Adversarial Policy Training against Deep Reinforcement Learning

Xian Wu, Wenbo Guo, Hua Wei, Xinyu Xing
Presented by Ethan Rathbun

Brief Overview of Actor Critic

Actor produces a probability distribution over actions given the current state, $P(a_t \mid s_t;\Theta)$

Critic produces action value (Q) function with which to measure expected utility of actions

$$\Delta\theta = \alpha\nabla_\theta(log\pi_\theta(s,a))\hat{q}_w(s,a)$$

Change in policy parameters (weights) — Action value estimate

Actor Update

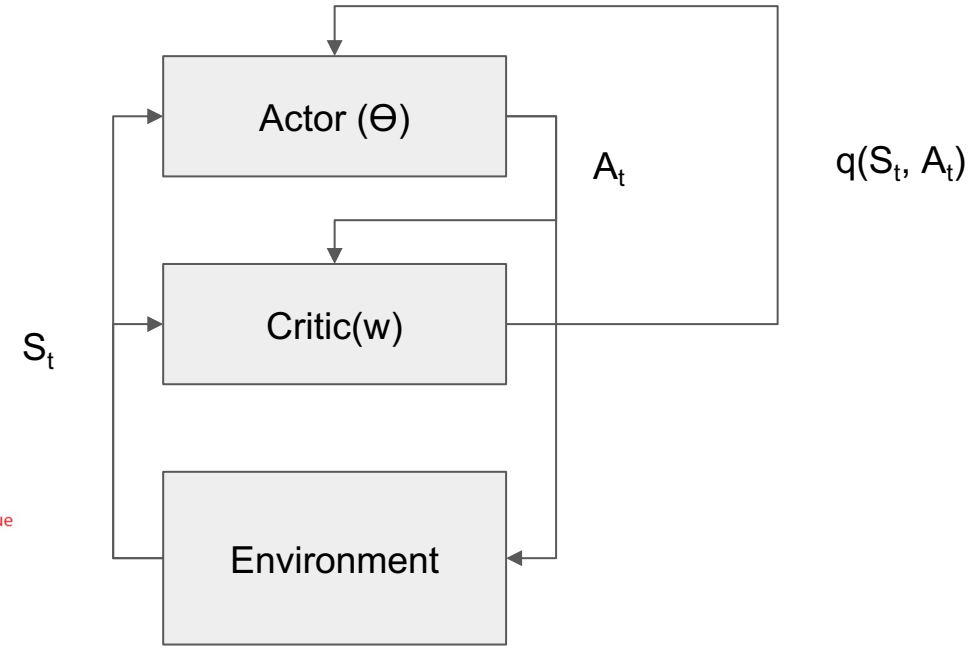
$$\Delta w = \beta\,(R(s,a) + \gamma\hat{q}_w(s_{t+1},a_{t+1}) - \hat{q}_w(s_t,a_t))\nabla_w\hat{q}_w(s_t,a_t)$$

Policy and value have different learning rates

TD error

Gradient of our value function

Critic Update (Q-learning)



Slides adapted from Thomas Simonini's course: "Deep Reinforcement Learning Class with Hugging Face"

Advantage

In practice, updating w.r.t. Q values produced by the critic cause convergence issues as many Q values tend to be similar in value at a given time step.

$$A(s,a) = Q(s,a) - V(s)$$

q value for action a in state s

average value of that state

$$L^{PG}(\theta) = E_t[\log \pi_\theta(a_t|s_t) * A_t]$$

log probability of taking that action at that state

Advantage if A>0, this action is better than the other action possible at that state

Slides adapted from Thomas Simonini's course: "Deep Reinforcement Learning Class with Hugging Face"

PPO

Step size may be too large with actor critic methods, therefore PPO clips the maximum step size within range [1-ε, 1+ε]

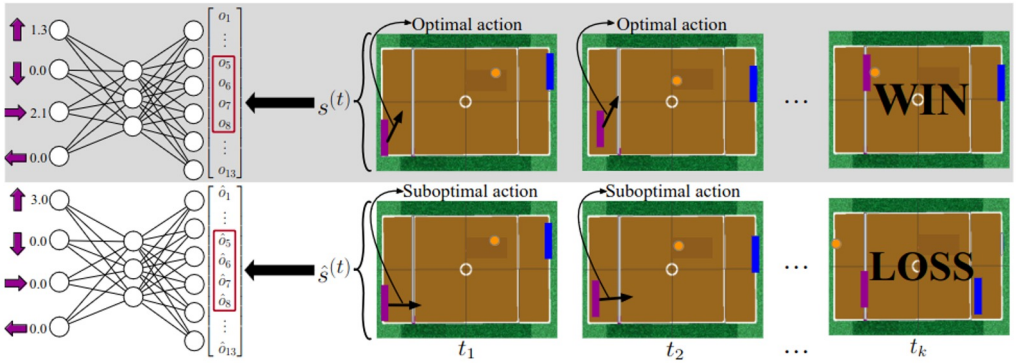
$$L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right]$$

$$r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}$$

Slides adapted from Thomas Simonini's course: "Deep Reinforcement Learning Class with Hugging Face"

Problem Statement

- Attacks which target state observations directly are unrealistic.
- Previously developed “Adversarial Policy” attacks (Gleave et al.) optimize opponent reward directly with PPO rather than trying to cause suboptimal actions in the victim.



Threat Model

- Victim policy is fixed and stochastic
  - Allows problem to be reduced to a single-agent MDP
- Black Box access to the victim
  - Allows adversary to train surrogate networks, and play against the victim

$$M = < \mathcal{S}, (\mathcal{A}_\alpha, \mathcal{A}_v), \mathcal{P}, (\mathcal{R}_\alpha, \mathcal{R}_v), \gamma >$$

Reduced to

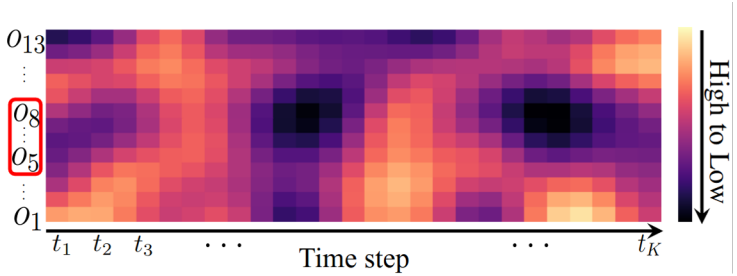
$$M_\alpha = < \mathcal{S}, \mathcal{A}_\alpha, \mathcal{P}_\alpha, \mathcal{R}_\alpha, \gamma >$$

Methods - High Level

1. Introduce an additional term into the loss to maximize victim action deviation while minimizing their observation

$$L_{ad} = \text{maximize}_{\theta}(-\|\hat{o}_v^{(t+1)} - o_v^{(t+1)}\| + \|\hat{a}_v^{(t+1)} - a_v^{(t+1)}\|)$$

2. Balance the vanilla PPO loss with this new, L_{ad} term according to the victim's attention to relevant observations
   a. In our case, relevant observations are those the adversary can control
   b. They use gradient based Ex-AI techniques, which require white-box access to some model



$$L_{ppo} + \lambda \cdot L_{ad}.$$

Methods - In practice

- Given we only have black-box access to the victim, we must approximate their behavior with surrogate models.
- Both models below are trained online along with the adversarial policy using supervised learning techniques.

$$\text{argmin}_{\theta_h} \| H(o_v^{(t)}, a_v^{(t)}, a_\alpha^{(t)}; \theta_h) - o_v^{(t+1)} \|_\infty$$

H approximates the next observation of the victim given a_v , a_a , and o_v

$$L_{op} = \text{minimize}_{\theta_f} \| (|F(o_v^{(t)}; \theta_f) - a_v^{(t)}| - \varepsilon_a)^+ \|_2^2$$

F approximates the victim's policy

Explainable AI Methods

They use gradient based methods to approximate the “importance” of each observation

$$I^{(t)} = \|F(o_v^{(t)}) - F(o_v^{(t)} \odot (\tilde{g}^{(t)} \odot M))\|_\infty, \ \lambda^{(t)} = \frac{1}{1 + I^{(t)}}$$

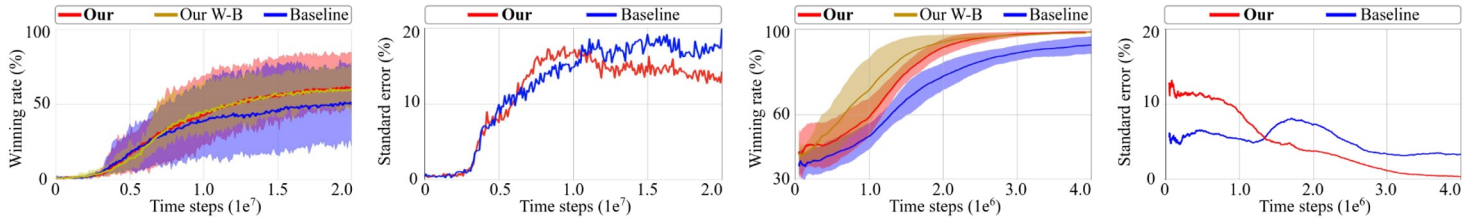
M is a vector of 0s and 1s, able by the adversary.

$$\tilde{g}^t = \sum_{j=1}^{q} \nabla_{o_v^t} F(o_v^t)_{ij}$$

i^{th} element of g represents importance of observation i according to F

Experiments

- Test on MuJoCo “you shall not pass” and RoboSchool Pong
  - Both environments have observations according to angles, positions, and velocities

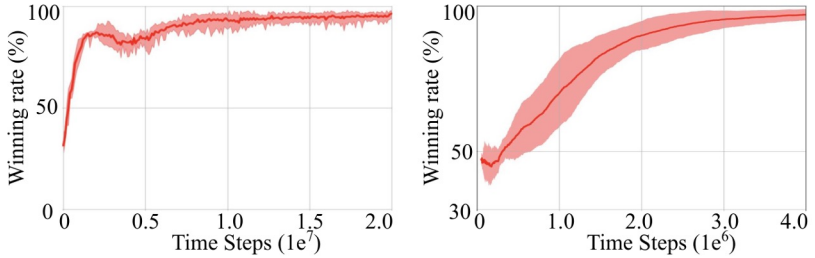


(a) MuJoCo winning rate. (b) MuJoCo standard error. (c) Roboshool winning rate. (d) Roboshool standard error.

Figure 7: Our attack vs. the baseline approach [10] in two different games. Note that “Our W-B” represents our attack in the white-box setting, where the approximated policy network of the victim agent was replaced with its actual policy network.

Experiments

They test what happens when one re-trains the victim model to perform well against a fixed adversarial policy, similar to Gleave et al.



(a) MuJoCo. (b) Roboschool Pong.

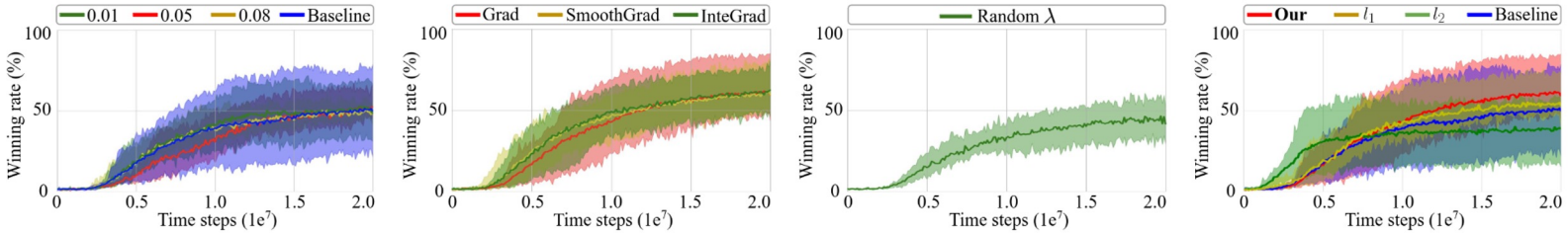
Figure 8: The winning rate of adversary-retrained victim agent against our adversarial agent in two different games.

Game	Min	Max	Mean	Std
MuJoCo	6.0%	25.0%	16.3%	6.2%
Roboschool Pong	40.0%	44.0%	41.4%	1.4%

Table 1: The winning rate of the adversary-retrained victim agent against the corresponding regular agent in two different games. Note that after retraining the victim agents, we test them for one hundred episodes.

Experiments

They provide some ablation with respect to the explainable AI methods used in the attack formulation.



(a) Constant λ. (b) Explainable AI techniques. (c) Random λ. (d) Distance measures.

Figure 6: The winning rates of our adversarial agent trained with different hyperparameter selection strategies and distance measures. The darker solid lines in the figures are the average winning rate of the corresponding agent. The lighter shadow represent the variation between the maximal and minimal winning rates.

Limitations

- Writing
  - Way too much writing, not enough math/results, hard to parse math integrated in text
  - Even the algorithm is loaded with words
- Experiments
  - Only test in two different environments. Could have chosen better ones to highlight strengths.
  - Testing environments are fairly simple and not diverse
  - Unable to replicate Gleave results
- Method
  - Optimizing w.r.t **4** different networks - likely has instability issues
  - No convergence guarantees

Strengths

- Writing
  - Provides an intro to RL
- Experiments
  - Show that the attack works
- Methods
  - New, more direct approach to generating adversarial policies
  - Intersection with explainable AI is interesting. The problem of determining when to “activate” an adversarial policy is interesting in general.

Discussion and Questions

- What are the best methods to balance adversarial policies and “optimal play”?
- Is it possible to transfer/generalize adversarial policies?
- What are the correct goals of adversarial policies?
  - What actions should we induce in the victim?
- What even are adversarial policies?
  - What separates an adversarial policy from a policy which is optimal against a fixed opponent?