Robust Adversarial Reinforcement Learning

Lerrel Pinto James Davidson Rahul Sukthankar Abhinav Gupta

***Abstract*—Deep neural networks coupled with fast simulation and improved computation have led to recent successes in the field of reinforcement learning (RL). However, most current RL-based approaches fail to generalize since: (a) the gap between simulation and real world is so large that policy-learning approaches fail to transfer; (b) even if policy learning is done in real world, the data scarcity leads to failed generalization from training to test scenarios (e.g., due to different friction or object masses). Inspired from H_∞ control methods, we note that both modeling errors and differences in training and test scenarios can be viewed as extra forces/disturbances in the system. This paper proposes the idea of robust adversarial reinforcement learning (RARL), where we train an agent to operate in the presence of a destabilizing adversary that applies disturbance forces to the system. The jointly trained adversary is reinforced – that is, it learns an optimal destabilization policy. We formulate the policy learning as a zero-sum, minimax objective function. Extensive experiments in multiple environments (InvertedPendulum, HalfCheetah, Swimmer, Hopper, Walker2d and Ant) conclusively demonstrate that our method (a) improves training stability; (b) is robust to differences in training/test conditions; and c) outperform the baseline even in the absence of the adversary. Full paper can be accessed here: https://arxiv.org/abs/1703.02702**

INTRODUCTION

High-capacity function approximators such as deep neural networks have led to increased success in the field of reinforcement learning [6, 10, 4, 5, 7]. However, a major bottleneck for such policy-learning methods is their reliance on data: training high-capacity models requires huge amounts of training data/trajectories. While this training data can be easily obtained for tasks like games (e.g., Doom, Montezuma's Revenge) [6], data-collection and policy learning for real-world physical tasks are significantly more challenging.

There are two possible ways to perform policy learning for real-world physical tasks: **Real-world Policy Learning:** The first approach is to learn the agent's policy in the real-world. However, training in the real-world is too expensive, dangerous and time-intensive leading to scarcity of data. Due to scarcity of data, training is often restricted to a limited set of training scenarios, causing overfitting. If the test scenario is different (e.g., different friction coefficient), the learned policy fails to generalize. Therefore, we need a learned policy that is robust and generalizes well across a range of scenarios.

Learning in simulation: One way of escaping the data scarcity in the real-world is to transfer a policy learned in a simulator to the real world. However the environment and physics of the simulator are not exactly the same as the real world. This reality gap often results in unsuccessful transfer if the learned policy isn't robust to modeling errors [2, 9].

Both the test-generalization and simulation-transfer issues are further exacerbated by the fact that many policy-learning algorithms are stochastic in nature. For many hard physical tasks such as Walker2D [3], only a small fraction of runs leads to stable walking policies. This makes these approaches even more time and data-intensive. What we need is an approach that is significantly more stable/robust in learning policies across different runs and initializations while requiring less data during training.

So, how can we model uncertainties and learn a policy robust to all uncertainties? How can we model the gap between simulations and real-world? We begin with the insight that modeling errors can be viewed as extra forces/disturbances in the system [1]. For example, high friction at test time might be modeled as extra forces at contact points against the direction of motion. Inspired by this observation, this paper proposes the idea of modeling uncertainties via an adversarial agent that applies disturbance forces to the system. Moreover, the adversary is reinforced – that is, it learns an optimal policy to thwart the original agent's goal. Our proposed method, Robust Adversarial Reinforcement Learning (RARL), jointly trains a pair of agents, a protagonist and an adversary, where the protagonist learns to fulfil the original task goals while being robust to the disruptions generated by its adversary.

We perform extensive experiments to evaluate RARL on multiple OpenAI gym environments like InvertedPendulum, HalfCheetah, Swimmer, Hopper, Walker2d and Ant. We demonstrate that our proposed approach is: **(a) Robust to model initializations:** The learned policy performs better given different model parameter initializations and random seeds. This alleviates the data scarcity issue by reducing sensitivity of learning. **(b) Robust to modeling errors and uncertainties:** The learned policy generalizes significantly better to different test environment settings (e.g., with different mass and friction values).

A. Overview of RARL

Our goal is to learn a policy that is robust to modeling errors in simulation or mismatch between training and test scenarios. For example, we would like to learn policy for Walker2D that works not only on carpet (training scenario) but also generalizes to walking on ice (test scenario). Similarly, other parameters such as the mass of the walker might vary during training and test. One possibility is to list all such parameters (mass, friction etc.) and learn an ensemble of policies for different possible variations [8]. But explicit consideration of all possible parameters of how simulation and real world might

differ or what parameters can change between training/test is infeasible.

Our core idea is to model the differences during training and test scenarios via extra forces/disturbances in the system. Our hypothesis is that if we can learn a policy that is robust to all disturbances, then this policy will be robust to changes in training/test situations; and hence generalize well. But is it possible to sample trajectories under all possible disturbances? In unconstrained scenarios, the space of possible disturbances could be larger than the space of possible actions, which makes sampled trajectories even sparser in the joint space.

To overcome this problem, we advocate a two-pronged approach:

(a) Adversarial agents for modeling disturbances: Instead of sampling all possible disturbances, we jointly train a second agent (termed the adversary), whose goal is to impede the original agent (termed the protagonist) by applying destabilizing forces. The adversary is rewarded only for the failure of the protagonist. Therefore, the adversary learns to sample hard examples: disturbances which will make original agent fail; the protagonist learns a policy that is robust to any disturbances created by the adversary.

(b) Adversaries that incorporate domain knowledge: The naive way of developing an adversary would be to simply give it the same action space as the protagonist – like a driving student and driving instructor fighting for control of a dual-control car. However, our proposed approach is much richer and is not limited to symmetric action spaces – we can exploit domain knowledge to: focus the adversary on the protagonist's weak points; and since the adversary is in a simulated environment, we can give the adversary "superpowers" – the ability to affect the robot or environment in ways the protagonist cannot (e.g., suddenly change a physical parameter like frictional coefficient or mass).

References

[1] Tamer Başar and Pierre Bernhard. *H-infinity optimal control and related minimax design problems: a dynamic game approach*. Springer Science & Business Media, 2008.

[2] Paul Christiano, Zain Shah, Igor Mordatch, Jonas Schneider, Trevor Blackwell, Joshua Tobin, Pieter Abbeel, and Wojciech Zaremba. Transfer from simulation to real world through learning deep inverse dynamics model. *arXiv preprint arXiv:1610.03518*, 2016.

[3] Tom Erez, Yuval Tassa, and Emanuel Todorov. Infinite horizon model predictive control for nonlinear periodic tasks. *Manuscript under review*, 4, 2011.

[4] Shixiang Gu, Timothy Lillicrap, Ilya Sutskever, and Sergey Levine. Continuous deep Q-learning with model-based acceleration. *arXiv preprint arXiv:1603.00748*, 2016.

[5] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[6] Volodymyr Mnih et al. Human-level control through deep reinforcement learning. *Nature*, 2015.

[7] Igor Mordatch, Kendall Lowrey, Galen Andrew, Zoran Popovic, and Emanuel V Todorov. Interactive control of diverse complex characters with neural networks. In *Advances in Neural Information Processing Systems*, pages 3132–3140, 2015.

[8] Aravind Rajeswaran, Sarvjeet Ghotra, Sergey Levine, and Balaraman Ravindran. Epopt: Learning robust neural network policies using model ensembles. *ICLR*, 2017.

[9] Andrei A Rusu, Matej Vecerik, Thomas Rothörl, Nicolas Heess, Razvan Pascanu, and Raia Hadsell. Sim-to-real robot learning from pixels with progressive nets. *arXiv preprint arXiv:1610.04286*, 2016.

[10] David Silver et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 2016.