Stealthy and Efficient Adversarial Attacks against Deep Reinforcement Learning (SEAADRL) – AAAI 2020

Jianwen Sun Tianwei Zhang
Xiaofei Xie Lei Ma Yan Zheng Kangjie Chen Yang Liu

Presented by Aayush Naik

Table of Contents

Table of Contents

Motivation

Analyzing attacks against DL & DRL important because:

- DL & DRL are becoming widely used.
- Especially in high-stakes and safety-critical scenarios: autonomous driving, weapon systems, medical assistance.

Image sources: `techofaq.org`, `wikipedia.org`, `medium.com`

Table of Contents

The RL Setting

The environment is modeled as a Markov Decision Process (MDP) [Bel57], given by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$.

- $\mathcal{S}$: a discrete/continuous space of states.
- $\mathcal{A}$: a discrete/continuous action space.
- $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$: transition function.
- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$: reward function.
- γ: discount factor.

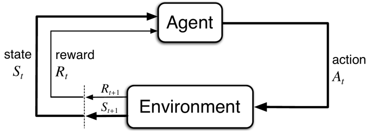


Figure: The RL Setting [SB18]

Attacks Against Supervised DL and DRL

Since the work of Szegedy et. al [SZS+13], several attacks have been demonstrated on supervised deep learning.

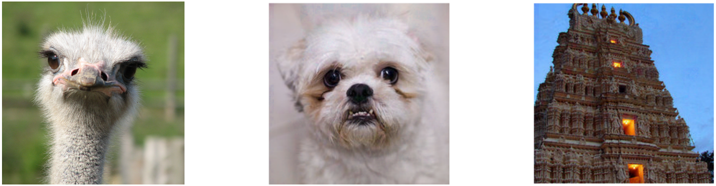


Figure: Ostrich (left), also ostrich (center), ostrich (right). Image credits: [SZS+13].

Attacks Against Supervised DL and DRL

Two types of attacks:

- Untargeted attacks: misclassification to an arbitrary class/label [SZS+13, MDFF16].
- Targeted attacks: misclassification to a *specific* class/label [CW17].

Moving to DRL (initial works):

- Apply attacks from supervised DL to inputs of DRL agents [HPG+17].

Key Ideas

1. Targeted attacks against supervised DL can be used to make DRL agents choose specific suboptimal actions [HGP19].
2. Going for a small number of *critical hits*, instead of attacking at every step. *Timing* attacks to deal the maximum “damage” in the least number of steps [KS17, SZX+20].

Table of Contents

SEADRL: Overview

Two components of a targeted DRL attack:

1. *How*: Generating noise added to the input (perturbation) to make the agent choose an action.
2. *When*: Deciding when to do perturbation to maximize "damage" in the least number of steps.

SEAADRL uses exisiting methods for the *how* [CW17], and its main contribution is in the *when*. Two algorithms/classes of attack:

1. Critical-point (CP) attack.
2. Antagonist attack.

SEAADRL: Critical-point Attack

- At every timestep t, consider all possible N-step action sequences.
- Compute the *divergence* of each of "final" states.
- Pretend that we have access to a transition function of the environment: $\hat{\mathcal{P}}$.

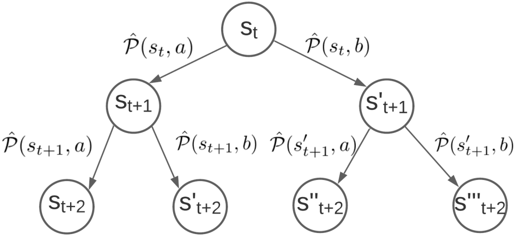


Figure: $\mathcal{A} = \{a, b\}$ and $N = 2$ in this example. $T(s_{t+2})$ is the divergence of state s_{t+2}.

- User-defined domain specific function.
- For TORCS,

$$T(s) := \text{distance between car and center of road.}$$

Figure: Low divergence; near center.

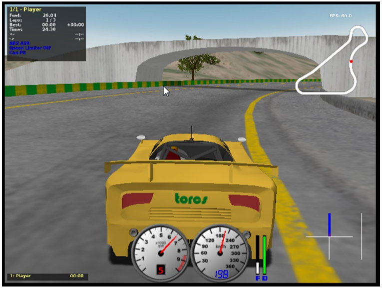


Figure: High divergence; off center.

Image credits: `malavida.com` and `apps4win.com`

SEAADRL: Critical-point Attack

- Let our trained DRL agent have policy π.
- Compare the divergence of the *N*-step action sequences with the *baseline*.
- Perform attack if $|T(s_{t+2}) - T(s'_{t+2})| > \Delta$. Here, $N = 2$.

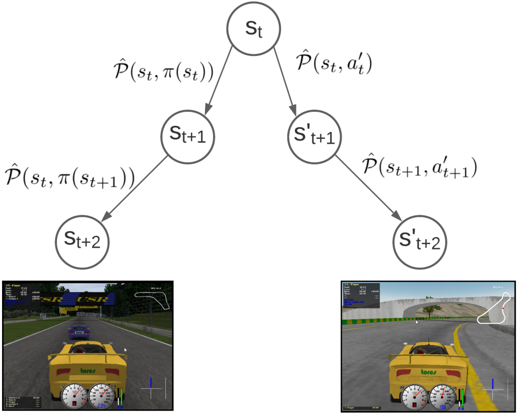


Figure: Baseline expected state (left); highest-divergence action sequence (right).

Image credits: `malavida.com` and `apps4win.com`

SEAADRL: Critical-point Attack: Summary

- Train environment transition model $\hat{\mathcal{P}}$.
- At every timestep t, look ahead N steps with all possible action sequences.
- If any of those sequences lead to high divergence, perform that attack.
- Attack only once per episode (efficiency).
- Big disadvantage: highly domain specific.

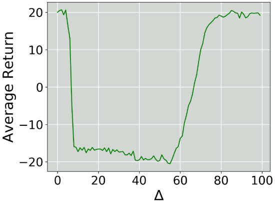
SEAADRL: Antagonist Attack

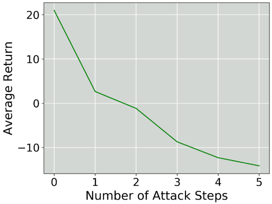
- Train an agent (called antagonist) that receives $-r_t$ at every time step.
- Allow this agent to decide whether to attack or not at each timestep (when).
  - Limit number of antagonist actions (attacks) to a small number of steps (2-5) per episode.
  - A regular, pretrained DRL agent acts for other steps.
- The antagonist learns to pick opportune moments to act and do significant damage.
- Main advantage over CP attack: requires no domain knowledge.

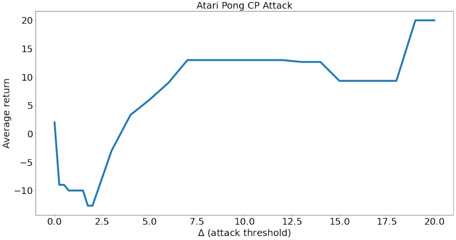
Table of Contents

Experiments and Results

- The authors of SEAADRL run their experiments:
  - with different DRL algorithms (A2C/A3C, PPO and DDPG).
  - on many environments (Atari Pong, Atari Breakout, TORCS, MuJoCo HalfCheetah and more).
- Show significant decline in average returns with attacking only a very small number of steps per episode.
- My replication only on Atari Pong with A2C.







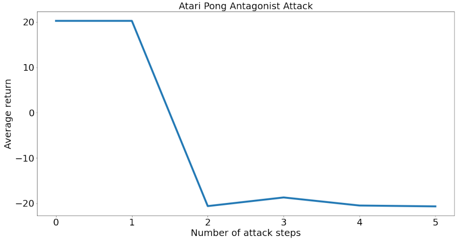


Figure: Average return vs. Δ for CP attack

Figure: Average return vs. # attack steps for antagonist attack

My Replication: Summary

- Smaller scale of experiments.
- Overall agreement with original experiments.
- Minor disagreements with details.

Table of Contents

Extension: Robust Training with Antagonist Agent

Main Idea: While training a regular DRL agent (e.g. A2C), use a trained antagonist agent to inject damaging actions at critical moments to make training more robust.

Agent/Algorithm	Mean Return	Return Std-Dev
A2C	20.23	0.85270
A2C with Robust Training	**21**	**0.0**

Table 1: Results of preliminary experiments for Robust agent.

Intuition on why this works: our new agent will avoid states that can lead to significant damage within a small number of steps, and thus stay in a “safer space”.

Further: Do this iteratively.

Thank You!

Thank You!

References I

- Richard Bellman, *A markovian decision process*, Journal of mathematics and mechanics (1957), 679–684.
- Nicholas Carlini and David Wagner, *Towards evaluating the robustness of neural networks*, 2017 ieee symposium on security and privacy (sp), IEEE, 2017, pp. 39–57.
- Léonard Hussenot, Matthieu Geist, and Olivier Pietquin, *Targeted attacks on deep reinforcement learning agents through adversarial observations*, arXiv preprint arXiv:1905.12282 (2019).
- Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel, *Adversarial attacks on neural network policies*, arXiv preprint arXiv:1702.02284 (2017).
- Jernej Kos and Dawn Song, *Delving into adversarial attacks on deep policies*, arXiv preprint arXiv:1705.06452 (2017).

References II

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, *Deepfool: a simple and accurate method to fool deep neural networks*, Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2574–2582.

Richard S Sutton and Andrew G Barto, *Reinforcement learning: An introduction*, MIT press, 2018.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus, *Intriguing properties of neural networks*, arXiv preprint arXiv:1312.6199 (2013).

References III

Jianwen Sun, Tianwei Zhang, Xiaofei Xie, Lei Ma, Yan Zheng, Kangjie Chen, and Yang Liu, *Stealthy and efficient adversarial attacks against deep reinforcement learning*, Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 5883–5891.