GENERALIZATION IN TRANSFER LEARNING

S. Ece Ada, Emre Ugur, H. Levent Akin
Department of Computer Engineering
Bogazici University
Istanbul,Turkey
{ece.ada, emre.ugur, akin}@boun.edu.tr

March 2, 2021

ABSTRACT

Agents trained with deep reinforcement learning algorithms are capable of performing highly complex tasks including locomotion in continuous environments. We investigate transferring the learning acquired in one task to a set of previously unseen tasks. Generalization and overfitting in deep reinforcement learning are not commonly addressed in current transfer learning research. Conducting a comparative analysis without an intermediate regularization step results in underperforming benchmarks and inaccurate algorithm comparisons due to rudimentary assessments. In this study, we propose regularization techniques in deep reinforcement learning for continuous control through the application of sample elimination, early stopping and maximum entropy regularized adversarial learning. First, the importance of the inclusion of training iteration number to the hyperparameters in deep transfer reinforcement learning will be discussed. Because source task performance is not indicative of the generalization capacity of the algorithm, we start by acknowledging the training iteration number as a hyperparameter. In line with this, we introduce an additional step of resorting to earlier snapshots of policy parameters to prevent overfitting to the source task. Then, to generate robust policies, we discard the samples that lead to overfitting via a method we call strict clipping. Furthermore, we increase the generalization capacity in widely used transfer learning benchmarks by using maximum entropy regularization, different critic methods, and curriculum learning in an adversarial setup. Subsequently, we propose maximum entropy adversarial reinforcement learning to increase the domain randomization. Finally, we evaluate the robustness of these methods on simulated robots in target environments where the morphology of the robot, gravity, and tangential friction coefficient of the environment are altered.

1 Introduction

Transfer learning refers to means of utilizing knowledge gained in one or multiple tasks for unseen novel tasks [1]. In transfer learning, the source task is the origin of knowledge whereas the target tasks are the test tasks that are not seen during training. A robot is expected to be robust to an unbounded number of tasks in real life. For instance, a bipedal robot will be required to be robust to different ground frictions, gravity, inclinations, and abrupt winds. Training a robot for all possible scenarios is time-consuming, impractical, and expensive. To develop intelligent systems, we expect the robots to generalize the learned model to the novel target task different from the source task, adequately and quickly.

Increasing generalization between tasks via Deep Reinforcement Learning (RL) is a useful strategy to acquire a robust robotic skill set. A common way of assessing generalization is evaluating a deep RL algorithm in a different task. Yet, deep RL algorithms, including the policy gradient methods [2, 3, 4], were commonly evaluated by the training performance because test and training task distinction do not exist. After successful results were obtained with policy gradient algorithms like Trust Region Policy Optimization (TRPO) [2], Proximal Policy Optimization (PPO) [3] and Soft Actor-Critic (SAC) [4], many efforts have been made to solve the problem of transferring the information from a given task to a different task in RL [5, 6, 7].

Quantifying generalization in state-of-the-art deep RL is challenging because the performance of deep RL algorithms is highly dependent on the hyperparameter and framework selection. This nature of deep RL algorithms affects the evaluation of transfer RL algorithms because transfer RL algorithms use deep RL algorithms as baselines. Henderson et al. [8] showed that different hyperparameters yield different results in non-transfer RL tasks. Statistically significant results can be obtained just by tuning the hyperparameters of the baseline algorithms. Yet, some hyperparameters (such as the clipping hyperparameter of PPO that controls the lower bound on the policy gradient loss) were set to fixed values while evaluating the results in many studies [9, 10, 11, 12, 13]. Similarly in the transfer RL domain, the trade-off between the target and the source task performance is oftentimes neglected when determining the hyperparameters. For instance, the best policy in the source task is not necessarily the best and the most robust policy in the target task.

One way of increasing the robustness of the policy is regularization. Hence, we propose regularization techniques that increase target task performance. We experimentally prove that the failure to acknowledge overfitting as well as diverse dimensionality of the robots induce inaccurate comparisons. On account of this, we first show that source task performance is not indicative of generalization capacity and target task performance. Without regularization of the baselines and the proposed methods, we can not distinguish improvements because the baselines will underperform. In this paper, we propose a method namely *Strict Clipping PPO* (SC-PPO) to discard the samples that cause overfitting by increasing the lower bound on the policy gradient loss. We apply SC by decreasing the clipping parameter in PPO to extremely small values and prove its impact numerically on a high dimensional Humanoid robot and a Hopper robot in the MuJoCo environment [14]. This regularization technique independently increases the performance of the policy in the target task and constitutes a competitive benchmark. We evaluated our method in Humanoid and Hopper benchmarks proposed in [15] and [7]. Subsequently, we verified our methods in both novel target tasks that involve transfer to robots with different morphologies and target tasks with an increased range of environmental dynamics. These novel tasks include tall, short, and delivery humanoid target tasks, each having a different center of mass and morphologies than the standard humanoid source task.

In transfer RL problems where the environmental dynamics of the target task diverges more from the source task, we applied early stopping. We have observed that the policy trained in the source task started to overfit the target task after several policy iterations. In this regard, we attest that earlier iterations of policies perform well in comparative analysis because stopping the training earlier avoids overfitting to the source task. We additionally show that the recognition of the policy iteration number as a hyperparameter increases the performance of state-of-the-art algorithms like PPO and Robust Adversarial Reinforcement Learning (RARL) [7].

Regularization techniques can also be used in combination. We perform comparative analysis with RARL to show the generalization capacity of using the combination of SC-PPO and early stopping regularization techniques. We compare RARL algorithm with our methods on the torso mass [7] and gravity [15] benchmarks where the hopper is expected to hop without falling. Table 1 shows the torso mass target tasks generated in the range [1,9] by modifying the mass of the torso and gravity target tasks generated in the range [0.5G,1.75G] by modifying the gravity of the environment. We use the same source task for training where the hopper torso mass is approximately 3.53 units and the gravity is $G = 9.81$. Early stopping is used in all of our methods as regularization. Our methods can successfully extrapolate to the target tasks in the corresponding range exceeding RARL in all benchmarks.

We compare the generalization capacities of different adversarial training methods in the target tasks, namely, entropy bonuses, advantage estimation techniques, and different curriculum learning [16] with RARL. We integrate an entropy bonus in adversarial RL in Maximum Entropy Robust Adversarial Reinforcement Learning (ME-RARL) to increase domain randomization. Our results show that entropy regularized adversarial RL significantly increases generalization in the target tasks. Furthermore, we increase generalization using the advantage estimation component by involving both value function estimator critic networks at each optimization iteration in Section 3.2.3 named Average Consecutive Critic Robust Adversarial Reinforcement Learning (ACC-RARL). Finally, we provide numerous experimental results to prove that our propositions outperform the existing methods significantly and increase the range of the commonly used benchmarks. We hope these techniques will aid in developing competitive benchmarks for future transfer RL research.

Table 1: Target task extrapolation success range

Task	**SC-PPO (ours)**	**ACC-RARL (ours)**	**ME-RARL(ours)**
Hopper-v2 Torso Mass	[1,6]	[1,7]	[1,9]
Hopper-v2 Gravity	[1G,1.5G]	[0.5G,1.75G]	[0.5G,1.75G]

In Section II, studies on the transfer RL and adversarial learning are provided. The proposed methods along with the background will be detailed in Section III. Experimental setup and results will be given in Section IV and V respectively.

Finally, in Section VI, the conclusion section will include a summary of our contributions, discussion of the results, and future directions for research.

2 Related Work

2.1 Transfer Reinforcement Learning

Transfer learning methods are categorized into three branches [17]. Transferring the knowledge and representation from one task to another task is called *forward transfer learning* [17]. In a *multi-task transfer learning* setting, the transferred knowledge, and the representation are based on multiple different tasks [18]. Muti-task RL is evaluated by the performance of the agent in multiple target environments. *Meta-learning* on the other hand focuses on reusing the learning experience [5]. Although the settings in multi-task transfer and meta-learning are the same, their main difference resides in the transferred representation. Most transfer learning problems are subject to the risk of negative transfer. Negative transfer occurs when the algorithm performs worse with transfer than not using any transfer at all [19]. In this case, initializing the algorithm without any information yields better performance than starting from a suboptimal point that leads to insufficient exploration. Most applications in transfer RL include the transfer of the policy parameters to the new task while avoiding negative transfer. The purpose of these applications is to increase the robustness in the target task via the manipulation of the training phase.

One way of extending the capabilities of existing robotic skills in transfer RL is by increasing only the generalization capacity of the RL component. Because the source and target tasks are identical in deep RL, algorithms' robustness to target tasks generated with domain shift is oftentimes overlooked [20]. Zhao *et al.* [21] studied generalization by parametrizing domain shift where the target task parameters are selected from a different distribution than the source task. In their work, the domain shift was realized by using systematic shifts of the environmental parameters, and noise scales injected to the transition, observation, actuator functions of the RL task.

Regularization is challenging in transfer RL because the performance on the source task is insufficient in determining the performance on the target task. Using unregularized deep RL algorithms results in inaccurate comparative analysis. In this respect, Cobbe *et al.* [22] developed a benchmark for generalization named CoinRun where a high number of tests and training levels could be generated. The generalization capacities of various agents were compared using the ratio of successfully solved target task levels by each agent. Few-shot learning is learning from a few labeled target task data. In zero-shot learning, however, no labeled target task data is provided during test time. Zero-shot performance can be improved using domain randomization [18] where task parameters are randomized when forming the training data. Dropout, L2 regularization, data augmentation, batch normalization, and increasing the stochasticity of the policy and the environment were the regularization techniques used in [22] to increase the zero-shot performance. Still, few shot-learning in the target task was required to tune the hyperparameters for generalization. Similar to [22], Zhao *et al.* [21] compared regularization techniques for deep RL such as policy network size reduction, soft actor-critic entropy regularizer [4], multi-domain learning [18], structured control net [23] and adversarially robust policy learning (APRL) [24].

2.2 Adversarial Learning

Adversarial algorithms are dynamic ways of generating challenging tasks for the agent at each iteration to increase robustness in unseen target tasks [12]. Adversarial learning is among the widely used techniques to increase generalization capacity in robotics [25, 26]. In the work of [25, 26], adversarial techniques are successfully utilized using two networks that learn together and advance by competing with each other. More specifically, the discriminator network is optimized to discriminate the real-world image data from the simulation whereas the generator network is optimized to generate simulator images that can fool the discriminator. Similarly in RARL[7], a separate adversarial network is optimized to destabilize the agent during training.

Inspired by the efficacious strategy in RARL [7], Shioya *et al.* [16] proposed two extensions to RARL by varying the adversary policies. First method they proposed was to add a penalty term to the adversary policy's reward function by sampling from the target task to adapt to the transition function of the source task. This is, however, tailored robustness for each target task at hand which requires sampling from the target task similar to the Bayesian update used in Ensemble Policy Optimization (EPOpt) [27]. The second extension was inspired by curriculum learning [28] that selects the adversarial agents based on the progress of learning instead of naively taking the latest adversarial policy. Training the protagonist in harder tasks does not guarantee a more robust policy. The level of difficulty required for high performance on the target task is highly dependent on the transfer problem. Bansal *et al.* [11] used uniform distribution to determine opponent humanoid policy's iteration from the subset of the latest iterations. In both Shioya *et al.*'s [16] and Bansal *et al.*'s [11] experiments, using the policy of the latest and the hardest adversary hindered the learning

progress of the agent. In [16], they used multiple adversaries and ranked each sample's performance to determine the set of samples that should be used for optimization. Each adversary policy was maximized using the negative reward of the agent and the sum of KL Divergence from all the other adversary policies to encourage diversity between the adversaries. Experiments were done using Hopper and Walker2d Open AI gym robot control environments[29] in MuJoCo [14] to compare the results to the RARL Algorithm [16]. In both of the environments, using the less rewarding rollouts performed worst. As a contrast, optimizing over the worst performing samples generated more robust policies in Hopper task when tested with different torso masses in EPOpt [27].

Adversarial algorithms are generating considerable interest in RL [7, 11]. Our method improves on previous work by employing regularization techniques to succeed in target tasks with higher domain shift. In the comparative analysis, we show that compared to RARL and PPO, the agent trained using our methods can successfully transfer skills to a wider range of unseen target tasks.

3 Method

In this paper, we propose a framework to obtain generalizable policies. Section 3.2.1 describes the Policy Buffer that is used to observe, and store the policies to select the generalizable ones for target task transfer. Section 3.2.2 provides the Strict Clipping component that prevents policy updates when the training samples improve the source task objective above a threshold. A low threshold stabilizes training and regularizes the source task objective. Regularization is applied using different methods in an adversarial framework in Section 3.2.3. More specifically, the agent is encouraged to explore the environment, different deep RL architectures are applied, and incremental learning is used to decrease overfitting. Before providing the details of our components, we will explain the background, namely the adversarial and deep reinforcement algorithms in the next section.

3.1 Background

We build our method on top of RARL algorithm which is characterized as a two-player zero sum discounted game [30, 31, 32]. The return of the agent is formalized as follows.

$$R^1 = E_{s_0 \sim \rho, a^1 \sim \mu(s), a^2 \sim \nu(s)} \left[\sum_{t=0}^{T-1} r^1 \left(s_t, a_t^1, a_t^2 \right) \right] \tag{1}$$

Actions a^1 are sampled from the policy μ of the agent and actions a^2 are sampled from the policy ν of the adversary. s_0 is the initial state sampled from the initial state distribution ρ. s_t is the state at timestep t and r corresponds to the reward function. The agent maximizes its return whereas the adversary minimizes the return of the agent. Thus, the return of the adversary is $R^2 = -R^1$.

We use PPO to update actor-critic networks of the agent and the adversary in RARL experiments as in [33]. Actor network determines the actions the agent takes whereas the critic network estimates the value function used for the advantage function estimation [34]. Advantage function estimates how rewarding it is to take the action in the state compared to the value of that state. We use a particular loss function, namely Clipped PPO Loss, L^{CLIP} [3] that optimizes the parameters θ of the actor policy network as follows.

$$L^{CLIP}(\theta) \quad = \hat{\mathbb{E}}_t \left[\min \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t, \text{clip} \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\varepsilon, 1+\varepsilon \right) \hat{A}_t \right) \right] \tag{2}$$

Generalized Advantage Estimator (GAE) [34] $\hat{A}_t$ is used to optimize the actor network. If the advantage $\hat{A}_t$ is negative then the probability ratios below $1-\varepsilon$ are clipped and if the advantage is positive, then the probability ratios above $1+\varepsilon$ are clipped. Gradient flow does not occur and the samples are discarded if the probability ratio $\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is clipped and the expression $\left(\text{clip} \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}, 1-\varepsilon, 1+\varepsilon \right) \hat{A}_t \right)$ is minimum. $\pi_\theta\left(a_t \mid s_t\right)$ is the current policy being optimized whereas $\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)$ is the previous policy. ε is the clipping parameter that controls the lower bound on the Clipped PPO Loss.

3.2 Proposed Method

3.2.1 Policy Buffer

Policies trained with different hyperparameters show different control patterns. We propose a policy buffer to store these policies that are trained with the same loss function but with different hyperparameters. We add the iteration number to the hyperparameter set for transfer RL benchmarks. In simulated experiments, we show that it is possible to extract a comprehensive set of policies capable of exhibiting various skills from a single source task.

Our method is inspired by the early stopping regularization technique that is frequently used in supervised learning. Similar to supervised learning, in the transfer RL setting, snapshots of policy parameters taken at different steps have different performances in the target task. We train policies with different hyperparameters and take snapshots of the corresponding policies at each optimization iteration at predetermined intervals and save them in the policy buffer. Previous works trained single policies for a constant number of iterations in the source task. In contrast, we evaluate multiple policies from the buffer to determine the iteration number where the model starts to overfit.

Our aim here is to generate successful policies for a range of target tasks that are represented by task parameters such as gravity, friction, mass, and center of mass of the robot. Overfitting is predominant in cases where the target tasks deviate significantly from the source task. Inspired by the validation dataset idea used in early stopping regularization for supervised learning, we propose designing a *proxy validation tasks set*. Given source and target task parameters and the policies learned in the source task, we form *proxy validation tasks* between source and target tasks with parameters closer to the target task. Expected rewards of the policies trained with different hyperparameters are informative in finding the generalizable policies from the buffer.

3.2.2 Strict Clipping

To train the aforementioned policies, we use policy gradient algorithms. In literature, hyperparameter tuning of the RL component is commonly overlooked and a fixed set of hyperparameters is used when integrating the RL component into the transfer domain. More specifically, *Open AI Baselines* framework and most of the literature has been using the clip parameter of 0.2 for continuous control tasks [35, 3, 11, 12, 13, 8].

Gradient clipping in Equation 2 is generally used to discourage catastrophic displacement in the parameter space. In our work, we additionally propose that it can be used for regularization in transfer RL. In particular, we propose a new regularization technique for PPO, namely *Strict Clipping* (SC) to avoid overfitting by constraining the gradient updates. During training, SC-PPO allows more source task samples to be discarded which would otherwise lead to overfitting. SC-PPO is used to further decrease the gradient movement in the policy parameter space in favor of generalization. This is achieved by decreasing the clipping parameter used in PPO by one or less order of magnitude. We prove that this method is superior to the unregularized RARL baseline in multiple transfer learning benchmarks.

3.2.3 Regularization in Adversarial Reinforcement Learning

Introducing an adversary to destabilize the agent using multidimensional simulated forces have generated successful results in continuous control tasks in RARL [7]. However, we experimentally prove that unregularized RARL performs worse than regularized PPO. In contrast to prior work, we acknowledge policy iteration as a hyperparameter in all our comparisons. Thus, we compare our methods in an adversarial framework by forming a policy buffer and extracting the most generalizable policies to increase the robustness of the algorithm. More specifically, we regularized critic networks, increased exploration, and integrated curriculum learning in an adversarial RL framework.

Average Consecutive Critic Robust Adversarial Reinforcement Learning (ACC-RARL) (I) Similar to actor networks, critic networks are function approximators that are prone to overfitting. We propose Average Consecutive Critic Robust Adversarial Reinforcement Learning (ACC-RARL), which computes advantage estimates using the mean of the critic outputs. Critics are optimized with their actor pair consecutively. We aim to decrease overfitting by using double critic networks with different random initializations and reuse the previous model by including the output of the previously updated critic in the advantage estimation. The temporal difference residual of the approximate value function with discount γ at timestep t corresponds to δ_t. The temporal difference residuals of the adversary and the protagonist in ACC-RARL are shown as follows.

$$\begin{aligned} \delta_t^{protagonist} &= (-V_{protagonist}(s_t) + V_{adversary}(s_t))/2 + r_t + \gamma(V_{protagonist}(s_{t+1}) - V_{adversary}(s_{t+1}))/2 \\ \delta_t^{adversary} &= -\delta_t^{protagonist} \end{aligned} \tag{3}$$

Maximum Entropy Robust Adversarial Reinforcement Learning (II) Entropy bonus is used for exploration by rewarding the variance in the distribution of the action probabilities. Following the entropy bonus idea in [36, 22], we incorporate an entropy bonus $H[\pi^{pro}]$ for the protagonist and an entropy bonus $H[\pi^{adv}]$ for the adversary in the reward functions. Updated reward functions of the adversary $R_t(\pi_\theta^{adv})$ and the protagonist $R_t(\pi_\theta^{pro})$ are given

$$\begin{aligned} R_t(\pi_\theta^{adv}) &= \hat{\mathbb{E}}_t[R_t^{actor}(\pi_\theta^{adv}) - R_t^{critic}(\pi_\theta^{adv}) + \beta_{adv} H[\pi^{adv}]\theta(s_t)] \\ R_t(\pi_\theta^{pro}) &= \hat{\mathbb{E}}_t[R_t^{pro}(\pi_\theta^{pro}) - R_t^{critic}(\pi_\theta^{pro}) + \beta_{pro} H[\pi^{pro}]\theta(s_t)] \end{aligned} \tag{4}$$

where π_θ^{adv}, π_θ^{pro} and β correspond to the policy of the adversary, policy of the protagonist, and the entropy coefficient respectively. Increasing exploration at the cost of decreasing source task performance increases the generalization capacity of the algorithm. Our hypothesis is motivated by the increase in robustness in competitive and adversarial environments [11, 16]. However, we acknowledge that adjusting the stochasticity of the environment while avoiding detrimental impact on learning is challenging. Adding an entropy bonus to the loss function of the adversary increases the domain randomization, affecting the performance of both the protagonist and the adversary. We compare the ME-RARL algorithms: entropy regularized RARL (ERARL) and entropy regularized ACC-RARL (EACC-RARL) in more challenging hopper morphology and gravity benchmarks where SC-PPO is not sufficient.

Curriculum Learning (III) Curriculum Learning, focuses on discovering the optimal arrangement of the source tasks to perform better on the target task. Similar to [11, 16], we utilize curriculum learning by randomizing the adversary policy iterations from different sets of advancement. We compare the target task performance of protagonist policies trained with adversaries randomly chosen from the last predefined number of iterations. We use uniform sampling from the policy set of the adversary [16]. At each iteration, based on the sampling from the Uniform$(\chi v, v)$ distribution [11], the adversary from the policy storage is used. Adversaries loaded from and recorded to the buffer during curriculum training become less capable and more inconsistent as the training progress and χ decreases. In consequence, we intend to show how a variety of design decisions during training affects the target task performance in the pursuit of developing novel regularization techniques and more reliable benchmarks.

4 Experimental Setup

Robots are expected to generalize to the target tasks using experience from the source tasks. We use a set of transfer learning experiments in the *Humanoid-v2* and *Hopper-v2* MuJoCo simulation environments [14]. To demonstrate the generalization capability of our methods, we create target tasks through the modification of the environment dynamics [7]. We use hopper environments to compare our methods to RARL [7] and humanoid environments to verify the transfer to tasks where the morphology as well as the functionality of the robot is altered. We use *OpenAI Baselines* framework [35] in all our experiments.

The humanoid target tasks are generated to show the performance of the policies trained with SC-PPO and early stopping. Rajeswaran *et al.* altered ground friction to create a target task for a hopper robot [27]. Similarly, we transfer the learning experience from a source task with ground tangential friction coefficient of 1 to a target task with ground tangential friction coefficient of 3.5. Transferring among morphologically different robots with different limb sizes or torso masses have been a popular multi-task learning benchmark [15, 27, 7]. Accordingly, we generate three novel target environments: a taller and heavier humanoid robot, a shorter and lighter humanoid robot and a delivery robot that carries a heavy box.

Hopper environments are used to compare adversarial methods with RARL. Similar to the multi-task transfer learning experiments in [15], we generate target tasks by altering the torso mass of the robot and the gravity of the environment. In [37], 4 target tasks were created by modifying the gravity parameters of the environment in the range [0.50G, 1.50G] where G=-9.81. In our experiments, we use a larger range [0.50G, 1.75G] to discover the range of tasks our method can solve. Similarly for the torso mass experiments, we create target tasks by significantly increasing the range of the torso mass of the robot to [1.0, 9.0] from the range [2.75, 4.5] used in RARL.

5 Results

After detailing source task training implementations in Sections 5.1 and 5.2, we provide results of our methods SC-PPO in Section 5.3 and ACC-RARL, maximum entropy RARL in Section 5.4 using various target tasks. The hyperparameters used in source task training are given in Table A. We report on the expected mean and standard deviation of the rewards obtained from 32 randomly seeded identical environments for each target task.

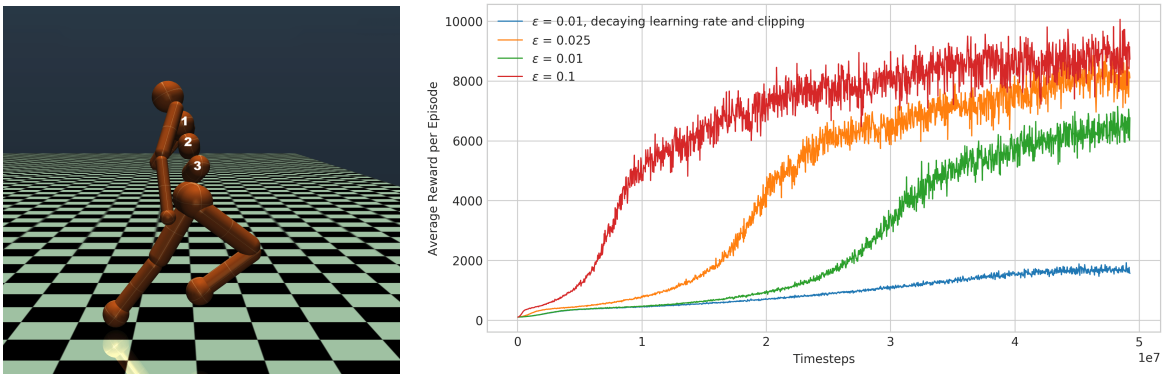
5.1 Humanoid Source Environment

The reward function of the *Humanoid-v2* environment in [29] consists of a linear forward velocity reward, a quadratic impact cost with lower bound 10, a quadratic control cost, and an alive bonus C_{bonus}.

$$r_{humanoid}(s,a) = 0.25 * r_{v_{fwd}} + min(5 \cdot 10^{-7} * c_{impact}(s,a), 10) + 0.1 * c_{control}(s,a) + C_{bonus} \tag{5}$$

If the *z-coordinate* of the center of mass of the agent is not in the interval [1,2], the episode terminates. The total loss function of the PPO algorithm used to update actor and critic networks is

$$L_t^{CLIP+VF}(\theta) = \hat{E}_t \left[L_t^{CLIP}(\theta) - c_1 L_t^{VF}(\theta) \right] \tag{6}$$



(a) (b)

Figure 1: (a) Humanoid running in the source environment. (b) Learning curves of SC-PPO and PPO on standard humanoid source task.

The average reward per episode and standard deviations of the policies trained with 4 different sets of hyperparameters are shown in Figure 1(b). Learning curve obtained with the latest PPO hyperparameters suggested in *OpenAI Baselines* framework [35] for the Humanoid environment are represented by the red curve in Figure 1(b). The learning curves for using strict clipping in the source tasks are shown with clipping hyperparameters 0.01, 0.025, and 0.01 decaying learning rate and clipping.

5.2 Hopper Source Environment

The Reward function of the hopper task consists of linear forward velocity reward, sum of squared actions and an alive bonus C_{bonus}.

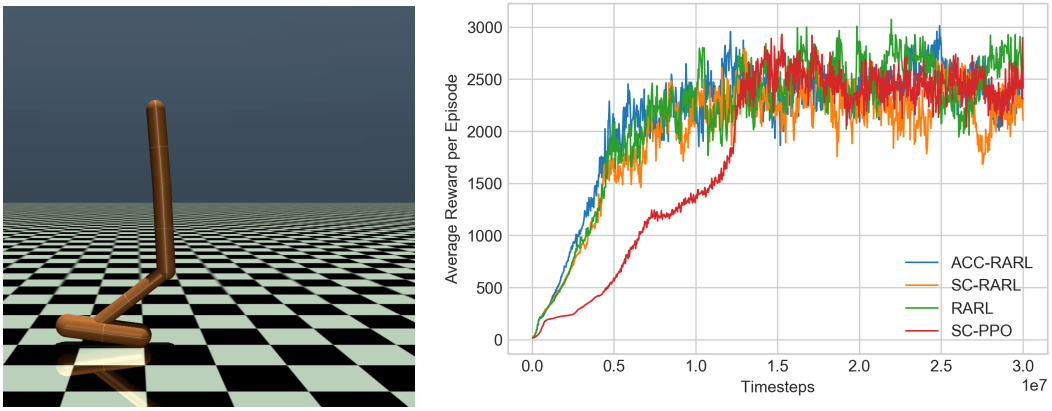
$$r_{hopper}(s,a) = \ r_{v_{fwd}} - 0.001 * \sum a^2 + C_{bonus} \tag{7}$$

SC-PPO was trained using $\alpha = 0.0003$, $\epsilon = 0.05$, $b = 2048$, and the adversarial algorithms were trained with $\alpha = 0.0003$, $\epsilon = 0.3$, $b = 512$. Source task learning curves of PPO and RARL trained with different critic architectures are shown in Figure 2(b). Results in hopper tasks are consistent with the humanoid experiments. To analyze the effect of different architectures on the generalization capacity we compare three different critic architectures: separate double critic networks used in RARL [30], single critic network in Shared Critic Robust Adversarial Reinforcement Learning (SC-RARL) [33] and ACC-RARL. The policies trained with different variations of RARL perform similar to SC-PPO in hopper morphology tasks in the range of 1-6 and hopper gravity tasks in the range of 1G-1.5G. Next, we compare performances of these policies in various target tasks.

5.3 Regularization via PPO Hyperparameter Tuning

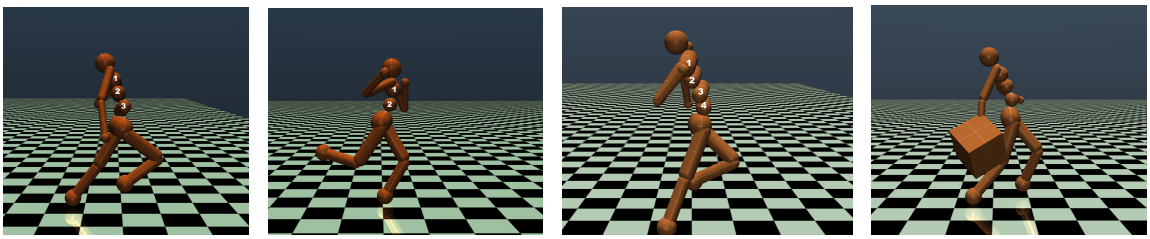
5.3.1 The Morphology Experiments

The morphological target tasks are created for inter-robot transfer learning. The short, tall, and delivery humanoid target tasks in Figures 3(b), (c), (d) are generated from the standard humanoid source task in Figure 3(a).



(a) (b)

Figure 2: (a) Hopping action, and (b) learning curves of ACC-RARL, SC-RARL, RARL, SC-PPO on standard hopper source task.



(a) (b) (c) (d)

Figure 3: (a) Standard humanoid source task with 3 torso components, (b) short humanoid target task with 2 torso components, (c) tall humanoid target task with 4 torso components, and (d) delivery humanoid target task

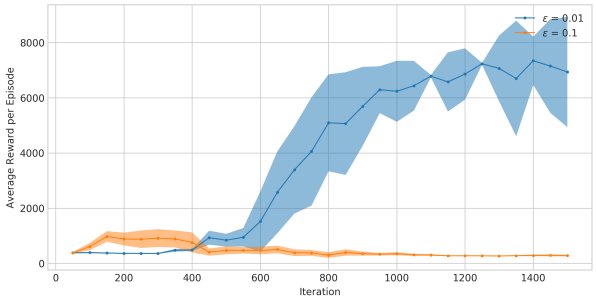
The total body weights of the short and tall humanoid target tasks differ from the humanoid source task by the exclusion and inclusion of the upper waist respectively. The tall humanoid task is more challenging than the short humanoid task because it is harder to balance heavier upper body mass where the center of mass is higher from the ground. The termination criterion for bipedal locomotion in humanoids depends on the location of the center of the torso. When the +z location of the center of mass of the robot is below a threshold, the robot is presumed to fall and the episode is terminated. This threshold is higher for the tall humanoid and lower for the short humanoid because of the different locations of the center of mass.

Table 5.2: Delivery Humanoid Environment

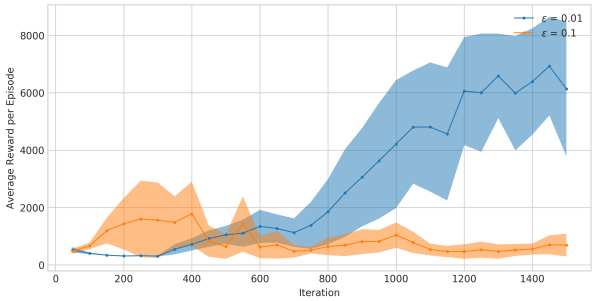
Body	**Unit Mass**
Delivery Box	5
Right Hand	1.2
Torso	8.3
Total Body without Delivery Box	40.0

Masses of the relevant body parts for the delivery humanoid are given in 5.3.1. Considering the total body mass of 40, a box with a unit mass of 5 constitutes a challenging benchmark. Our purpose is to create a horizontal imbalance by enforcing the humanoid to carry the box only by the right hand.

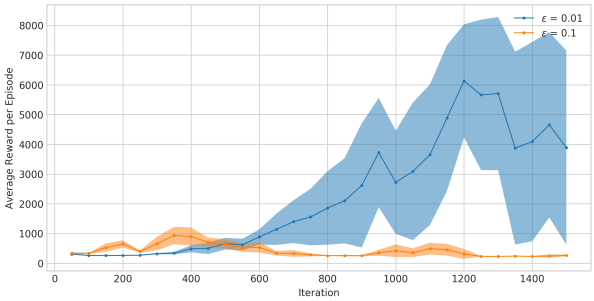
The comparison of the target task performances of policies trained with clipping parameters $\epsilon = 0.1$ and $\epsilon = 0.01$ are provided in Figures 4(a), 4(b) and 4(c). High average reward per episode obtained after 1200^{th} iteration with SC-PPO $\epsilon = 0.01$ show that humanoid learns transferable general characteristics of forward locomotion from the source task. In



(a)



(b)



(c)

Figure 4: Performance of SC-PPO and PPO on (a) shorter and lighter humanoid target task, (b) taller and heavier humanoid target task, and (c) delivery humanoid target task.

contrast, all iterations of the policies trained with clipping parameter $\epsilon = 0.1$ failed in all target tasks. Early stopping regularization technique alone is not sufficient because the short and tall humanoids can not stand still using the earlier iterations of the policy trained with clipping parameter $\epsilon = 0.1$.

The reduction in the performance after the 1200^{th} iteration in Figure 4(c), supports the method of resorting to the earlier policy iterations. This concavity suggests that overfitting occurs and early stopping is an effective regularization technique.

The same policy iteration (1200^{th}) trained with SC-PPO can be successfully transferred to short, tall and delivery humanoid tasks as shown in Figures 4(a), 4(b) and 4(c). Evaluation of the best source task policies in the target tasks assesses our claim that source task performance is not indicative of the generalization capacity. Thus, all transfer RL methods should account for regularization first.

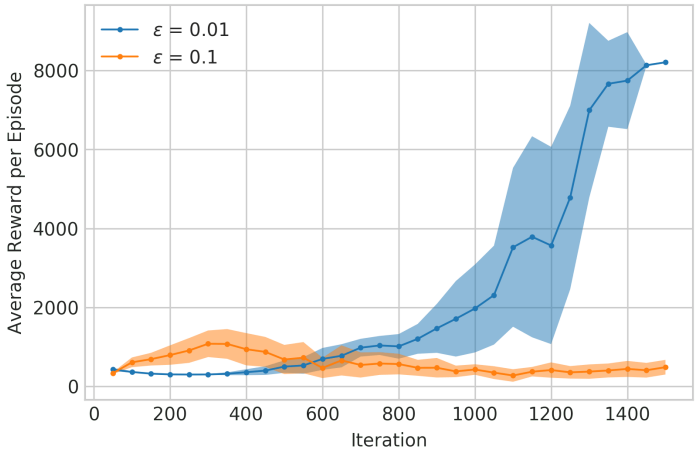
5.3.2 The Friction Environment

Our aim in this experiment is to evaluate our methods in transfer learning benchmarks where the environment dynamics are changed. We generate a target task by increasing the ground friction coefficient in MuJoCo environment. The humanoid sinks due to high tangential friction but can still run following the regularized policies trained with SC-PPO.

Table 5.3: Comparison of SC-PPO and PPO in Target Friction Environment

Clip	Iteration	Average Reward per Episode
0.01	1500	8283 ± 24.3
0.1	300	1078 ± 336.4

The best jumpstart performances for each clipping parameter are given in Figure 5. For instance, the last iteration of the policy with a SC-PPO parameter $\epsilon = 0.01$ has an average target task reward of 8283 with 24.3 standard deviation as provided in Table 5.3. In contrast, the best performing policy in the source task has a target task performance which assesses our claim that source task performance on transfer RL is not indicative of the target task performance. These results show that agent trained using our methods learns generalizable skills for environments with changing dynamics.

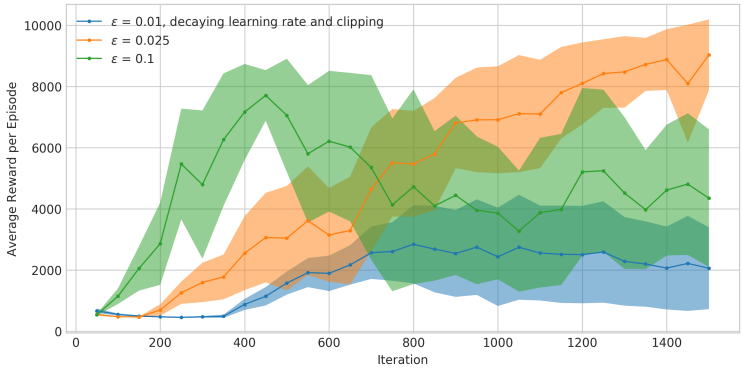


(a)

Figure 5: Performance of SC-PPO and PPO on target environment with tangential friction 3.5 times the source environment.

5.3.3 The Gravity Environments

Our aim in these experiments is to evaluate our methods in a different set of target tasks where environment parameters are changed. Similar to the friction experiments, we generate gravity target tasks in the range of 0.5G-1.75G where G=-9.81 is the gravity of the source task.

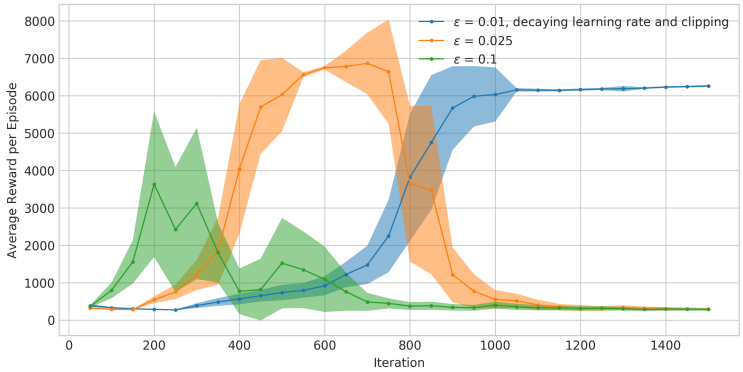


(a)

Figure 6: Performance of SC-PPO and PPO on target environment with gravity= -4.905 ($0.5G_{Earth}$).

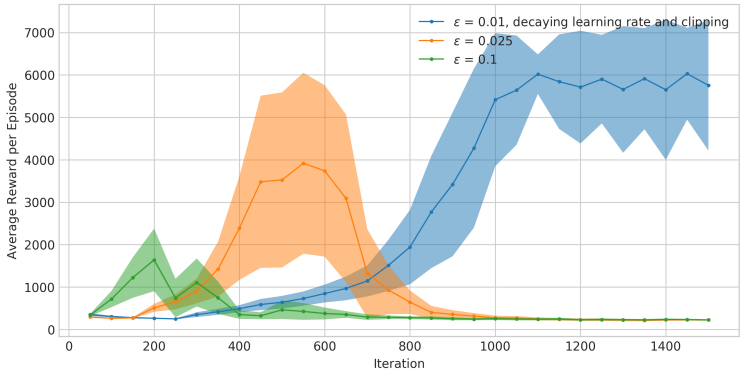
When the last iteration of the policy trained with SC-PPO $\epsilon = 0.025$ is used in the target task with gravity= -4.905 ($0.5G_{Earth}$) the humanoid in Figure 6(a) can run. Performance of SC-PPO is slightly better than early stopping for this target task. Similarly, in the target task with gravity= -14.715 ($1.5G_{Earth}$), the humanoid needs to resort to the previous snapshots of the policy trained with SC-PPO $\epsilon = 0.025$ as plotted in the Figure 7(a). The bipedal locomotion pattern in the simulated target environment with gravity= -14.715 ($1.5G_{Earth}$) when the humanoid jumpstarts with the 600^{th} policy trained with clipping parameter $\epsilon = 0.025$ is shown in Figure 7(b).

Gravity benchmarks for the humanoid indicate that snapshots of different policies should be used for the target task with gravity= - 17.1675 ($1.75G_{Earth}$). The policy iterations trained with hyperparameters "$\epsilon = 0.01$ *decaying learning rate and clipping*" performed poorly in the source task and target task with lower gravity. However, they perform consistently well in environments with a higher magnitude of gravity (- 17.1675,- 14.715). Figures 7 and 8 show that decaying the clipping parameter and the learning rate during training decreased the exploration and restricted the humanoid to stick to a more careful way of moving forward after 1000 training iterations.



(a)

Figure 7: Performance of SC-PPO and PPO on target environment with gravity= -14.715 ($0.5G_{Earth}$).



(a)

Figure 8: Performance of SC-PPO and PPO on target environment with gravity= -17.1675 ($1.75G_{Earth}$).

5.4 Regularization in Adversarial Reinforcement Learning

5.4.1 The Morphology Experiments

In this set of experiments, we compare our methods with RARL. In RARL, target tasks are generated by modifying the torso mass of the robot in the range of 2.5-4.75 [7]. Following the same procedure, we modify the torso mass in the range of 1-9. Table 5.4 provides additional information on the morphology of the Hopper source task.

Table 5.4: Hopper Source Environment

Body	Unit Mass
Torso	3.5
Thigh	3.9
Leg	2.7
Foot	5.1
Total Body	15.3

We evaluate the target task performances of RARL [30], Shared Critic Robust Adversarial Reinforcement Learning (SC-RARL) [33], and ACC-RARL. The only difference between SC-RARL and RARL is that in SC-RARL, a shared critic network is used for value function approximation. No regularization is used in [33], thus its generalization capacity is similar to RARL. In our work, we first regularize all adversarial architectures, then compare their generalization capacity.

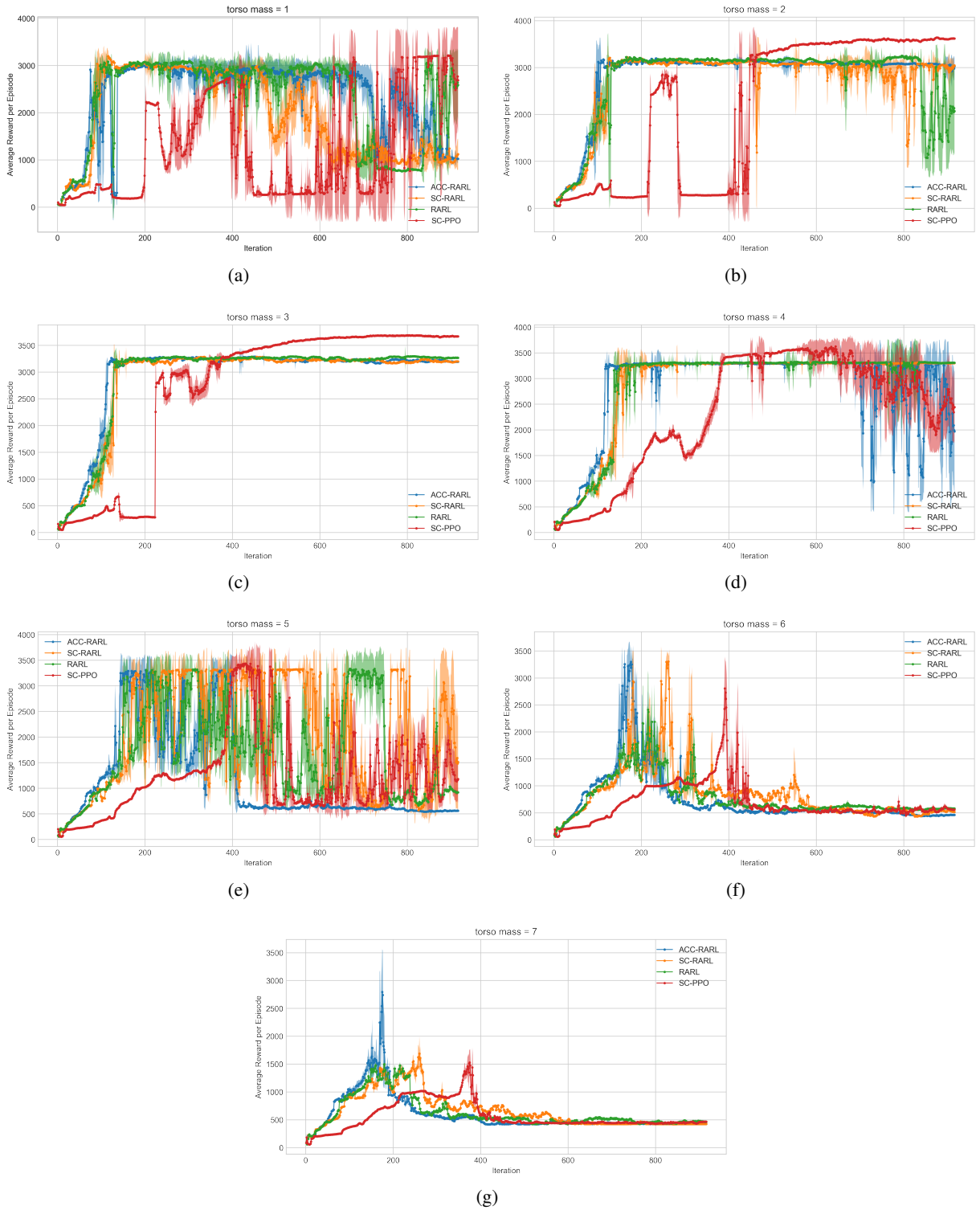
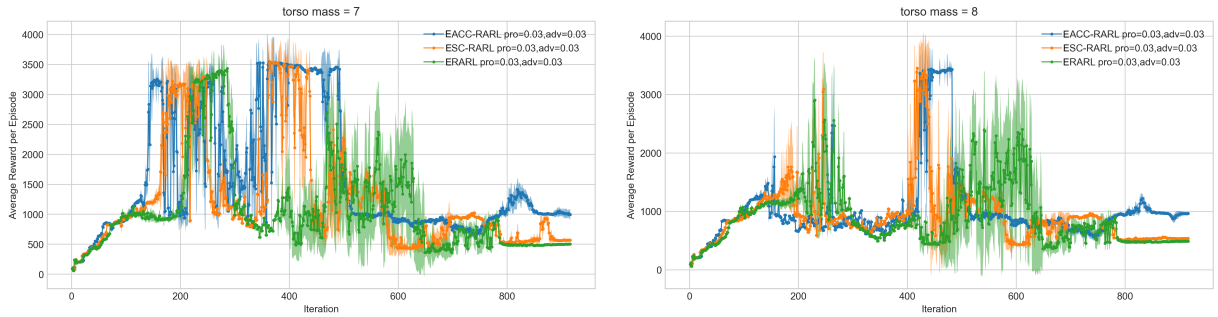


Figure 9: Performance of SC-PPO and adversarial methods on Hopper target tasks with torso mass (a)1, (b)2, (c)3, (d)4, (e)5, (f)6, (g)7

The robot can successfully generalize using any regularized adversarial technique or SC-PPO with early stopping when the target tasks are in the range of 1-6. Based on the target task performances of ACC-RARL, SC-RARL, and RARL, in Figure 9, different critic value function approximation techniques have an effect on the type of control behavior learned. However, the performance of some policy iterations trained with RARL, ACC-RARL, SC-RARL starts to become unstable in Figures 9(a) and 9(e). In harder target tasks like these, the agent should first resort to earlier snapshots of the policy and use early stopping for adversarial RL. Let us assume that the agent only has several snapshots of the policy in its buffer trained in the source task with standard torso mass. Then, the agent is put in a target task with torso mass=6 which is analogous to an agent expected to carry weight while performing a control task. We propose that in cases like these, instead of training from the very beginning, the agent should primarily resort to earlier policy iterations because the generalizable policies performing at the expert level are readily available in the agent's memory and were extracted from the source task training.

If we had not recognized the policy iteration as a hyperparameter then comparing the algorithms at arbitrarily selected number of training iterations would not constitute a fair comparison. More importantly, the performance of PPO without adversaries, generally used as a benchmark algorithm, is highly dependent on the number training iterations. Hence for target tasks with torso masses $[1-6]$, the right snapshots of SC-PPO are capable of obtaining high average reward per episode as seen in Figures 9(a), 9(b), 9(c), 9(d), 9(e), 9(f). Furthermore, the skills learned with RARL in the last 150 iterations is successful in the source task and target task where torso mass is 3 and 4 units as provided in Figure 9(c). However, they are clearly unstable in tasks where torso mass is 1,2,5,6,7 as illustrated in Figures 9(a). The policy buffer allows us to do a fair comparison among different methods. Training the algorithms for an arbitrary number of iterations will produce inaccurate results such as underperforming baselines or proposed algorithms. Furthermore, the reproducibility of the transfer RL algorithms will be affected.

Regarding the parametric form of target tasks, the performance of the policies residing in the policy buffer can be anticipated. Coinciding with the performances seen in Figures 9(a), 9(f), 9(g), 11(b), it is anticipated that the earlier policy iterations trained with less samples perform better as the distance between the target task and the source task increases in the task parameter space. We reproduced the original RARL experiment in [7] where results are consistent with the performance of the last policy iteration trained with RARL (Figures 9(b), 9(c), 9(d)).



(a) (b)

Figure 10: Performance of EACC-RARL, ESC-RARL and ERARL on target tasks with torso mass (a)7 and (b)8

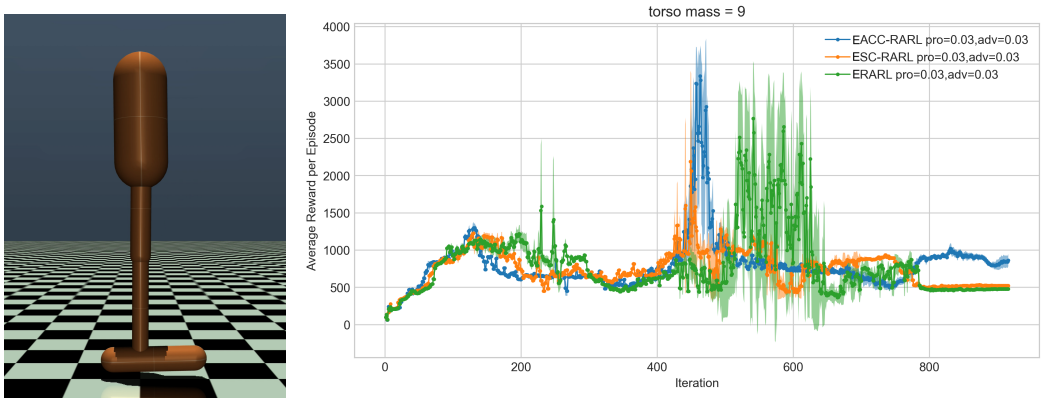
When the torso mass is increased to 6, significant target performance drop (Figure 9(f)) occurs until the last policy iteration where all policies are affected. This implies overfitting occurs and all policy iterations trained with SC-PPO can still perform optimally for policy iterations around 395. The best performing policy iterations of adversarial algorithms start to get accumulated in the range $[150, 300]$ when torso mass is greater than 5 thus a mapping between the target task parameters and the policy iterations is highly probable.

In a harder environment with a torso mass of 7, the earlier iterations of the policy trained with ACC-RARL performs the best. Figure 9(g) shows that the range of the best-performing policy iterations is contracted more.

Using ME-RARL algorithms (EACC-RARL, ESC-RARL, and ERARL) increased the fluctuation of the average rewards for all algorithms as provided in Figure 10. Because entropy bonus encourages exploration, the adversary takes more randomized actions which often change the protagonist's hopping behaviour. None of the adversarial techniques were successful in completing target tasks without entropy regularization where the torso mass of the hopper is increased to 8 and 9 units. EACC-RARL shows the highest performance on the 9 unit torso mass target task (Figure 11(b)). The average reward per episode of the policy iterations trained with EACC-RARL is provided in Table 5.6. These results

Table 5.5: Performance of ACC-RARL

Unit Mass	Iteration	Average Reward per Episode
1	175	2921 ± 12
2		3072 ± 6
3		3196 ± 6
4		3253 ± 4
5		3279 ± 3
6		3259 ± 221
7		2792 ± 768



(a) (b)

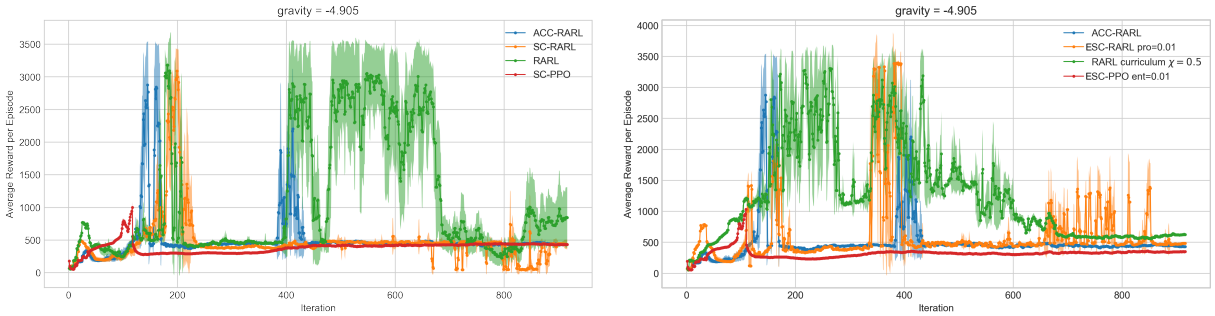
Figure 11: (a) Hopper that has torso mass of 9 units,and (b) Performance of EACC-RARL, ESC-RARL and ERARL on target task with torso mass 9.

show that earlier policy iterations show better performance as the target task becomes more challenging. Any target task from the set can be used as a proxy validation task for the tasks that belong to the same set.

Table 5.6: Performance of EACC-RARL

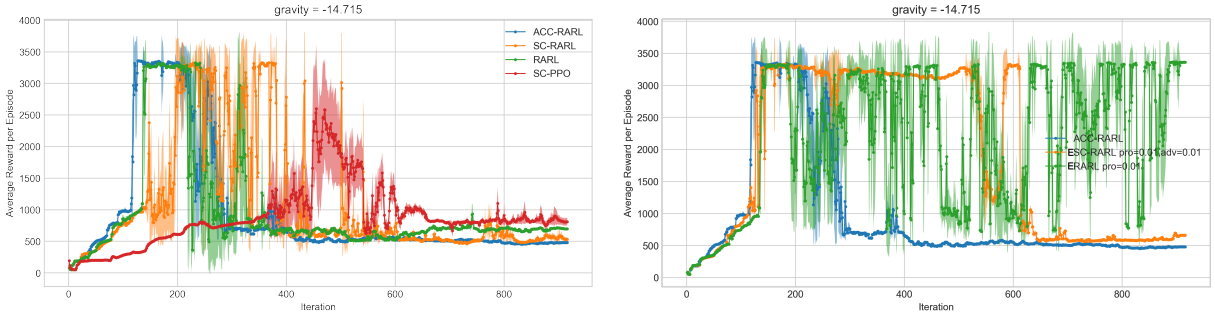
Set	Unit Mass	Iteration	Average Reward per Episode
1	1	508	2872 ± 36
	2		3377 ± 501
	3		3196 ± 13
	4		3474 ± 7
2	5	479	2704 ± 764
	6		2764 ± 901
	7		3425 ± 3
3	8	463	3423 ± 5
	9		3283 ± 500

First, using SC-PPO and early stopping we formulate a more competitive baseline for the adversarial algorithms. Then, we show the performance of one policy iteration (175^{th}) with high generalization capacity trained with the ACC-RARL algorithm in Table 5.5 to demonstrate that a policy is capable of performing forward locomotion when torso mass is in the range of 1-7. Finally, using EACC-RARL and early stopping we have increased the target task success range used in RARL[7] from $[2.5, 4.75]$ to $[1, 9]$.



(a) (b)

Figure 12: Performance of (a)SC-PPO, ACC-RARL, SC-RARL and RARL, and (b)ESC-PPO, ACC-RARL, ESC-RARL and RARL with curriculum on target environment with gravity= -4.905 ($0.5G_{Earth}$)



(a) (b)

Figure 13: Performance of (a)SC-PPO, ACC-RARL, SC-RARL and RARL, and (b) ACC-RARL, ESC-RARL and ERARL on target environment with gravity= -14.715 ($1.5G_{Earth}$)

5.4.2 The Gravity Environments

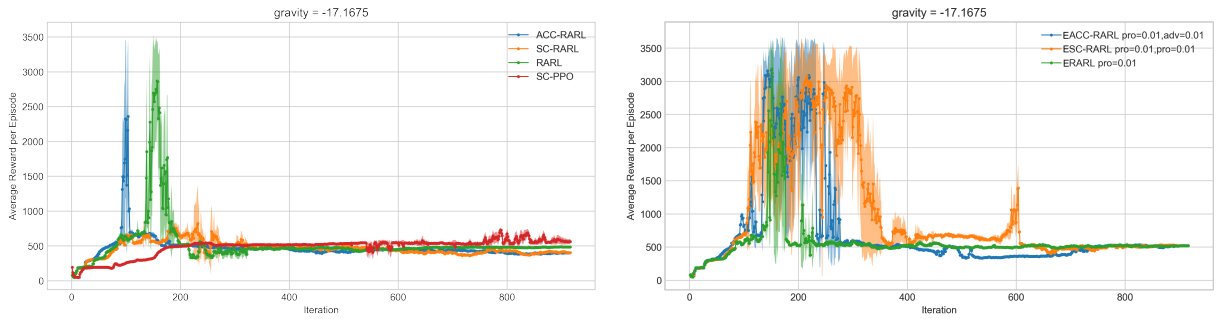
In this set of experiments, we will assess our methods in a large range $[0.5G_{Earth}, 1.75G_{Earth}]$ of gravity tasks. In Learning Joint Reward Policy Options using Generative Adversarial Inverse Reinforcement Learning (OptionGAN) [37], the gravity of the generated environments is in the range of $0.5G_{Earth} - 1.5G_{Earth}$ for both humanoid and hopper. The policy over options converges to 2 different policies for Hopper tasks in [37]: one for lower and one for higher than the gravity of Earth indicating that the tasks are complex enough to be solved with different policies. For these tasks, we will use the same policy buffer created using SC-PPO and different variations of RARL for the hopper morphology experiments.

The performance evaluations in Figures 12(a) and 12(b) prove that training with curriculum and maximum entropy changes the number of generalizable policy iterations in $0.5G_{Earth}$ target task. ESC-PPO included in Figure 12(b) still performs poorly compared to adversarial techniques. Although the improvement is negligible, only for this case the entropy regularized SC-PPO (ESC-PPO) performs better than the SC-PPO.

Policies trained with SC-PPO and adversarial methods can be transferred to the target task with gravity= -14.715 in Figure 13(a). Figure 13b shows that using entropy regularized methods ESC-RARL and ERARL, not only increased the average reward per episode but also increased the number of generalizable policy iterations.

As the target task gets harder by moving further away from the source task, the best performing policy iterations are aggregated around earlier iterations similar to the torso mass and the delivery humanoid target tasks. This concavity is analogous to the convexity of the test error curve in supervised learning problems where earlier training iterations lead to underfitting and the later iterations overfit to the training set. The regularization effect of early stopping is pivotal in increasing the generalization capacity. The domain randomization in SC-RARL does not suffice for generalization in harder target tasks when gravity is $1.75G_{Earth}$ (Figure 14(a)). We observe that encouraging the exploration of the protagonist policy and the adversary through the inclusion of entropy bonus increases the performance of adversarial

algorithms in Figure 14(b). Results in the gravity environments are in line with the morphology experiments. Above all, entropy regularized adversarial techniques, early stopping and strict clipping produce competitive results in hard target tasks.



(a) (b)

Figure 14: Performance of (a)SC-PPO, ACC-RARL, SC-RARL and RARL, and (b) EACC-RARL, ESC-RARL and ERARL on target environment with gravity= -17.1675 ($1.75G_{Earth}$)

6 Conclusion

In this study, we propose a set of methods to extract generalizable knowledge from a single source task. Training independently for each task is a customary sample inefficient process in deep RL. With each environment interaction, the agent's strategy of solving the source task is expected to advance. However, we find that the transferability of these strategies to similar tasks relies significantly on the number of environment interactions in the source task. The best strategy to solve the source task fails substantially in the target task.

Our experiments show that the performance of transfer RL algorithms is dramatically dependant on the choice of hyperparameters and the number of policy iterations used in the training of the source task. This dependency affects the reproducibility and evaluation accuracy of the algorithms in transfer RL. We addressed this issue in 19 different transfer RL experiments to show how experimental results can be manipulated in favor of an algorithm in the transfer RL domain.

In our work, we proposed keeping a policy buffer to capture different skills because source task performance is not indicative of target task performance. Accordingly, transferring the policy with the best source task performance to the target task becomes a less adequate evaluation technique as the difference between the source and target task increases. In line with this, we suggest using a proxy validation task to extract the generalizable policies. To account for the early stopping regularization technique, we propose the inclusion of the policy iteration to the hyperparameter set. After this inclusion, we have retrieved the generalizable skills that generate high rewards in the target tasks.

We introduced SC-PPO and early stopping as a regularization technique in transfer RL. Training SC-PPO promote the elimination of samples that overfit the source task. We experimentally show that these robust policies generate a higher jumpstart performance than the baseline in the target tasks. Using SC-PPO and early stopping, we transferred the forward locomotion skills of a standard humanoid to humanoids with different morphologies (a taller humanoid, a shorter humanoid, and a delivery humanoid), humanoids in environments with different gravities (0.5G, 1.5G, and 1.75G) and a humanoid in an environment with 3.5 times the ground friction coefficient of the source task. Moreover, our results show that we can increase the extrapolation range of Hopper morphology tasks from the range of [2.5,4.75] used in RARL to [1,6] via SC-PPO and early stopping. We applied the same technique to transfer the skills of a hopper to the target task with gravity 1.5G.

We conducted a comparative analysis with RARL with different critic methods, entropy bonus, and curriculum learning. Our results show that the choice of hyperparameters and training iteration affect the generalization capacity substantially in adversarial RL. Using entropy regularized ACC-RARL and early stopping via policy buffer, we have increased the extrapolation range of Hopper torso mass tasks from the range of [2.5,4.75] used in RARL to [1,9]. We prove that a hopper is capable of performing forward locomotion in a target task in the range of [0.5G,1.75G] by learning in the source task with regularized adversarial RL. Using entropy regularized adversarial RL, significantly increased the performance on target tasks and the number of generalizable policies extracted from the source task.

We believe that the first step of determining the most promising policy parameters lies in the accurate parameterization of the source and target task space. In the future, we plan to investigate the relationship between the environment parameters and the source task training hyperparameters.

Acknowledgement

This work was supported by the Scientific and Technological Research Council of Turkey (TUBITAK, 118E923).

References

[1] Alessandro Lazaric. Transfer in reinforcement learning: a framework and a survey. In *Reinforcement Learning*, pages 143–173. Springer, 2012.

[2] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

[3] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[4] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

[5] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

[6] Kevin Frans, Jonathan Ho, Xi Chen, Pieter Abbeel, and John Schulman. Meta learning shared hierarchies. *arXiv preprint arXiv:1710.09767*, 2017.

[7] Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2817–2826. JMLR. org, 2017.

[8] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[9] Yasuhiro Fujita and Shin-ichi Maeda. Clipped action policy gradient. *arXiv preprint arXiv:1802.07564*, 2018.

[10] Seungyul Han and Youngchul Sung. Dimension-wise importance sampling weight clipping for sample-efficient reinforcement learning. In *International Conference on Machine Learning*, pages 2586–2595, 2019.

[11] Trapit Bansal, Jakub Pachocki, Szymon Sidor, Ilya Sutskever, and Igor Mordatch. Emergent complexity via multi-agent competition. *arXiv preprint arXiv:1710.03748*, 2017.

[12] Maruan Al-Shedivat, Trapit Bansal, Yuri Burda, Ilya Sutskever, Igor Mordatch, and Pieter Abbeel. Continuous adaptation via meta-learning in nonstationary and competitive environments. *arXiv preprint arXiv:1710.03641*, 2017.

[13] Ye Huang, Chaochen Gu, Kaijie Wu, and Xinping Guan. Reinforcement learning policy with proportional-integral control. In *International Conference on Neural Information Processing*, pages 253–264. Springer, 2018.

[14] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5026–5033. IEEE, 2012.

[15] Peter Henderson, Wei-Di Chang, Florian Shkurti, Johanna Hansen, David Meger, and Gregory Dudek. Benchmark environments for multitask learning in continuous domains. *arXiv preprint arXiv:1708.04352*, 2017.

[16] Hiroaki Shioya, Yusuke Iwasawa, and Yutaka Matsuo. Extending robust adversarial reinforcement learning considering adaptation and diversity. In *International Conference on Learning Representations*, 2018.

[17] S. Levine. Lecture 15-Transfer and Multi-Task Learning. http://rail.eecs.berkeley.edu/deeprlcourse-fa17, 2017.

[18] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 23–30. IEEE, 2017.

[19] Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime Carbonell. Characterizing and avoiding negative transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11293–11302, 2019.

[20] Chiyuan Zhang, Oriol Vinyals, Remi Munos, and Samy Bengio. A study on overfitting in deep reinforcement learning. *arXiv preprint arXiv:1804.06893*, 2018.

[21] Chenyang Zhao, Olivier Siguad, Freek Stulp, and Timothy M Hospedales. Investigating generalisation in continuous deep reinforcement learning. *arXiv preprint arXiv:1902.07015*, 2019.

[22] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. *arXiv preprint arXiv:1812.02341*, 2018.

[23] Mario Srouji, Jian Zhang, and Ruslan Salakhutdinov. Structured control nets for deep reinforcement learning. In *International Conference on Machine Learning*, pages 4749–4758, 2018.

[24] Ajay Mandlekar, Yuke Zhu, Animesh Garg, Li Fei-Fei, and Silvio Savarese. Adversarially robust policy learning: Active construction of physically-plausible perturbations. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3932–3939. IEEE, 2017.

[25] Konstantinos Bousmalis, Alex Irpan, Paul Wohlhart, et al. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4243–4250. IEEE, 2018.

[26] Eric Tzeng, Coline Devin, Judy Hoffman, et al. Adapting deep visuomotor representations with weak pairwise constraints. *arXiv preprint arXiv:1511.07111*, 2015.

[27] Aravind Rajeswaran, Sarvjeet Ghotra, Balaraman Ravindran, and Sergey Levine. Epopt: Learning robust neural network policies using model ensembles. *arXiv preprint arXiv:1610.01283*, 2016.

[28] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.

[29] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[30] Lerrel Pinto. *Rllab implementation for Robust Adversarial Reinforcement Learning*. 2017, https://github.com/lerrel/rllab-adv, accessed in December 2018.

[31] Lerrel Pinto. *Gym environments with adversarial disturbance agents*. 2017, https://github.com/lerrel/gym-adv, accessed in March 2019.

[32] Rocky Duan, Peter Chen, Houthooft, Rein, John Schulman, and Pieter Abbeel. *rllab*. 2016, https://github.com/rll/rllab, accessed in December 2018.

[33] Zisu Dong. *Tensorflow implementation for Robust Adversarial Reinforcement Learning*. 2018, https://github.com/Jekyll1021/RARL, accessed in March 2019.

[34] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[35] Prafulla Dhariwal, Christopher Hesse, Oleg Klimov, et al. *OpenAI Baselines*. 2017, https://github.com/openai/baselines, accessed in May 2018.

[36] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, et al. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[37] Peter Henderson, Wei-Di Chang, Pierre-Luc Bacon, David Meger, Joelle Pineau, and Doina Precup. Optiongan: Learning joint reward-policy options using generative adversarial inverse reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

A Hyperparameters

Table A.1: Source Task Training Hyperparameters

Hyperparameters	**Symbol**	**Values**					
Clipping Parameter	ε	0.01	0.025	0.05	0.1	0.2	0.3
Batch size	b	64			512		
Step Size	α	0.0001			0.0003		
Curriculum Parameter	χ	0.3			0.5		
Learning Schedule		constant			linear		
Clipping Schedule		constant			linear		
Entropy Coefficient	β_{pro}, β_{adv}	0.01			0.03		
Trajectory Size	H	2048					
Discount	γ	0.99					
GAE Parameter	λ	0.95					
Adam Optimizer	β_1	0.9					
Adam Optimizer	β_2	0.999					
Number of Epochs		10					
Number of Hidden Layers		2					
Hidden Layer Size		64					
Activation Function		tanh					