual performance values, and instead relative improvements. Furthermore, the lower CI does not fall below 0.5, which means improvements due to A-LIX are statistically significant.

A.3. Rliable: A Primer

In addition to providing traditional methods of evaluation (e.g., performance tables, significance testing), we use robust metrics and evaluation strategies introduced in *Rliable* (Agarwal et al., 2021). Rliable advocates for computing aggregate performance statistics not just across many seeds, but also across the many tasks within a benchmark suite.

We give details on how these metrics achieve reliable performance evaluation in RL, denoting number of seeds as N and number of tasks as M. We follow Agarwal et al. (2021) as closely as possible; please refer to their paper for further details.

A.3.1. Seed and Task aggregation

In order to aggregate performances across different tasks in the same benchmark suite, we must first normalize each benchmark to the same range. In Atari, this is usually done by normalizing scores with respect to those achieved by humans, and in DMC this is done with respect to the maximum achievable score (i.e., $1,000$). We refer to this normalized score as τ.

A.3.2. IQM and OG

Interquartile Mean (IQM) takes the middle 50% of the runs across seeds and benchmarks (i.e., $[NM/2]$) and then calculates its mean score, improving outlier robustness whilst maintaining statistical efficiency. Optimality Gap (OG) calculates the proportion of performances (NM) that fail to meet a minimum threshold γ, with the assumption that improvements beyond γ are not important. In both cases, stratified bootstrap sampling is used to calculate confidence intervals (CIs).

A.3.3. Performance Profiles

Performance profiles are a form of empirical CDF, but with stratified bootstrap sampling to produce confidence bands that account for the underlying variability of the score. We can also establish 'stochastic dominance' by observing whether one method's performance profile is consistently above another's for all normalized performance values τ.

A.3.4. Ranking

Ranking shows the proportion of times a given algorithm ranks in a given position across *all* tasks, with distributions produced using stratified bootstrap sampling having $200,000$ repetitions.

A.3.5. Probability of Improvement

Probability of improvement is calculated by calculating the Mann-Whitney U-statistic (Mann & Whitney, 1947) across all M tasks. The distribution is then plotted as a boxplot, and if the lower CI > 0.5, the improvement is *statistically significant*.

B. Experiments Description

B.1. Offline Experiments

We follow the original training hyperparameters of DrQ-v2, and run policy evaluation and policy improvement until we saw convergence in the TD-loss, which would occur at similar points in all agents (i.e., between 10-20k and 5-10k steps of SGD in policy evaluation and policy iteration respectively). For the proprioceptive experiments, we keep everything consistent, except the input to the critic and actor MLP layers are now the proprioceptive states from the DMC simulator, not the latent representation z from the encoder. That is to say we do not modify the MLP architectures nor their learning rates in the interests of a fair comparison. Furthermore, for any given seed of the offline experiment, we also instantiate all networks in the agents identically and train on the same random offline data, with minibatches presented in the same order.

We also note that a similar algorithm is described in Brandfonbrener et al. (2021), but in the context of minimizing extrapolation errors.

Now we present some additional analysis to provide further context to our offline experiments. First, we see that the proprioceptive statistics mirror those of the augmented agent, further illustrating the crucial role of CNN regularization for successful TD-learning from pixels:

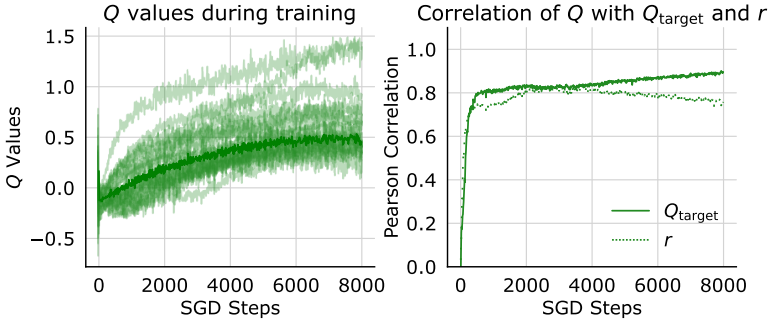


Figure 16. Q values and Pearson Correlation of the offline Proprioceptive agent on an offline fixed batch.

Secondly, we observe that the exact same self-overfit also manifests in the online setting by plotting the Pearson correlation values over the initial stages of training in 5 seeds, confirming that phenomena of our offline analysis applies to the online RL problem:

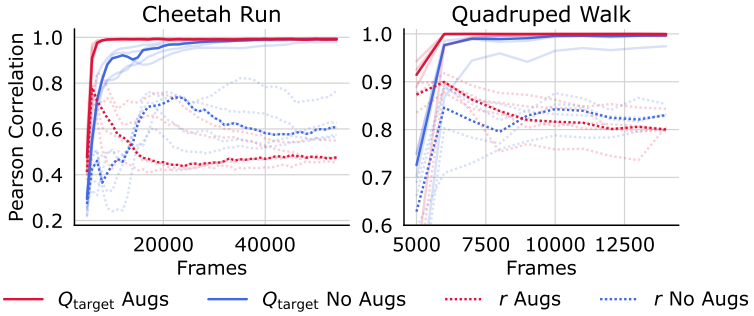


Figure 17. Pearson Correlation of augmented and non-augmented online agents in Cheetah Run and Quadruped Walk across 5 seeds. Shaded lines represent individual runs, and solid lines represent the median. We see that augmented agents do not immediately overfit to their target networks, and become correlated only after useful signal is learned.

B.2. Jacobian Analysis

In order to measure local sensitivity, we linearize the encoder around its input using a Taylor series expansion. Consider an N-dimensional input $\mathbf{x} \in \mathbb{R}^N$ and perturbation $\boldsymbol{\epsilon} \in \mathbb{R}^N$, an M-dimensional output $\mathbf{y} \in \mathbb{R}^M$, and a function $\mathbf{F} : \mathbb{R}^N \to \mathbb{R}^M$. Now, performing a Taylor series expansion around $\tilde{\mathbf{x}}$:

$$\mathbf{F}(\tilde{\mathbf{x}} + \boldsymbol{\epsilon}) = \mathbf{F}(\tilde{\mathbf{x}}) + \boldsymbol{\epsilon}\mathbf{F}(\mathbf{x})\nabla^T|_{\mathbf{x}=\tilde{\mathbf{x}}} + \frac{\boldsymbol{\epsilon}^2}{2}\nabla\mathbf{F}(\mathbf{x})\nabla^T|_{\mathbf{x}=\tilde{\mathbf{x}}} + \dots \tag{6}$$

$$\approx \mathbf{F}(\tilde{\mathbf{x}}) + \mathbf{J}(\tilde{\mathbf{x}})\boldsymbol{\epsilon} \tag{7}$$

$$= \tilde{\mathbf{y}} \tag{8}$$

where we make the approximation in the second line by dropping the second order/Hessian and higher terms under the assumption the perturbation vector $\boldsymbol{\epsilon}$ is small. This allows us to write $\mathbf{F}$ in the form of a local linear system: $\mathbf{y} = \mathbf{F}(\mathbf{x}) + \mathbf{J}(\mathbf{x})\boldsymbol{\epsilon}$. It is straightforward to see that if the entries of the Jacobian matrix $\mathbf{J}$ are larger, then small perturbations $\boldsymbol{\epsilon}$ will cause larger changes in the output $\mathbf{y}$. To measure the magnitude of the Jacobian entries, we take the Frobenius norm:

$$||\mathbf{J}(\mathbf{x})||_F = \sum_n \sum_m \left(\frac{\partial \mathbf{F}_m(\mathbf{x})}{\partial \mathbf{x}_n} \right)^2 \tag{9}$$

where $\mathbf{x}_n$ is the 'n'th entry of $\mathbf{x}$ and $\mathbf{F}_m$ is the 'm'th entry of the codomain of $\mathbf{F}$. The calculation of the Jacobian is trivial through the use of an automatic differentiation framework.

In our analysis we calculate the Jacobians of both agents on of a fixed batch of 128 frame stacked images taken from the offline training dataset, and compare the corresponding ratios of their Frobenius norms, and take this average ratio over the batch across 4 seeds.

C. Additional Analysis

C.1. Adaptive ND Dual Objective Optimization

The alternative ND score with increased outlier robustness, $\widetilde{ND}$, proposed in Section 4.3 is inspired by recordings of signal-to-noise ratio measurements. In particular, by passing the individual normalized $D(z)$ terms through a $\log(1+x)$ smoothing function we downweight the effect that large individual outliers might have on this aggregated metric. We would like to remark that since we set up the optimization of S with a dual objective, changes in the actual target value relating to some appropriate smoothness constraint are mostly irrelevant when considering the optimization's dynamics. Therefore, we argue that tuning S with the actual ND should not considerably diverge from tuning S based on a re-scaled appropriate target for $\widetilde{ND}$.

We provide further plots comparing agent performance and respective adaptive parameter S during training:

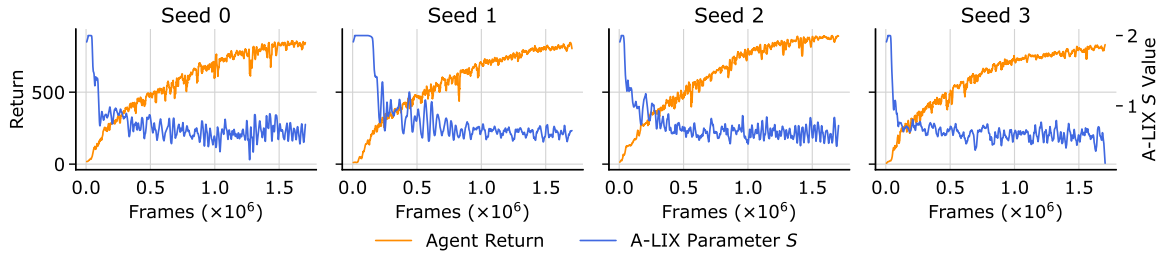


Figure 18. Performance of agents across 4 different seeds of the Cheetah Run environment and their adaptive scalar parameter S. We observe that initially, S is high until agents learn useful behaviors, whereupon it drops to maintain ND due to presence of useful signal in the feature gradients.

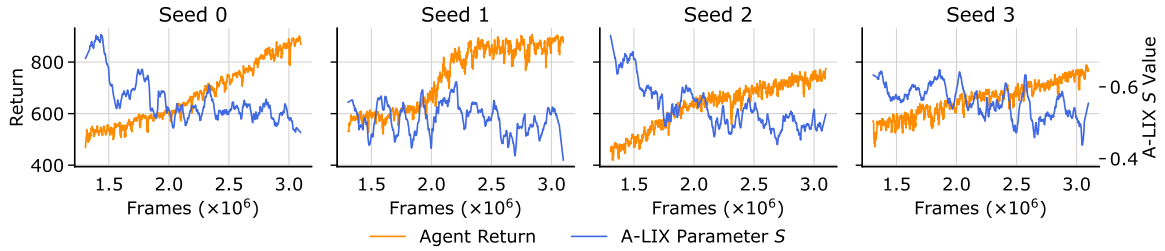


Figure 19. Performance of agents across 4 different seeds of the Quadruped Run environment and their adaptive scalar parameter S. We observe that as meaningful behaviors are learned in agents towards the end of training, S falls accordingly, whereupon it drops to maintain ND due to presence of useful signal in the feature gradients.

We see the same effect in these two contrasting environments; in Cheetah Run, where learning is more stable due to more predictable initializations and fewer degrees of freedom, we see the A-LIX parameter S drop almost immediately as the TD-targets quickly become more accurate. In the less stable Quadruped Run, we also notice this annealing effect, however this occurs later on in training, when the agent can consistently recover from poor initializations.

C.2. N-Step Returns

Large n-step rewards have become an important part of many algorithms that use TD-learning from visual observations. As motivated in Section 3, large n-step rewards can help towards mitigating self-overfitting by densifying the reward and downweighting the contribution of the inaccurate target critic, especially early in training; indeed as shown in (Yarats et al., 2022), using 1-step learning has a significant negative impact on performance. However, it is known that there is a bias-variance trade-off with multi-step approaches (Kearns & Singh, 2000), and furthermore, almost all approaches using this method do not apply off-policy bias correction when sampling from a replay buffer. While we motivate the use of n-step returns as a way to mitigate self-overfitting through incurring fewer 0 reward tuples (especially common in sparse reward environments early in training), we believe there is evidence to show that this introduces bias when n is sufficiently large, despite prior work suggesting this is not the case (Hernandez-Garcia & Sutton, 2019).

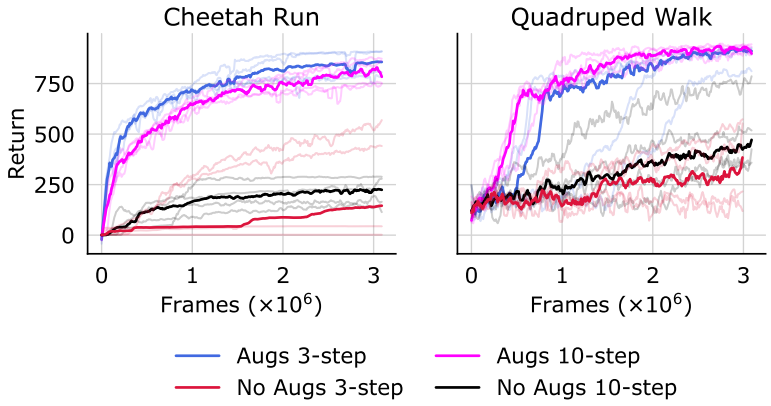


Figure 20. Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs.

We show in Fig. 20 that 10-step (as is commonly done in algorithms used to solve Atari) returns can mitigate failure seeds as predicted under the visual deadly triad framework (indeed in Cheetah Run there are no seeds that completely flat-line when 10-step returns are used). However, we also see evidence that applying 10-step returns can have negative impacts on convergence and asymptotic performance in Cheetah Run when the deadly triad is sufficiently managed, such as using augmentations; in Quadruped Run we see moderate benefit initially, but note that asymptotically the 10-step and 3-step agents converge to the same performance. We also provide further evidence in App. E.3, where applying 10-step returns to an A-LIX agent generally has a laregely negative impact on performance. Finally, we note that trying 20-step returns, as is done in some algorithms that solve Atari (Laskin et al., 2020b), caused significant performance reductions in DMC. In conclusion, this provides evidence that we should consider using lower values of 'n' in multi-step returns, and achieve this through addressing other elements of the deadly triad.

D. Implementation Details

In Tables 6 and 7 we provide the full list of hyperparameters used in our implementations for DMC and Atari 100k, respectively. We show significant differences from standard practices in **bold**. In particular, A-LIX uses *the same encoder architecture and n-step returns* for both benchmarks, highlighting its lower reliance to environment-specific heuristics. Moreover, unlike prior state-of-the-art algorithms it *does not employ any data augmentation or auxiliary loss function*. These factors show the effectiveness of our adaptive method in counteracting instabilities from the visual deadly triad without any additional help, highlighting its applicability.

Table 6. Full hyperparameters list used for the DeepMind Control A-LIX experiments. **Bolded** values represent significant differences from canonical implementations.

DDPG-integration hyperparameters (following (Yarats et al., 2022))	
Replay data buffer size	1000000 (100000 for *Quadruped Run*)
Batch size	256 (512 for *Walker Run*)
Minimum data before training	4000
Random exploration steps	2000
Optimizer	*Adam* (Kingma & Ba, 2014)
Policy/critic learning rate	medium: 0.0001 hard: 0.00008
Policy/critic β_1	0.9
Critic UTD ratio	0.5
Policy UTD ratio	0.5
Discount γ	0.99
Polyak coefficient ρ	0.99
N-step returns	3 (1 for *Walker Run*)
Hidden dimensionality	1024
Feature dimensionality	medium: 50 hard: 100
Nonlinearity	ReLU
Exploration stddev. clip	0.3
Exploration stddev. schedule	medium: linear: $1 \rightarrow 0.1$ in 500000 steps hard: linear: $1 \rightarrow 0.1$ in 2000000 steps
Augmentations	**OFF**
A-LIX-specific hyperparameters	
Initial maximum sampling shift S	1.0
Normalized discontinuity targets $\overline{ND}$	0.635
Maximum sampling shift learning rate	0.003
Maximum sampling shift β_1	0.5

Table 7. Full hyperparameters list used for the Atari 100k A-LIX experiments. **Bolded** values represent significant differences from canonical implementations.

DER-integration hyperparameters	
Gray-scaling	True
Down-sampling	84×84
Frames stacked	4
Action repetitions	4
Reward clipping	$[-1, 1]$
Max episode frames	108000
Replay data buffer size	100000
Replay period every	1
Batch size	32
Minimum data before training	1600
Random exploration steps	1600
Optimizer	*Adam* (Kingma & Ba, 2014)
Critic learning rate	0.0001
Critic β_1	0.9
Critic ϵ	0.000015
Max gradient norm	10
Critic UTD ratio	2
Discount γ	0.99
Target update period	1
***N*-step returns**	$\mathbf{3}$
Feature maps	$\mathbf{32}, \mathbf{32}, \mathbf{32}$
Filter sizes	$\mathbf{3 \times 3}, \mathbf{3 \times 3}, \mathbf{3 \times 3}$
Strides	$\mathbf{2}, \mathbf{1}, \mathbf{1}$
Hidden dimensionality	256
Feature dimensionality	50
Nonlinearity	ReLU
Exploration noisy nets parameter	0.1
Augmentations	**OFF**
A-LIX-specific hyperparameters	
Initial maximum sampling shift S	1.0
Normalized discontinuity targets $\overline{ND}$	0.75
Maximum sampling shift learning rate	0.0001
Maximum sampling shift β_1	0.5

E. Additional Ablations

E.1. Smoothness Regularization through Spectral Normalization

To distinguish between general smoothness contraints in convolutional features, and the smoothness that arises as a result spatial consistency, we apply spectral normalization (Miyato et al., 2018) to the final convolutional layer in the encoder to represent the former class of constraints. Spectral normalization operates on the parameters of a network and constrains its outputs to be 1-Lipschitz and has shown benefits in prior work (Gogianu et al., 2021), but does not explicitly enforce a *spatial regularization* in the features. We train agents without augmentations using spectral normalization.

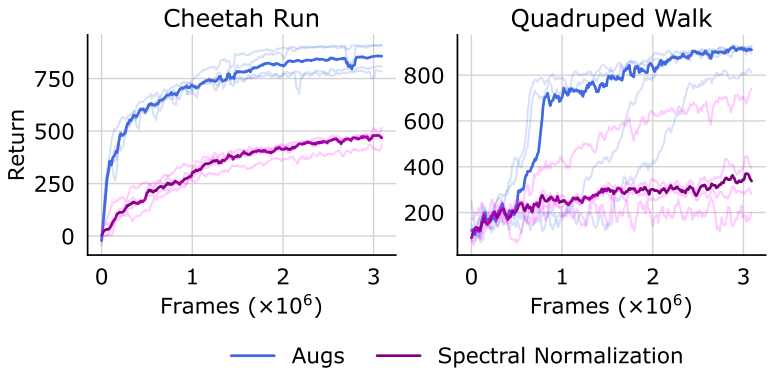


Figure 21. Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs.

We see that whilst there is clear improvement above the original non-augmented agents in some cases, the performance is still lower than agents that use spatial consistency regularization, such as random shift augmentations.

E.2. Is Gradient Smoothing All We Need?

Following the argument in Section 4.1, we can view augmentations as a gradient smoothing regularizer. This naturally leads us to ask the following: can we replace the stochastic shifting mechanism with a fixed smoothing mechanism? To test this, we instead apply a Gaussian smoothing kernel to the feature gradients in the CNN, and utilize our ND score to vary the width of the kernel adaptively through training; we call this method A-Gauss (**A**daptive **Gauss**ian Feature Gradient Kernel).

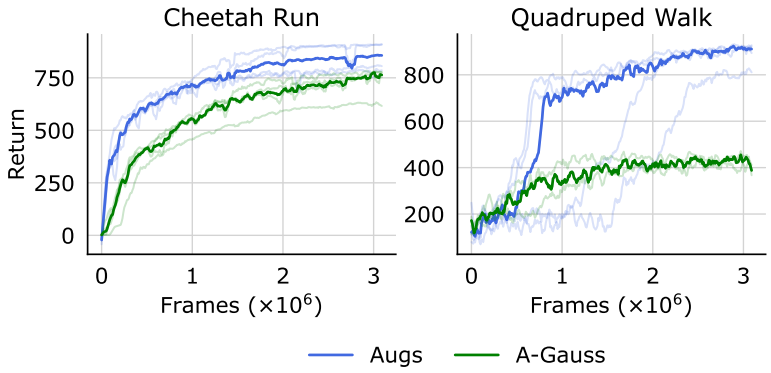
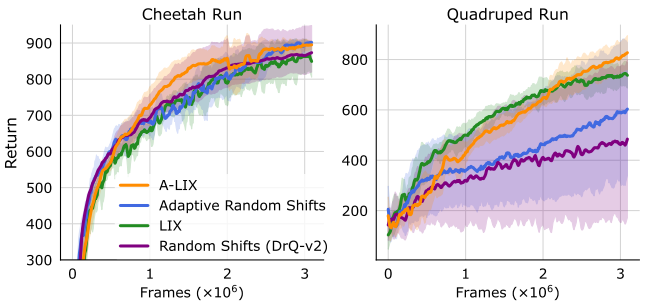


Figure 22. Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs.

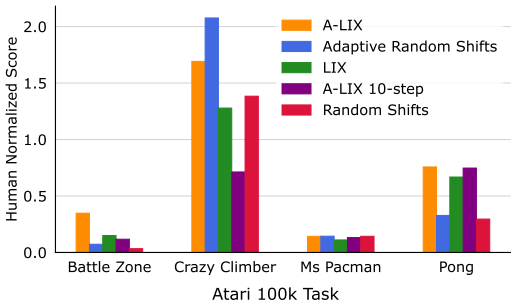
We see that while there is improvement over non-augmented agents, overall performance is still lower than even simple non-adaptive augmentation. We believe this is due to the Gaussian kernel having too significant an effect on the information contained in the feature gradients during backpropagation, causing information to be lost. We believe this explains the effectiveness of shift-augmentations in reinforcement learning, which is that they effectively balance the information contained in the gradients, as well as ensuring their smoothness to reduce overfitting.

E.3. Ablations to A-LIX

We now provide a set of ablations on both DMC and Atari, assessing the impact of individual components in A-LIX.



(a) DMC Control ablations in Cheetah Run (**left**) and Quadruped Run (**right**) evaluated over 4 seeds.



(b) Atari 100k ablations evaluated over 4 seeds in 4 different Atari 100k tasks.

Figure 23. An ablation study of A-LIX, showing the contribution of its individual components to ultimate performance in DMC and Atari 100k.

In Fig. 23a we choose the following ablations for DMC:

- A-LIX
- Adaptive Random Shifts (where the magnitude of the random shift image augmentation is adjusted using the dual ND objective)
- LIX
- Random Shifts (i.e., DrQ-v2)

While we see a slight asymptotic performance improvement in Cheetah Run by using LIX layers instead of random shifts, we notice significant differences in the less stable Quadruped Run environment. Concretely, we see much greater stability in both LIX approaches compared with image augmentation approaches, with the former having no failure seeds. Furthermore, we observe stronger asymptotic performance with the inclusion of the adaptive dual objective for both approaches. As motivated in Fig. 19, this is likely a result of reducing the shift parameter as the signal in the target values increases.

In Fig. 23b, we choose the following ablations for Atari 100k on a subset of environments that represent a diverse set of tasks and performances with baseline algorithms:

- A-LIX
- Adaptive Random Shifts (as before)
- LIX
- A-LIX with 10-step returns
- Random Shifts

We see that A-LIX performs consistently strongly across the environments tested, always placing in the top 2 with regards to Human Normalized Score. We also notice that generally, LIX layer methods outperform random shift methods apart from in Crazy Climber, where the opposite is true. We believe this may be due to random shift augmentations actually reflecting the inductive biases concerning generalization in this environment, and believe this merits further investigation. Finally, we observe that using 10-step returns instead of 3 generally harms performance with A-LIX, with justification given in App. C.2.

F. Additional Offline Experiment Analysis

F.1. Behavior Cloning without Augmentations

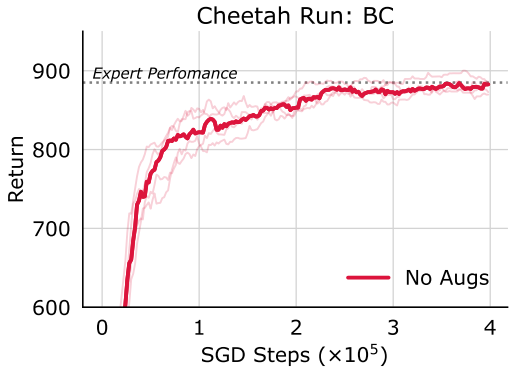


Figure 24. Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs. The grey dotted horizontal line represents mean expert performance.

To illustrate that test time shift invariance is not required, we show that it is possible to learn a policy through supervised learning. To do this, we generate a pixel-based dataset of 500,000 timesteps under an expert policy in Cheetah Run, and jointly train a CNN encoder and policy using behavior cloning/supervised learning by minimizing the loss $\mathcal{L} = (a - \pi(o))^2$ until convergence, where o follows the stacked frame image inputs of (Mnih et al., 2013). We see that the pixel-based policy performs as well as the behavior agent, despite using both higher dimensional data and fewer than half the samples compared to existing expert offline RL benchmarks from *proprioceptive* states (Fu et al., 2021).

This provides clear evidence that shift invariance is not required at test time, and motivates us to find an alternative explanation for why random shift augmentations help the learning process in TD-learning. An alternative perspective is that when the learning signal is strong, as is the case for supervised learning (and later stages during online learning when target values are more accurate), the natural bias of CNNs to learn lower order representations acts as an implicit regularizer (Rahaman et al., 2019) that results in test-time generalization.

F.2. Turning Off Augmentations

We present more evidence showing that augmentations benefit learning the most at the beginning of training. In Fig. 25 we show the effect of turning off augmentations at 200,000 steps in Cheetah Run, and at 500,000 in Quadruped Walk. In both instances, we see large improvements over not augmenting at all, and both nearly converge to the same value as DrQ-v2, showing further evidence that stability initially in learning is vital. We posit that turning off augmentations here did not yield similar benefits to Fig. 2 due to the fact that there is still high-frequency information in the targets (consider that the augmentations in Cheetah Run are switched off significantly earlier) that cause a marginal amount of self-overfitting, reducing the rate of learning due to feature space degeneration.

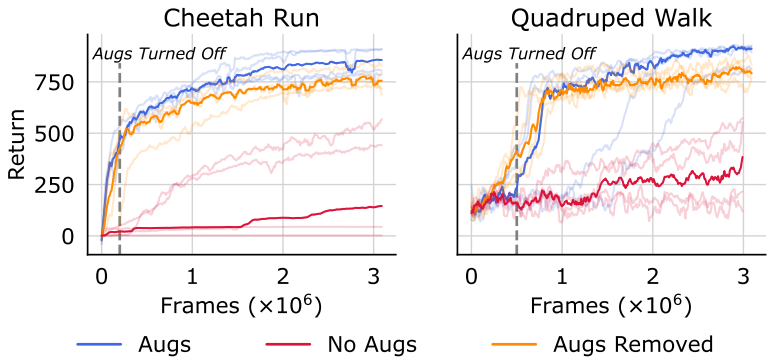


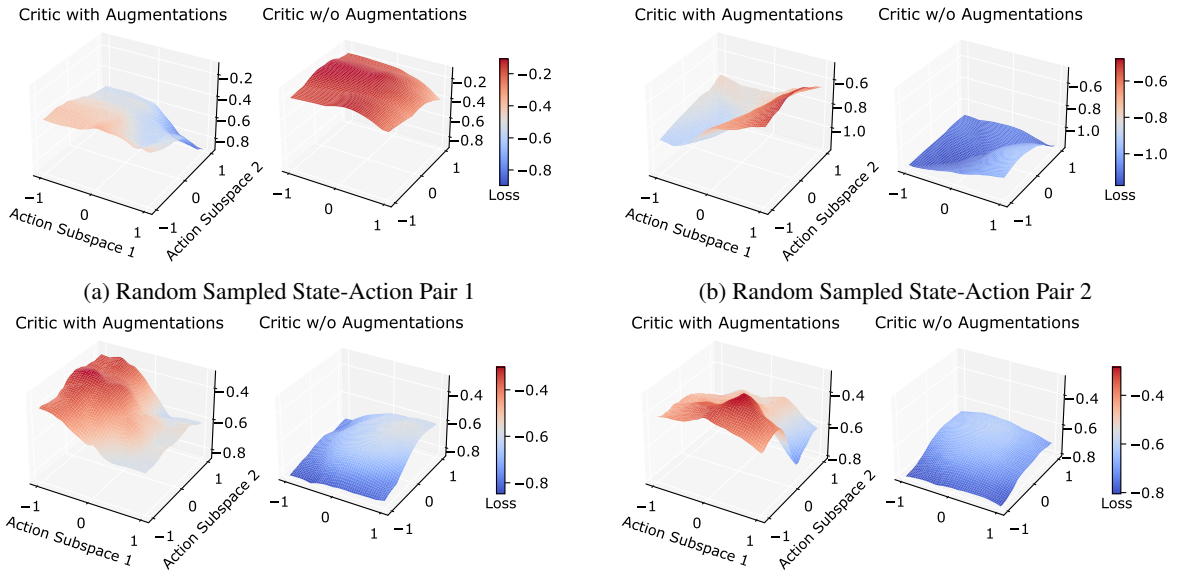
Figure 25. Returns of agents over 5 seeds. Solid lines represent median performance, faded lines represent individual runs. The grey dashed line shows when augmentations are turned off.

F.3. Action-Value Surfaces

Here we show the action-value surfaces of the offline agents' critics at various tuples sampled from the data. This provides us with an intuition over the loss landscape that the policies will be optimizing during the policy improvement, as accordingly the policy under the deterministic policy gradient (Silver et al., 2014) updates its own weights towards maximizing the action-values defined by the critic through the chain rule:

$$\nabla_\phi J_\pi \approx \mathbb{E}_{s\sim E}\left[\nabla_a Q_\theta(s,a)|_{a=f_\phi(s)} \nabla_\phi f_\phi(s)\right] \tag{10}$$

where ϕ and θ are policy and critic weights respectively. We hypothesize that self-overfitting reduces the sensitivity of the critic to actions, discarding important information regarding the causal link between actions and expected returns. To evaluate this, we sample state-action pairs from our replay buffer, and then visualize the action-value surface by sampling two random *orthogonal* direction vectors from the action space A. We then normalize the direction vectors to have a 2-norm of 1, and then multiply each direction vector by scalars $\alpha, \beta \in [-2, 2]$ respectively. We then plot the action-value surface as a result of adding the random vectors multiplied by their respective scalars onto the action sampled from offline dataset, giving us a 3-D surface. We clip actions to $a \in [-1, 1]^{|A|}$ as actions are squashed to this range in the policy through a truncated normal distribution.



(a) Random Sampled State-Action Pair 1

(b) Random Sampled State-Action Pair 2

(c) Random Sampled State-Action Pair 4

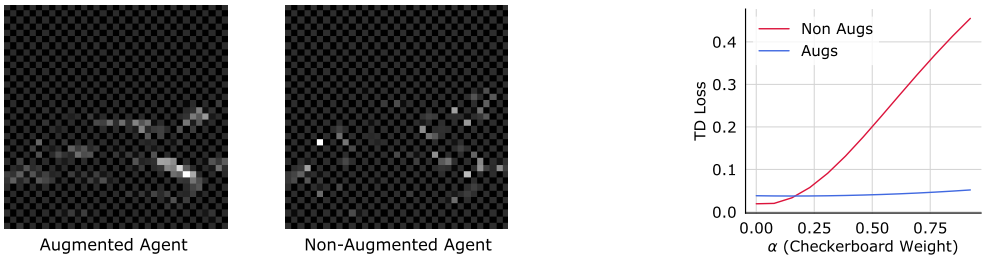
(d) Random Sampled State-Action Pair 4

Figure 26. Action-Value loss surface plotted with respect two orthogonal random directions sampled from the action space (i.e., $d_r \in A$ and $d_1 \perp d_2$).

We see that the critics learned by the augmented agents are more sensitive to changes in action. We believe this is due to the non-augmented agents overfitting to the observations, thus ignoring the lower-dimensional action inputs. To validate this, we sampled 128 random state-action tuples from the offline buffer, and calculated the average variance across the loss surfaces. We see a significant difference, with the augmented agent having an average loss surface variance of **0.0129**, whereas the non-augmented agent has an average loss surface variance of **0.0044**, suggestive of lower sensitivity.

F.4. Evidence of Critic MLP Overfitting from High-Frequency Features

We provide further evidence that measuring high-frequency features through the ND score is vital to understanding overfit by showing how overfitting is able to occur in the fully-connected critic layers, which are usually stable under proprioceptive observations (see Table 2). To do this, we construct a pattern containing high frequency checkerboard noise $c \in \mathbb{R}^{H\times W}$, and produce as many patterns as there are channels C in the final layer. To ensure consistency across each individual feature map, we normalize each checkerboard pattern by the maximum value in its respective feature map, and then divide by the width of the checkerboard. We then add this pattern multiplied by a scalar α onto each feature map.



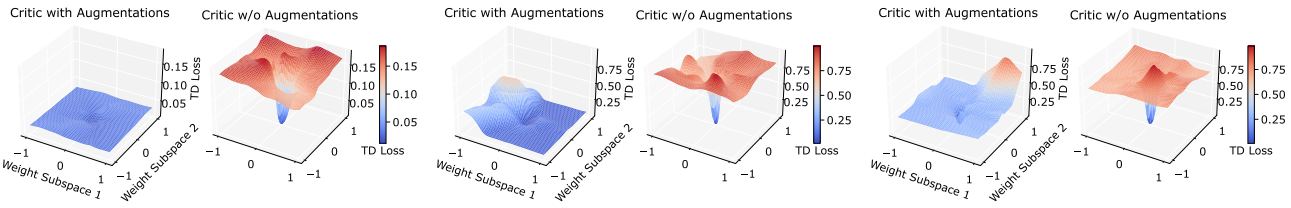
(a) Example checkerboard artefacts. (b) Sensitivity of agents to checkerboard artifact weight

Figure 27. Effect of checkerboard artifacts on feature maps and resultant loss sensitivity. We see the non-augmented agent is significantly more sensitive to this high-frequency noise.

As we see, the loss is significantly more sensitive to high-frequency perturbations in the non-augmented agent, justifying its reliance on high-frequency patterns in the feature maps to enable self-overfitting.

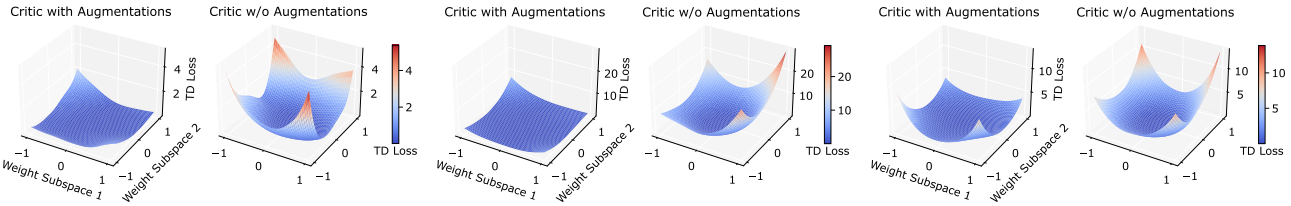
F.5. Additional Loss Surfaces

Here we show the loss surfaces of the offline agents under policy evaluation with at 1,000, 5,000, and 10,000 training steps. We also show the surfaces respect to only the MLP layers, again following the normalization approach of Li et al. (2018).



(a) 1,000 SGD Steps (b) 5,000 SGD Steps (c) 10,000 SGD Steps

Figure 28. Loss surface plotted with respect to *Encoder* parameters at various stages of training.



(a) 1,000 SGD Steps (b) 5,000 SGD Steps (c) 10,000 SGD Steps

Figure 29. Loss surface plotted with respect to *Critic MLP* parameters at various stages of training.

As we see, the loss surface with respect to the MLP parameters is significantly less sharp, lending further evidence that self-overfitting is predominately a result of the flexibility of the CNN layers to learn high-frequency features.