# Generalization of Reinforcement Learning with Policy-Aware Adversarial Data Augmentation

**Hanping Zhang** [1] **Yuhong Guo** [1 2]

## Abstract

The generalization gap in reinforcement learning (RL) has been a significant obstacle that prevents the RL agent from learning general skills and adapting to varying environments. Increasing the generalization capacity of the RL systems can significantly improve their performance on real-world working environments. In this work, we propose a novel policy-aware adversarial data augmentation method to augment the standard policy learning method with automatically generated trajectory data. Different from the observation transformation based data augmentations, our proposed method adversarially generates new trajectory data based on the policy gradient objective and aims to more effectively increase the RL agent's generalization ability with the policy-aware data augmentation. Moreover, we further deploy a mixup step to integrate the original and generated data to enhance the generalization capacity while mitigating the over-deviation of the adversarial data. We conduct experiments on a number of RL tasks to investigate the generalization performance of the proposed method by comparing it with the standard baselines and the state-of-the-art mixreg approach. The results show our method can generalize well with limited training diversity, and achieve the state-of-the-art generalization test performance.

## 1. Introduction

Benefiting from the power of deep neural networks, deep reinforcement learning (RL) has recently demonstrated incredible performance on many human-level challenging tasks. In addition to traditional board games like Go (Silver et al., 2016; 2017), deep reinforcement learning agents have defeated professional human players in large scale video games like StarCraft (Vinyals et al., 2017) and Dota 2 (Berner et al., 2019). Meanwhile, deep RL systems also suffer from the vulnerabilities of deep neural networks such as overfitting and data memorization, which often induce generalization gaps between the training and testing performances of the RL agents. The generalization gap prevents the RL agents from learning general skills in a simulation environment to handle the real-world working environments (Cobbe et al., 2020; Tobin et al., 2017). As a result, deep RL agents are reportedly performing poorly on unseen environments, especially when the training environments lack diversities (Zhang et al., 2018b;a; Cobbe et al., 2020; Song et al., 2019; Cobbe et al., 2019).

Towards the goal of reducing the generalization gap, previous works have exploited conventional data augmentation techniques such as cropping, translation, and rotation to augment the input observations and increase the diversity of training data in RL (Shorten & Khoshgoftaar, 2019; Cobbe et al., 2019; Lee et al., 2019; Laskin et al., 2020). Some have also explored the selection of data augmentation techniques (Raileanu et al., 2020; Jiang et al., 2021) and improved the RL architectures (Raileanu & Fergus, 2021; Ball et al., 2021). Recently, a mixture regularization method has been introduced to learn generalizable RL systems (Wang et al., 2020), which deploys *Mixup* to increase the data diversity and yields the state-of-the-art generalization performance. However, these methods work specifically at the level of the input observation data without taking the RL system into consideration, which prevents them from generating data that are most particularly suitable for the target RL system and hence could limit their performance's improvement.

In this paper, we propose a novel policy-aware adversarial data augmentation method with Mixup enhancement (PAADA+Mixup) to improve the generalization ability of policy learning based RL systems. This method first generates adversarial augmenting trajectory data by minimizing the expected rewards of the given RL policy based on the initial observed source trajectory data. Next it combines each adversarial augmenting trajectory and the corresponding ob-

[*] Equal contribution [1] School of Computer Science, Carleton University, Canada [2] CIFAR AI Chair, Amii, Canada. Correspondence to: Hanping Zhang <jagzhang@cmail.carleton.ca>, Yuhong Guo <yuhong.guo@carleton.ca>.

*Decision Awareness in Reinforcement Learning Workshop at the 39th International Conference on Machine Learning (ICML)*, Baltimore, Maryland, USA, 2022. Copyright 2022 by the author(s).

servation source trajectory together by randomly selecting each observation step from one of them. The combined trajectory data can then be used to update the policy of the RL system. As deep RL systems typically inherit the vulnerability of deep neural networks to adversarial examples (Lin et al., 2017), some previous works have investigated the topic of adversarial attacks in deep RL (Zhao et al., 2020; Gleave et al., 2019). Our work however is the first that deploys adversarial data augmentation in online RL systems to improve their generalization capacity. The proposed adversarial augmentation is conducted in a policy-aware manner to induce direct impact on the to-be-learned policy. Moreover, we further deploy a mixup operation on the combined trajectory to enhance the robustness and generalization of the RL system. We conduct extensive experiments on several RL tasks to investigate the proposed approach under different generalization test settings. The experimental results show the proposed PAADA greatly outperforms the strong RL baseline method, proximal policy optimization (PPO) (Schulman et al., 2017). With the mixup enhancement, PAADA+Mixup can achieve the state-of-the-art performance, surpassing mixreg (Wang et al., 2020) with notable performance gains.

The main contributions of the proposed work can be summarized as follows:

- We introduce adversarial data augmentation to deep RL and develop the very first policy-aware adversarial data generation method to improve the generalization capacity of deep RL agents.
- We integrate the generalization strengths of both adversarial data generation and mixup and demonstrate superior empirical performance than using either one of them alone.
- We conduct experiments on the Procgen benchmark with different generalization settings. Our proposed method demonstrates good generalization performance with limited training environments and outperforms the state-of-the-art mixreg approach.

## 2. Related Works

Generalization gap has been an increasing concern in deep reinforcement learning. Recent studies (Zhang et al., 2018b;a) show that the main cause of generalization gap is the over-fitting and memorization inherited from deep neural networks (Arpit et al., 2017). Data augmentation is a conventional technique for solving over-fitting in deep learning (Shorten & Khoshgoftaar, 2019). In recent studies, some data augmentation approaches broadly applied in deep learning have been brought into reinforcement learning. Cobbe et al. (2019) introduce the cutout technique into deep RL, where data are generated by partially blocking the input observations with randomly generated black occlusions. Lee et al. (2019) propose a randomized convolutional network to perturb the input observations. Laskin et al. (2020) propose to integrate the commonly used data augmentation skills such as cropping, translation, and rotation to improve the generalization of RL. Ball et al. (2021) introduce augmented world models to specifically address the generalization of model-based offline RL problems. Raileanu et al. (2020) propose a data-regularized actor-critic approach to regularize policy and value functions when applying data augmentation, whereas the upper confidence bound (Auer, 2002) is borrowed to select the proper data augmentation method. Wang et al. (2020) propose a simple but efficient mixture regularization approach, *mixreg*, to improve the generalization capacity in RL systems. Following the mixup method (Zhang et al., 2017) in supervised learning, *mixreg* generates new observations from two randomly selected observations through linear combinations. The deep RL agents trained on mixreg augmented observations demonstrate significant improvements in generalization.

Inspired by the success of data augmentation on generalization of deep reinforcement learning, we introduce a novel adversarial data augmentation technique to deep RL. Deep neural networks are known to be vulnerable to adversarial examples (Szegedy et al., 2013). Many previous works have studied the attack and defence strategies of deep neural networks regarding adversarial examples (Yuan et al., 2019; Madry et al., 2017; Akhtar & Mian, 2018). Lin et al. (2017) show deep reinforcement learning agents inherited the vulnerability of deep neural networks to adversarial examples. They propose two tactics to perform adversarial strategies to attack RL agents: the strategically-timed attack minimizes the agent's reward for small subsets of time steps, while the enchanting attack lures the agent to a specific target state. Following this, a few works have further contributed to this rising topic of adversarial attacks in deep RL. Zhao et al. (2020) propose an approach to generate adversarial observations without previous knowledge on the network architecture and RL algorithm of the deep reinforcement learning agent. Instead of adding perturbations to the observations, Gleave et al. (2019) propose to train the agent with an adversarial policy.

Distinct from adversarial attacks, our study focuses on adversarial data augmentation. Adversarial data augmentation has been investigated in a number of works on supervised learning, but has not been explored to enhance generalization in deep RL systems. Goodfellow et al. (2014) find the linear property of deep neural networks is vulnerable to adversarial perturbations and propose to train the supervised model with adversarial examples to improve generalization. Sinha et al. (2017) propose a Lagrangian formulation of adversarial perturbations in a Wasserstein ball (Lee &

Raginsky, 2017) to enhance the robustness of deep learning models. Volpi et al. (2018) further propose to exploit adversarial data augmentation for domain adaptation with unknown target domains.

## 3. Method

This study focuses on increasing the generalization ability of a deep RL agent. Following the previous generalization study in RL (Wang et al., 2020), we consider the following RL setting. The agent is trained on a set of $n$ environments $\{\mathcal{K}_1, ..., \mathcal{K}_n\}$ sampled from a distribution $p(\mathcal{K})$ to learn an optimal policy $\pi^*$, and then tested on another set of environments $\{\hat{\mathcal{K}}_1, ..., \hat{\mathcal{K}}_m\}$ sampled from $p(\mathcal{K})$. Its generalization performance is measured as the zero-shot expected cumulative reward in the test environments:

$$\mathbb{E}_{\tau\sim\mathcal{D}_{\pi^*}^{test}} \sum_{t=0}^{T} \gamma_*^t r_t \tag{1}$$

where $\tau$ denotes a trajectory $(s_0, a_0, r_0, s_1, a_1, r_1, \cdots, r_T)$, $\mathcal{D}_{\pi^*}^{test}$ denotes the distribution of $\tau$ in the test environments under policy $\pi^*$, and $\gamma_* \in (0, 1]$ is the discount factor. Moreover, we assume the training environments can be much fewer than the test environments such as $n = \lceil \xi m \rceil$ with $\xi \in (0, 1]$. The goal is to train an optimal policy $\pi^*$ that can generalize well in terms of the expected cumulative test reward above in Eq.(1). Towards this goal, in this section we present a policy-aware adversarial data augmentation with mixup enhancement (PAADA+Mixup) method for the RL training process.

### 3.1. Policy-Aware Adversarial Data Augmentation

When the training and test environments are different, a standard RL learning algorithm, e.g, Proximal Policy Optimization (PPO) (Schulman et al., 2017), will inevitably suffer from the domain gap between the training and test environments, and demonstrate generalization gaps between the training and test performances. Inspired by the effectiveness of adversarial data augmentation in supervised learning (Volpi et al., 2018), we propose to augment the deep RL process by generating adversarial trajectories from the current policy, aiming to adaptively broaden the experience of the RL agent and increase its generalization capacity to unseen test environments.

Specifically, given the current parametric policy $\pi_\theta$ with parameter $\theta$, in each epoch of the policy optimization based RL training, the standard procedure is to collect a set of trajectories $\{\tau_1, \cdots, \tau_n\}$ from the training environments, and then update the policy parameter $\theta$ by performing gradient ascent with respect to the policy optimization objective over the observed trajectories. For policy gradient, the following surrogate objective is often used:

$$L^{PG}(\theta) = \hat{\mathbb{E}}_t[\log \pi_\theta(a_t|s_t)\hat{A}_t] \tag{2}$$

where $\hat{\mathbb{E}}_t[\cdot]$ denotes the empirical average over the set of transitions in the collected trajectories; $\hat{A}_t$ is the estimated advantage function at timestep $t$, and can be approximated as $\hat{A}_t = r_t - V(s_t)$, where $V(s_t)$ is the value function at state $s_t$ (Degris et al., 2012). For the more advanced high-performance policy gradient algorithm PPO (Schulman et al., 2017), a clipping modulated objective is typically used:

$$L^{PPO\text{-}C}(\theta) = \hat{\mathbb{E}}_t\Big[\min\Big(\rho_\theta\hat{A}_t, \text{clip}\,(\rho_\theta, 1-\epsilon, 1+\epsilon)\hat{A}_t\Big)\Big] \tag{3}$$

where $\rho_\theta = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta'}(a_t|s_t)}$ and $\epsilon$ is a small constant. This PPO objective regularizes the new policy $\pi_\theta$ from being severely deviated from the previous policy $\pi_{\theta'}$ and aims to avoid large destructive policy updates associated with the vanilla policy gradient. Although PPO has demonstrated great performance in standard RL, it may be overly bounded to the available training observations and hence yields poor test performance in the generalization settings. We therefore propose to expand the observation space by generating adversarial examples based on the collected trajectories.

#### 3.1.1. Adversarial Trajectory Generation

For each observed trajectory $\tau$, we generate an adversarial example for each of its observation points (i.e., transitions), $P_t = (x_t, y_t)$ with $x_t = s_t$ and $y_t = (a_t, r_t)$. That is, we find the worst example $P = (x, y)$ in the close neighborhood of the current point $P_t = (x_t, y_t)$ by minimizing the cumulative award objective $L$ the RL agent needs to maximize, such as

$$\min_P \ L(\theta; P) \qquad \text{s.t. } D(P, P_t) \le \rho \tag{4}$$

where $D(\cdot, \cdot)$ denotes a distance metric such as the Wasserstein distance and the constraint bounds the point $P$ to be within the neighborhood of $P_t$. For two points $P = (x, y)$ and $P_t = (x_t, y_t)$, the Wasserstein distance $D(P, P_t)$ can be specified as follows through a transportation cost $c$ (Volpi et al., 2018):

$$\begin{aligned} D(P, P_t) &= c((x, y), (x_t, y_t)) \\ &= ||x - x_t||^2 + \infty \cdot \mathbf{1}\{y \neq y_t\} \end{aligned} \tag{5}$$

Moreover, the constrained problem in Eq.(4) can be equivalently reformulated as the following regularized optimization problem with a proper Lagrangian parameter $\gamma$:

$$\begin{aligned} &\min_P \ L(\theta; P) + \gamma D(P, P_t) \\ \Longleftrightarrow \quad &\min_x L(\theta; x, y_t) + \gamma\|x - x_t\|^2 \end{aligned} \tag{6}$$

**Algorithm 1** Adversarial Observation Generation

```
Input: $\pi_\theta, V,\ (s_t, a_t, r_t);$ stepsize $\eta$,
       maximum step number $K_{max}$, tolerance $\epsilon_a$
Output: adversarial observation $s$
$s = s_t$
for $k = 1, ..., K_{max}$ do
    if $\|\nabla_s[\log \pi_\theta(a_t|s)(r_t - V(s)) + \gamma(s - s_t)^2]\|^2 < \epsilon_a$ then break
    $s = s - \eta\nabla_s[\log \pi_\theta(a_t|s)(r_t - V(s)) + \gamma(s - s_t)^2]$
```

As this point generation is conducted under the current policy $\pi_\theta$ and does not involve policy update, we use the policy gradient objective $L^{PG}$ as the objective $L$ for adversarial example generation due to its simplicity and easy computation. Therefore for each observation point $(s_t, a_t, r_t)$ in the collected trajectory data, its corresponding adversarial point $(\hat{s}_t, a_t, r_t)$ will be generated as follows:

$$\begin{aligned}\hat{s}_t &= \underset{s}{\operatorname{argmin}}\ L^{PG}(\theta; s, t) + \gamma D(s, s_t)\\ &= \underset{s}{\operatorname{argmin}}\ \log \pi_\theta(a_t|s)\hat{A}_t + \gamma\|s - s_t\|^2\\ &= \underset{s}{\operatorname{argmin}}\ \log \pi_\theta(a_t|s)(r_t - V(s)) + \gamma\|s - s_t\|^2 \quad (7)\end{aligned}$$

We can solve this generation problem by performing gradient descent with a stepsize $\eta$ and a maximum step number $K_{max}$. The algorithm is shown in Algorithm 1. With this simple procedure, for each observation trajectory $\tau$, we can produce a corresponding adversarial trajectory $\hat{\tau}$ by generating an adversarial point $(\hat{s}_t, \hat{a}_t = a_t, \hat{r}_t = r_t)$ for each observation point $(s_t, a_t, r_t)$ in $\tau$.

### 3.1.2. Trajectory Augmentation

Given the observed source trajectory $\tau$ and the generated adversarial trajectory $\hat{\tau}$, simply appending one after the other to fed to the deep RL agent for training turns out not to be a suitable solution, as it may cause the policy parameter update to dramatically switch between very different directions. Moreover, the suitable degree of augmentation could also vary for different RL tasks. Here we propose to augment the source trajectory $\tau$ with the adversarial trajectory $\hat{\tau}$ by combining them into a new trajectory $\overline{\tau}$ with an augmentation degree $\nu \in [0, 1]$. Specifically, we construct the new trajectory $\overline{\tau}$ by randomly selecting $\lfloor \nu \cdot |\tau| \rfloor$ points (transitions) from the adversarial trajectory $\hat{\tau}$ and taking the other $\lceil (1 - \nu) \cdot |\tau| \rceil$ points from the original trajectory $\tau$. In this way, we not only can better blend the adversarial points with the original observations, but also have control over the contribution degree of the augmentation data through the hyperparameter $\nu$.

We deploy this policy-aware adversarial data augmentation scheme on the PPO method. The overall training algorithm is depicted in Algorithm 2, which first pre-trains the RL agent for $K_{pre}$ epochs and then performs adversarial data augmentation in the ensuing epochs.

**Algorithm 2** Adversarial Data Augmentation on PPO

```
Input: initial policy parameter $\theta$, initial value function
       parameter $\phi$, the pre-training epoch number
       $K_{pre}$, the augmentation degree $\nu$
Output: trained policy $\pi_\theta$
for $k = 1, 2, ...$ do
    Collect a set of trajectories $\{\tau_1, \tau_2, ..., \tau_n\}$ by
     running policy $\pi_\theta$ on the training environments
    for $\tau_i$ in $\{\tau_1, \tau_2, ..., \tau_n\}$ do
        Compute the advantage estimates $\{A_t\}$ for all
         the $t$ transition points in $\tau_i$
        if $k < K_{pre}$ then
            $\bar{\tau}_i = \tau_i$; continue
        for $t = 0, 1, 2, ..., |\tau_i| - 1$ do
            Generate the adversarial state $\hat{s}_t$ with Eq.(7)
            $\hat{A}_t = r_t - V_\phi(\hat{s}_t)$
            Add $(\hat{s}_t, a_t, r_t, \hat{A}_t)$ into the augmentation
             trajectory $\hat{\tau}_i$
        Combine $\tau_i$ and $\hat{\tau}_i$ into an augmented trajectory
         $\overline{\tau}_i$ with the augmentation degree $\nu$
        %[place holder for additional step]
    Update the policy function parameter $\theta$ by maximiz-
    ing the PPO-Clip objective in Eq.(3) on the augmented
    trajectories $\{\overline{\tau}_1, \overline{\tau}_2, ..., \overline{\tau}_n\}$
    Update the value function parameter $\phi$ on the aug-
    mented trajectories
```

## 3.2. Enhancement with Mixup

In addition to the adversarial data augmentation technique above, we consider further enhancing the diversity of the training data with a Mixup procedure. Mixup generates data points through linear interpolation and has demonstrated effective performance in both supervised learning (Zhang et al., 2017) and reinforcement learning (Wang et al., 2020). Here we propose to deploy the Mixup procedure on each augmented trajectory $\overline{\tau}_i$ generated above. Specifically, for a trajectory $\overline{\tau}_i$ with $|\overline{\tau}_i|$ transition points, we first make a copy of $\overline{\tau}_i$ as $\overline{\tau}'_i$, and randomly shuffle the indices $\{0, 1, \cdots, |\overline{\tau}_i| - 1\}$ into $\{I_0, I_1, \cdots, I_{|\overline{\tau}_i|-1}\}$. Then we linearly combine $\overline{\tau}_i$ and $\overline{\tau}'_i$ with the following mixup steps:

$$\bar{s}_t = \lambda \bar{s}_t + (1 - \lambda)\bar{s}'_{I_t} \quad (8)$$

$$\bar{r}_t = \lambda \bar{r}_t + (1 - \lambda)\bar{r}'_{I_t} \quad (9)$$

$$\bar{A}_t = \lambda \bar{A}_t + (1 - \lambda)\bar{A}'_{I_t} \quad (10)$$

while $\bar{a}_t$ is set as $\bar{a}_t$ with probability $\lambda$ and set as $\bar{a}'_{I_t}$ with probability $1 - \lambda$. The hyperparameter $\lambda$ is sampled from a beta distribution $\lambda \sim B(\alpha, \beta)$. Normally, the parameters of the beta distribution are set to $\alpha = \beta$ as suggested in (Zhang et al., 2017). However, when the training environments are limited, it is beneficial to have different $\alpha$ and $\beta$ values to shift the mean value of the $\lambda$ samples. This Mixup step can be deployed on the augmented trajectory and performed in line 12 within the trajectory loop in Algorithm 2.

## 4. Experiments

We conducted experiments to validate the empirical performance of the proposed method under different generalization settings. In this section, we report our experimental settings and results.

### 4.1. Experiment Setting

We conduct experiments on the Procgen benchmark (Cobbe et al., 2020), which contains procedurally generated environments designed to test the generalization ability of deep RL agents. Each environment takes visual input and has significant change among different levels of the environments. A Procgen environment can generate a maximum of 500 different levels for the RL generalization task. We choose 4 game environments (starpilot, dodgeball, climber, fruitbot) from this benchmark as different RL tasks and treat different levels of each RL task as different training and test environments from the generalization perspective. Follow the settings in (Cobbe et al., 2020; Wang et al., 2020), we do not limit the levels in the testing environments and use the total $m = 500$ levels for testing, while a relatively smaller number, $n = \lceil \xi m \rceil$, of level environments, are sampled for training. In particular, we consider different $\xi$ values such as $\xi \in \{0.25, 0.5, 1\}$. A smaller $\xi$ value indicates a more difficult generalization setting as the diversity of training environments is reduced.

We adopt PPO (Schulman et al., 2017) as our RL baseline, although the proposed methodology in principle can be generalized into other RL methods as well. In addition, we compare our approach, PAADA+Mixup, to the state-of-the-art generalization method, mixreg, which has been shown to outperform the conventional data augmentation techniques in (Wang et al., 2020). Following the Procgen benchmark, we adopt the same convolutional neural network architecture as IMPALA (Espeholt et al., 2018). We use the mean episode return in each epoch of the zero-shot testing as the generalization evaluation metric on each of the four Procgen benchmark games (starpilot, dodgeball, climber, fruitbot). Moreover, following (Cobbe et al., 2020) we compute the mean normalized return over the four games to summarize the overall generalization performance.

**Hyperparameters** For the adversarial observation generation, the stepsize $\eta$ is set to 10, the maximum number of steps $K_{max}$ is set to 50, and the tolerance $\epsilon_a$ is set to $5e^{-6}$. For the training algorithm of the proposed approach, PAADA, we set the pre-training epoch number, $K_{pre}$, as 50. That is, in the first 50 training epochs, the deep RL agent is trained with standard PPO (Schulman et al., 2017). After 50 epochs, our augmentation method is applied to enhance the generalizability. In the evaluation plots, the epoch number does take these pre-training epochs into account. The Lagrangian hyperparameter $\gamma$ in Eq.(6) and Eq.(7) is set to 0.01. In each training epoch, we collect $n$ trajectories and each trajectory has 256 transitions. For testing, $m$ trajectories, one from each testing environment, are used. The discount factor $\gamma_*$ is set to 0.999. Each experiment is tested on 3054 training epochs in total. For the Mixup procedure, we use $\alpha = 0.2$ and $\beta = 0.2$ for $\xi = 1$, and increase $\beta$ to $\beta = 0.5$ for $\xi = 0.5$ and to $\beta = 1$ for $\xi = 0.25$.

### 4.2. Experiments with Full Set of Training Environments

In the first set of less challenging experiments, for each game, we have $\xi = 1$ and use the full set of 500 training environments, while testing on unlimited environments. For the proposed approach PAADA, we tested two different variants of it: (1) PAADA ($\nu = 0.5$), which uses an augmentation degree $\nu = 0.5$ and indicates that 50% of the observation transitions of the adversarial trajectory are randomly merged into the original observation trajectory. (2) PAADA+Mixup ($\nu = 0.5$), which indicates the Mixup enhancement is added to PAADA ($\nu = 0.5$). We compared these variants with the PPO baseline and one state-of-the-art method, mixreg.

The training and test evaluation results on the four games are reported in Figure 1, where dotted lines are used to present the training evaluation results and solid lines are used to present the test evaluation results. The test results demonstrate the working performance of the system. The left four plots in Figure 1 report the comparison results on the four games separately. We can see that the policy-aware adversarial data augmentation alone outperforms the baseline PPO: PAADA($\nu = 0.5$) greatly improves the generalization performance of PPO on the four games and the performance gain in terms of the mean normalized return is very notable. With the Mixup enhancement, PAADA+Mixup($\nu = 0.5$) further improves its generalization performance, and yields similar performance to the state-of-the-art method, mixreg, on three games, *dodgeball, fruitbot* and *starpilot*, and outperforms mixreg on *climber* with large margins. The right plot in Figure 1 summarizes the overall performance over the four games in terms of a mean normalized return metric. The mean normalized return is a useful metric that allows us to compare the overall performance of different reinforcement learning systems across multiple environments (Cobbe et al., 2020). We can see from the plot

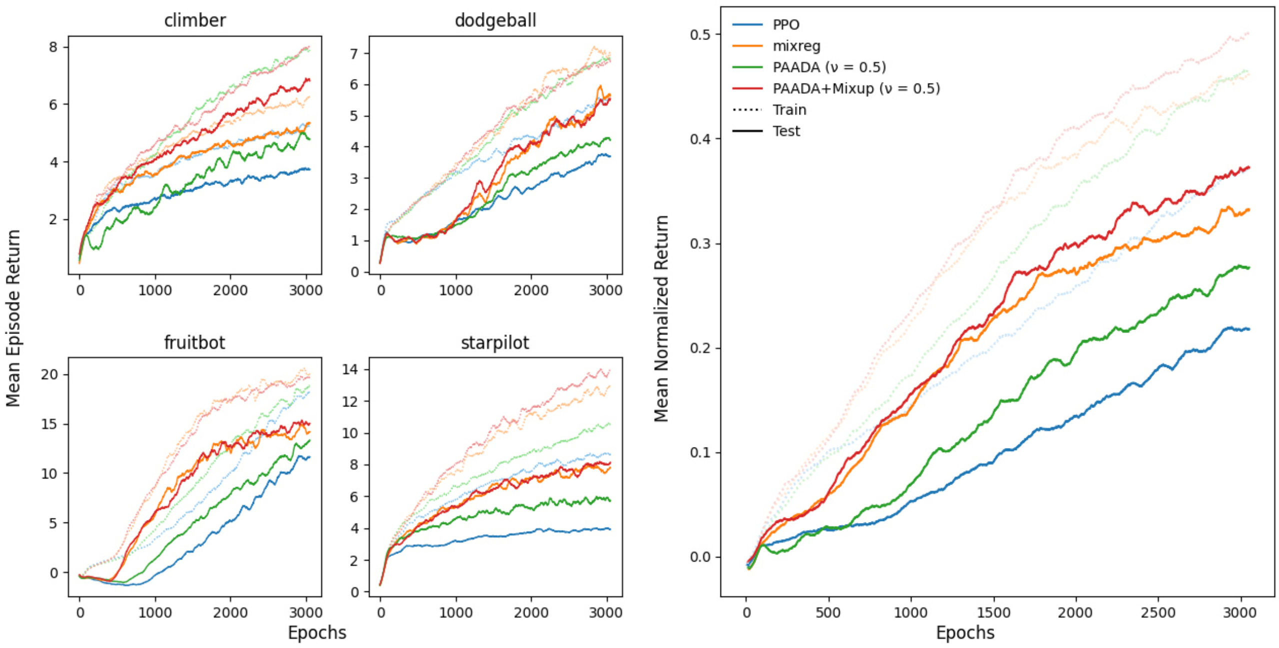

*Figure 1.* Training and testing performance on full training environments ($\xi = 1$). **Left**: Mean episode return per epoch on different game environments. **Right**: Mean normalized return over all the four game environments.

that PAADA+Mixup($\nu = 0.5$) outperforms all the other methods, including mixreg, with distinct performance gains. These results clearly demonstrate the efficacy of the proposed adversarial based data augmentation and mixup.

### 4.3. Experiments on Partial Set of Training Environments

In this set of experiments, we consider more challenging generalization settings with $\xi < 1$. That is, the number of training environments are much smaller and hence the training diversity is greatly reduced. In particular, we consider $\xi = 0.5$ and $\xi = 0.25$. In such settings, with limited training diversity, it is more difficult to achieve better generalization.

Again we compared the two variants of the proposed method, PAADA($\nu = 0.5$) and PAADA+Mixup($\nu = 0.5$), with PPO and mixreg on the four game tasks. Figure 2 reports the testing results of the comparison methods with 50% training environments, i.e., $\xi = 0.5$. We can see that in this setting, both PAADA($\nu = 0.5$) and PAADA+Mixup($\nu = 0.5$) outperform PPO on all the four games with clear performance gains. Between the two variants, mixup still produces certain improvements on three games, *starpilot, dodgeball* and *fruitbot*, whereon PAADA+Mixup($\nu = 0.5$) outperforms PAADA($\nu = 0.5$). Moreover, PAADA+Mixup($\nu = 0.5$) outperforms mixreg on three games, *climber, fruitbot* and *starpilot*, while mixreg produces the best performance on *dodgeball*. Nevertheless, as shown in the right plot of Figure 2, PAADA+Mixup($\nu = 0.5$) demonstrates a consistent and clear advantage over mixreg in terms of the overall mean normalized return.

Figure 3 reports the results of the comparison methods with 25% training environments, i.e., $\xi = 0.25$. This setting is more challenging than the one above with $\xi = 0.5$. In this setting, we can see that PAADA+Mixup($\nu = 0.5$) still outperforms mixreg on *dodgeball, fruitbot* and *starpilot*, but produces inferior result on *climber*, whereon PAADA($\nu = 0.5$) produces the best results. In terms of mean normalized return, as shown in the right plot of Figure 3, all three data augmentation methods demonstrate similar superior generalization capacity over the baseline PPO, All these experiments show our propose adversarial data augmentation with mixup enhancement can effectively increase the generalization capacity of RL systems.

### 4.4. Overall Generalization Capacity Comparison

To demonstrate a more concrete comparison between all the methods and across different settings and games, we collected the test results for the last 100 of the 3054 training epochs and reported the mean and standard deviation of these test returns in Table 1. From these concrete result numbers we can see that all the three methods with generalization strategies, PAADA($\nu = 0.5$), PAADA+Mixup($\nu =$

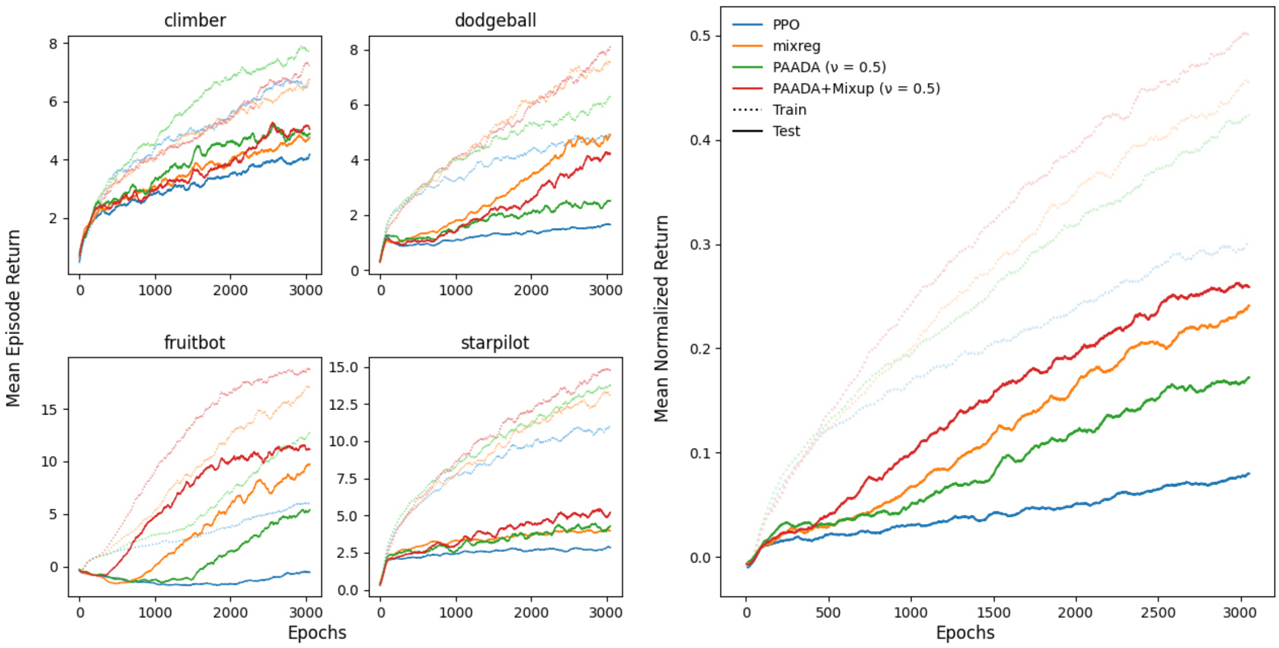

*Figure 2.* Training and testing performance on partial training environments ($\xi = 0.5$). **Left**: Mean episode return per epoch on different game environments. **Right**: Mean normalized return over all the four game environments

*Table 1.* Mean and standard deviation of the average test returns among 100 test epochs. MNR: mean normalized return over all the four games, which is a criterion that shows the overall performance of a generalization method across different games. The best result in each setting is shown in bold font.

| Environment ($\xi = 1$) | PPO | mixreg | PAADA ($\nu = 0.5$) | PAADA+Mixup ($\nu = 0.5$) |
|---|---|---|---|---|
| climber | $3.74 \pm 0.56$ | $5.31 \pm 0.57$ | $4.79 \pm 0.55$ | $\mathbf{6.79 \pm 0.59}$ |
| dodgeball | $3.70 \pm 0.45$ | $\mathbf{5.64 \pm 0.62}$ | $4.24 \pm 0.42$ | $5.48 \pm 0.56$ |
| fruitbot | $11.58 \pm 1.34$ | $14.12 \pm 1.40$ | $13.19 \pm 1.12$ | $\mathbf{14.89 \pm 1.18}$ |
| starpilot | $3.94 \pm 0.46$ | $7.67 \pm 0.84$ | $5.74 \pm 0.92$ | $\mathbf{8.09 \pm 0.87}$ |
| MNR | $0.22 \pm 0.02$ | $0.33 \pm 0.02$ | $0.28 \pm 0.02$ | $\mathbf{0.37 \pm 0.02}$ |
| Environment ($\xi = 0.5$) | PPO | mixreg | PAADA ($\nu = 0.5$) | PAADA+Mixup ($\nu = 0.5$) |
| climber | $4.14 \pm 0.52$ | $4.72 \pm 0.51$ | $4.87 \pm 0.54$ | $\mathbf{5.09 \pm 0.56}$ |
| dodgeball | $1.64 \pm 0.23$ | $\mathbf{4.93 \pm 0.54}$ | $2.53 \pm 0.44$ | $4.23 \pm 0.67$ |
| fruitbot | $-0.55 \pm 0.58$ | $9.66 \pm 1.19$ | $5.26 \pm 1.27$ | $\mathbf{11.13 \pm 1.32}$ |
| starpilot | $2.85 \pm 0.34$ | $3.98 \pm 0.39$ | $4.27 \pm 0.88$ | $\mathbf{5.10 \pm 0.94}$ |
| MNR | $0.08 \pm 0.01$ | $0.24 \pm 0.02$ | $0.17 \pm 0.02$ | $\mathbf{0.26 \pm 0.02}$ |
| Environment ($\xi = 0.25$) | PPO | mixreg | PAADA ($\nu = 0.5$) | PAADA+Mixup ($\nu = 0.5$) |
| climber | $3.05 \pm 0.47$ | $3.07 \pm 0.55$ | $\mathbf{3.64 \pm 0.62}$ | $2.08 \pm 0.46$ |
| dodgeball | $1.12 \pm 0.18$ | $2.10 \pm 0.30$ | $1.82 \pm 0.47$ | $\mathbf{2.38 \pm 0.46}$ |
| fruitbot | $-1.90 \pm 0.39$ | $2.48 \pm 0.81$ | $1.19 \pm 0.98$ | $\mathbf{4.02 \pm 1.06}$ |
| starpilot | $2.30 \pm 0.28$ | $3.03 \pm 0.34$ | $2.79 \pm 0.91$ | $\mathbf{3.13 \pm 1.02}$ |
| MNR | $0.03 \pm 0.01$ | $\mathbf{0.09 \pm 0.01}$ | $\mathbf{0.09 \pm 0.02}$ | $\mathbf{0.09 \pm 0.02}$ |

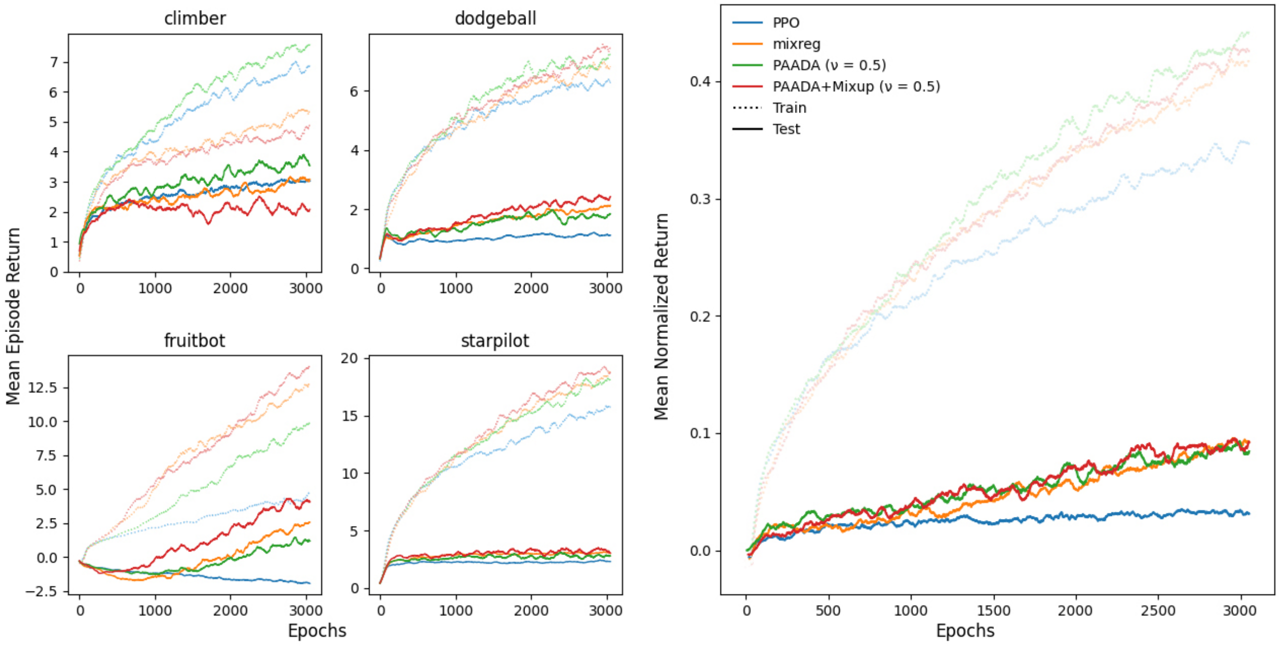

*Figure 3.* Testing performance on partial training environments ($\xi = 0.25$). **Left**: Mean episode return per epoch on different game environments. **Right**: Mean normalized return over all the four game environments

$0.5$), and mixreg, consistently outperform PPO across different game tasks and different generalization settings. This suggests both adversarial data augmentation and mixup are individually effective in improving the generalization performance of the baseline RL. By effectively integrating both adversarial data augmentation and mixup enhancement, our proposed PAADA+Mixup($\nu = 0.5$) outperforms the state-of-the-art mixreg across almost all cases (12 out of 15 cases) except on dodgeball with $\xi = 1, 0.5$ and on climber with $\xi = 0.25$. These results again validated the efficacy of the proposed method.

## 5. Conclusion and Future Work

In this paper, we proposed a novel policy-aware adversarial data augmentation method with Mixup enhancement (PAADA+Mixup) to improve the generalization capacity of RL systems. It generates augmenting trajectories by adversarially minimizing the expected reward that a RL agent would *desire to maximize*, and then uses them to augment the original trajectories under controlled augmentation degrees. Moreover, a mixup operation is further deployed to enhance the diversity of the augmented trajectory. This is the first work that deploys adversarial data augmentation to learn generalizable RL systems in an online manner. It also presents the first experience of integrating adversarial augmentation and mixup generalization. We conducted experiments on the Procgen benchmark by comparing the proposed method with both the baseline PPO and the state-of-the-art method under different generalization settings. The results show PAADA surpasses PPO in general, while PAADA+Mixup outperforms the state-of-the-art mixreg with notable performance gains, especially in the challenging generalization settings.

Our empirical study indicates that when the generalization setting is very challenging—that is, there are very limited training environments, the performance gains of the existing generalization methods can decrease substantially. This is a fundamental challenge for methods that focus on generalizing the training data. Our future work aims to tackle this challenge by extending the generalization effort into the test time. In addition to the training time generalization, we will consider enhancing the performance of RL systems by performing test time adaptation.

## References

Akhtar, N. and Mian, A. Threat of adversarial attacks on deep learning in computer vision: A survey. *Ieee Access*, 6:14410–14430, 2018.

Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al. A closer look at memorization in deep networks. In *International Conference on Machine Learning*, pp. 233–242. PMLR, 2017.

Auer, P. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.

Ball, P. J., Lu, C., Parker-Holder, J., and Roberts, S. Augmented world models facilitate zero-shot dynamics generalization from a single offline environment. *arXiv preprint arXiv:2104.05632*, 2021.

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.

Cobbe, K., Klimov, O., Hesse, C., Kim, T., and Schulman, J. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, pp. 1282–1289. PMLR, 2019.

Cobbe, K., Hesse, C., Hilton, J., and Schulman, J. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*, pp. 2048–2056. PMLR, 2020.

Degris, T., Pilarski, P. M., and Sutton, R. S. Model-free reinforcement learning with continuous action in practice. In *2012 American Control Conference (ACC)*, pp. 2177–2182, 2012.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pp. 1407–1416. PMLR, 2018.

Gleave, A., Dennis, M., Wild, C., Kant, N., Levine, S., and Russell, S. Adversarial policies: Attacking deep reinforcement learning. *arXiv preprint arXiv:1905.10615*, 2019.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Jiang, M., Grefenstette, E., and Rocktäschel, T. Prioritized level replay. In *International Conference on Machine Learning*, pp. 4940–4950. PMLR, 2021.

Laskin, M., Lee, K., Stooke, A., Pinto, L., Abbeel, P., and Srinivas, A. Reinforcement learning with augmented data. *arXiv preprint arXiv:2004.14990*, 2020.

Lee, J. and Raginsky, M. Minimax statistical learning with wasserstein distances. *arXiv preprint arXiv:1705.07815*, 2017.

Lee, K., Lee, K., Shin, J., and Lee, H. Network randomization: A simple technique for generalization in deep reinforcement learning. *arXiv preprint arXiv:1910.05396*, 2019.

Lin, Y.-C., Hong, Z.-W., Liao, Y.-H., Shih, M.-L., Liu, M.-Y., and Sun, M. Tactics of adversarial attack on deep reinforcement learning agents. *arXiv preprint arXiv:1703.06748*, 2017.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Raileanu, R. and Fergus, R. Decoupling value and policy for generalization in reinforcement learning. *arXiv preprint arXiv:2102.10330*, 2021.

Raileanu, R., Goldstein, M., Yarats, D., Kostrikov, I., and Fergus, R. Automatic data augmentation for generalization in reinforcement learning. *arXiv preprint arXiv:2006.12862*, 2020.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

Sinha, A., Namkoong, H., Volpi, R., and Duchi, J. Certifying some distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571*, 2017.

Song, X., Jiang, Y., Tu, S., Du, Y., and Neyshabur, B. Observational overfitting in reinforcement learning. *arXiv preprint arXiv:1912.02975*, 2019.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30. IEEE, 2017.

Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhnevets, A. S., Yeo, M., Makhzani, A., Küttler, H., Agapiou, J., Schrittwieser, J., et al. Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782*, 2017.

Volpi, R., Namkoong, H., Sener, O., Duchi, J., Murino, V., and Savarese, S. Generalizing to unseen domains via adversarial data augmentation. *arXiv preprint arXiv:1805.12018*, 2018.

Wang, K., Kang, B., Shao, J., and Feng, J. Improving generalization in reinforcement learning with mixture regularization. *arXiv preprint arXiv:2010.10814*, 2020.

Yuan, X., He, P., Zhu, Q., and Li, X. Adversarial examples: Attacks and defenses for deep learning. *IEEE transactions on neural networks and learning systems*, 30(9): 2805–2824, 2019.

Zhang, A., Ballas, N., and Pineau, J. A dissection of overfitting and generalization in continuous reinforcement learning. *arXiv preprint arXiv:1806.07937*, 2018a.

Zhang, C., Vinyals, O., Munos, R., and Bengio, S. A study on overfitting in deep reinforcement learning. *arXiv preprint arXiv:1804.06893*, 2018b.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Zhao, Y., Shumailov, I., Cui, H., Gao, X., Mullins, R., and Anderson, R. Blackbox attacks on reinforcement learning agents using approximated temporal information. In *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, pp. 16–24. IEEE, 2020.