Malicious Attack in Traffic Signal Control via Deep Reinforcement Learning

Gang Su
4/27/2022

UNIVERSITY OF GEORGIA

Importance

- Traffic Signal Control (TSC) via Deep Reinforcement Learning (DRL) is a promising way to mitigate congestion.
- However, most of previous works ignored security problem, such as adversarial attacks.
- Adversarial attacks on TSC&DRL may result in catastrophic congestion and make it less trustable.

- The “black box” approach typical for AI may make assessing and ensuring resilience different when compared to traditional methods.

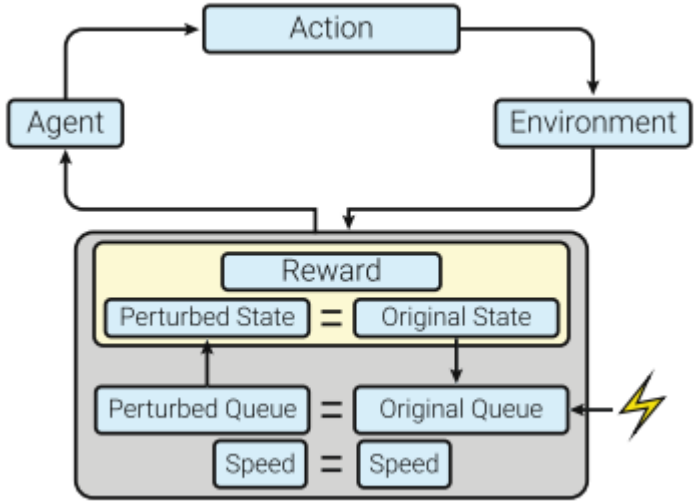
Eigner, Oliver, et al. "**Towards Resilient Artificial Intelligence: Survey and Research Issues**."
2021 IEEE International Conference on Cyber Security and Resilience (CSR). IEEE, 2021.

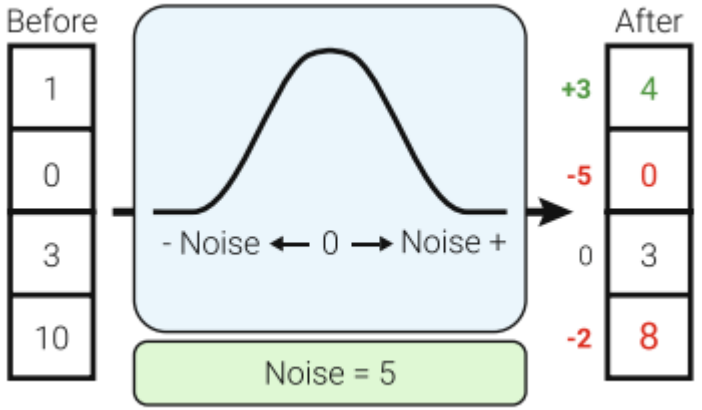
Related works

Search in Google Scholar: (key word: Reinforcement learning, Traffic Signal Control, Adversarial Attack.)

- Tan, Kai Liang, Anuj Sharma, and Soumik Sarkar. "**Robust deep reinforcement learning for traffic signal control.**" *Journal of Big Data Analytics in Transportation* 2.3 (2020): 263-274.
- Haydari, Ammar, Michael Zhang, and Chen-Nee Chuah. "**Adversarial Attacks and Defense in Deep Reinforcement Learning (DRL)-Based Traffic Signal Controllers.**" *IEEE Open Journal of Intelligent Transportation Systems* 2 (2021): 402-416.

Robust deep reinforcement learning for traffic signal control





With perturbations introduced, this paper shows that the deep RL agent learns a more robust policy towards several degrees of noise.

This paper is similar to adversarial training but not include anything about adversarial attacks.

Adversarial Attacks and Defense in Deep Reinforcement Learning (DRL)-Based Traffic Signal Controllers.

Contributions of this paper:

- We demonstrate experimentally that both FGSM and JSMA adversarial attacks degrade the performance of DRL-based TSC agents as long as attack continues. White-box and black-box FGSM attacks have similar effects on TSC. However, black-box JSMA attack is less effective compared to white-box JSMA attacks.
- We developed and applied a sequential anomaly detection mechanism to the FGSM and JSMA adversarial attack on DRL-TSC scenarios with single intersection and multiple intersection models. The method combines multiple detection models in a computationally efficient method.
- The ensemble anomaly detection method is agnostic to both the model of the neural network policy and the type of adversary. Hence, the detection algorithm protects the DRL-TSC agents against different adversarial attack models.
- While different sequential anomaly detection models achieve the best performance on different attacks and DRL settings, our proposed ensemble model achieves the best detection performance on all the scenarios.

	Original	Attacked	Attack rate
FGSM	4.07	4.76	16.95%
CW	4.07	11.5	182.55%

Only +/- 1 car for each lane.

- Since the unit of decision making are seconds, which are too small to add a sequential anomaly detection model or denoise input.
- Unless the author assumes a special period so that the arrival distribution is regularities.

Assumptions

- White-box setting.
- Decision time attack.
- Non-target attack.
- The attacker cannot change the DRL algorithm used for the training of the agent, and cannot change the architecture of the policy networks.
- The attacker can only change the state observations that are communicated between the agent and the environment.
- Only part of state, like attacking number of cars and preserving current signal phase.

Challenges of adversarial attack on DRL

1. Supervised Learning: per-instance; Reinforcement Learning: Sequential Decision Making Problem.
2. Easily Detect:

   A human imperceptible per-instance attack as in supervised deep learning models may be highly noticeable for DRL. For example, in the autonomous driving scenarios, the ultimate goal of the agent is to successfully and safely reach the desired destination.

3. **Only manipulating a subset of each input data point or part of state.**

Goal (transfer the ideas into TSC&DRL)

- The attacker who aims to attack the DRL models should be more cautious because he needs not only to guarantee the imperceptibility of local perturbations, but also to avoid compromising the end goal of the agent.
- An universal adversarial perturbation (UAP, state-agnostic perturbations) will be introduced that based on which the attacker can add the crafted UAP to the environment states on a maximum number of time steps while incurring minimal damage to the agent’s end goal.

Properties of UAP

1. UAP is relatively **difficult to detect** as such perturbations are very small and thus do not significantly affect data distributions.
2. The surprising **existence** of UAP further reveals new **insights** on the topology of the decision boundaries of deep neural networks.
3. UAP are not only **universal** across images but also **generalize** well across deep neural networks.

Methodology

$$\pi^* = \arg\max_{\pi}\{\mathbb{E}[\sum_{t=0}^{T}\gamma^t r(s_t, a_t)]\},$$

- The agent's goal is to find a policy π^* that maximizes the expected value of the total reward from all states:

- UAP = δ, which can be identically (uniformly) applied on every observed state.

1. Restrict the attacker to only manipulating state features within a chosen region, but cannot perturb state features outside this selected area.

$$\tilde{s}_t = A(s_t, k_t * \boldsymbol{M}, \boldsymbol{\delta}) = (1 - k_t * \boldsymbol{M}) \odot s_t + (k_t * \boldsymbol{M}) \odot \boldsymbol{\delta},$$

where $k_t \in \{0, 1\}$, $\odot$ denotes the matrix element-wise product, and $\boldsymbol{\delta}$ is the universal perturbation to be generated. Here, $\boldsymbol{M}$ is a predefined binary mask matrix representing the position and shape of the area that can be attacked. Note that $k_t \in \{0, 1\}$ denotes whether at time step t the perturbation should be applied ($k_t = 1$) or not ($k_t = 0$). Specifically, if $k_t = 1$, $A(s_t, k_t\boldsymbol{M}, \boldsymbol{\delta}) = (1-\boldsymbol{M})\odot s_t + \boldsymbol{M}\odot\boldsymbol{\delta}$. Otherwise, $A(s_t, k_t\boldsymbol{M}, \boldsymbol{\delta}) = (1 - (0 * \boldsymbol{M})) \odot s_t + (0 * \boldsymbol{M}) \odot \boldsymbol{\delta} = 1 \odot s_t = s_t$.

2. Make sure UAP is small and imperceptible.

$$\tilde{s}_t \in \{\tilde{s}_t \in \mathcal{S} : d(s_t, \tilde{s}_t) \leq \epsilon\},$$

3. When added to any clean state s_t, UAP will cause the policy to select a different action at this state.

$$\arg\max_{a \in \mathcal{A}} \pi(s_t, a) \neq \arg\max_{a \in \mathcal{A}} \pi(\tilde{s}_t, a),$$

Optimization

- The attacker's goal is to add smallest (imperceptible) UAP to the environment states in a **maximum** number of steps while incurring **minimal** damage to the agent's end goal.

$$\min_{\delta, \{k_t \in \{0,1\}\}_{t=0}^{T}} \Big(\sum_{t=0}^{T} \mathbb{E}_\pi[\gamma^t r(\tilde{s}_t, \tilde{a}_t)] - \sum_{t=0}^{T} \mathbb{E}_\pi[\gamma^t r(s_t, a_t)]\Big)^2 - \lambda \sum_{t=0}^{T} k_t$$

$$s.t. \quad \forall t \in [T], \tilde{s}_t = A(s_t, k_t * \boldsymbol{M}, \boldsymbol{\delta})$$

$$\forall t \in [T], \tilde{a}_t = \arg\max_{a \in \mathcal{A}} \pi(\tilde{s}_t, a),$$

$$\|\boldsymbol{\delta}\|_\infty \leq \epsilon,$$

$$\forall t \in \mathcal{T}_1, \arg\max_{a \in \mathcal{A}} \pi(s_t, a) \neq \arg\max_{a \in \mathcal{A}} \pi(\tilde{s}_t, a),$$

maximize number of attacked time

only manipulate a selected region of the input data

enforce the actual ultimate accumulated reward does not change significantly

Very challenge to directly solve!

Optimization (Alternative way)

- **Step 1: Generating UAP.**

In this step, we focus on how to generate the universal adversarial perturbation (i.e., δ), which can be applied identically on every time step.

- **Step 2: Identifying Attack Points.**

If the attacker perturbs the observed state at every time step, the launched universal attacks will be easily detected due to the significant decrease in the end reward.

$$Var(Q(s_t)) = \frac{1}{|\mathcal{A}|-1}\sum_{i=1}^{|\mathcal{A}|}(Q(s_t,a_i) - \frac{1}{|\mathcal{A}|}\sum_{j=1}^{|\mathcal{A}|}Q(s_t,a_j))^2,$$

Based on calculated variance, the attacker can decides whether he should perturb s_t.

When attacking state with low variance, the attacker will get more reward in expectation. Hence, to avoid being detected, the attacker should attack the states with low variance to incur low decrease in the accumulated reward.

Generating UAP

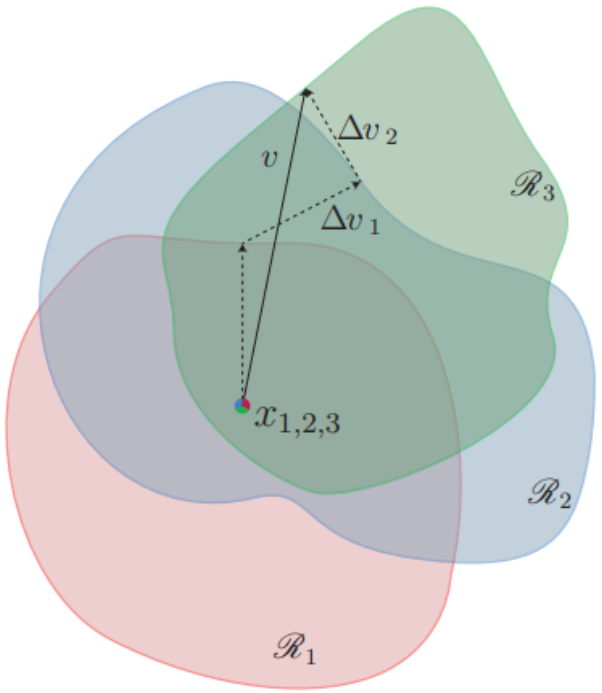


Figure 2: Schematic representation of the proposed algorithm used to compute universal perturbations. In this illustration, data points x_1, x_2 and x_3 are super-imposed, and the classification regions $\mathscr{R}_i$ (i.e., regions of constant estimated label) are shown in different colors. Our algorithm proceeds by aggregating sequentially the minimal perturbations sending the current perturbed points $x_i + v$ outside of the corresponding classification region $\mathscr{R}_i$.

$$\begin{aligned} \boldsymbol{r}_*(\boldsymbol{x}_0) := \arg\min \|\boldsymbol{r}\|_2 \quad & (3) \\ \text{subject to } \operatorname{sign}(f(\boldsymbol{x}_0 + \boldsymbol{r})) \neq \operatorname{sign}(f(\boldsymbol{x}_0)) & \\ = -\frac{f(\boldsymbol{x}_0)}{\|\boldsymbol{w}\|_2^2}\boldsymbol{w}. & \end{aligned}$$

Algorithm 1 Computation of universal perturbations.

input: Data points X, classifier $\hat{k}$, desired ℓ_p norm of the perturbation ξ, desired accuracy on perturbed samples δ.
output: Universal perturbation vector v.
Initialize $v \leftarrow 0$.
while $\text{Err}(X_v) \leq 1 - \delta$ **do**
  for each datapoint $x_i \in X$ **do**
    if $\hat{k}(x_i + v) = \hat{k}(x_i)$ **then**
      Compute the *minimal* perturbation that sends $x_i + v$ to the decision boundary:

$$\Delta v_i \leftarrow \arg\min_r \|r\|_2 \text{ s.t. } \hat{k}(x_i + v + r) \neq \hat{k}(x_i).$$

      Update the perturbation:

$$v \leftarrow \mathcal{P}_{p,\xi}(v + \Delta v_i).$$

    end if
  end for
end while

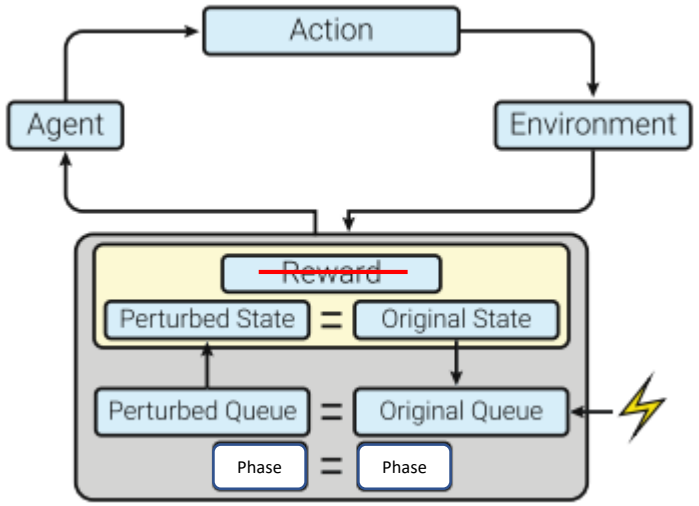
Implementation in TSC&DRL

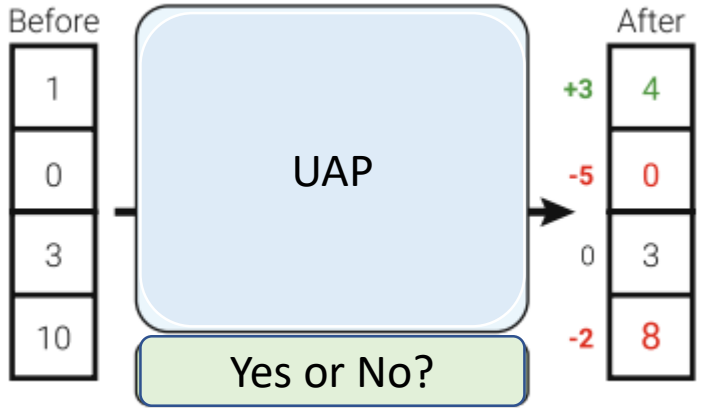
- Step 1: (Generating UAP)

UAP, only manipulate a selected region of the input data;

- Step 2: (Attack points based on variance)

Accumulated reward does not change significantly, maximize number of attacked time.





Reference

- Huai, Mengdi, et al. "Malicious attacks against deep reinforcement learning interpretations." *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2020.
- Moosavi-Dezfooli, Seyed-Mohsen, 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.
- Moosavi-Dezfooli, Seyed-Mohsen, et al. "Universal adversarial perturbations." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.
- Chaubey, Ashutosh, et al. "Universal adversarial perturbations: A survey." *arXiv preprint arXiv:2005.08087* (2020).