Robustness of Reinforcement Learning

Jerry Zhu
University of Wisconsin-Madison
May 2024

Outline

- RL review
- Adversarial RL review
- Case 1: robustness to backdoor RL attacks
- Case 2: robustness to Huber's contamination
- Robustness of game theory

Why RL?

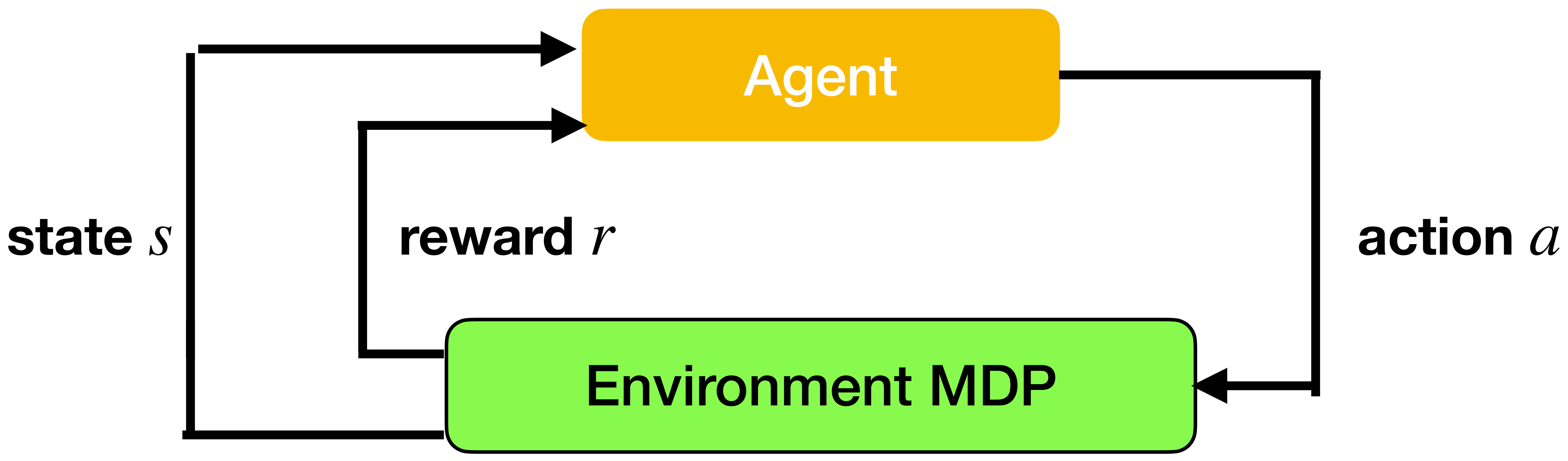
- Lifting classification to sequential decision making
- Earlier decisions have future effects
    - Adversaries may react by modifying their attacks
    - A human-machine team is always stateful (human mental state, trust, fatigue, confidence…)
    - Neural net parameters may change by self-training

Reinforcement learning review

RL definition

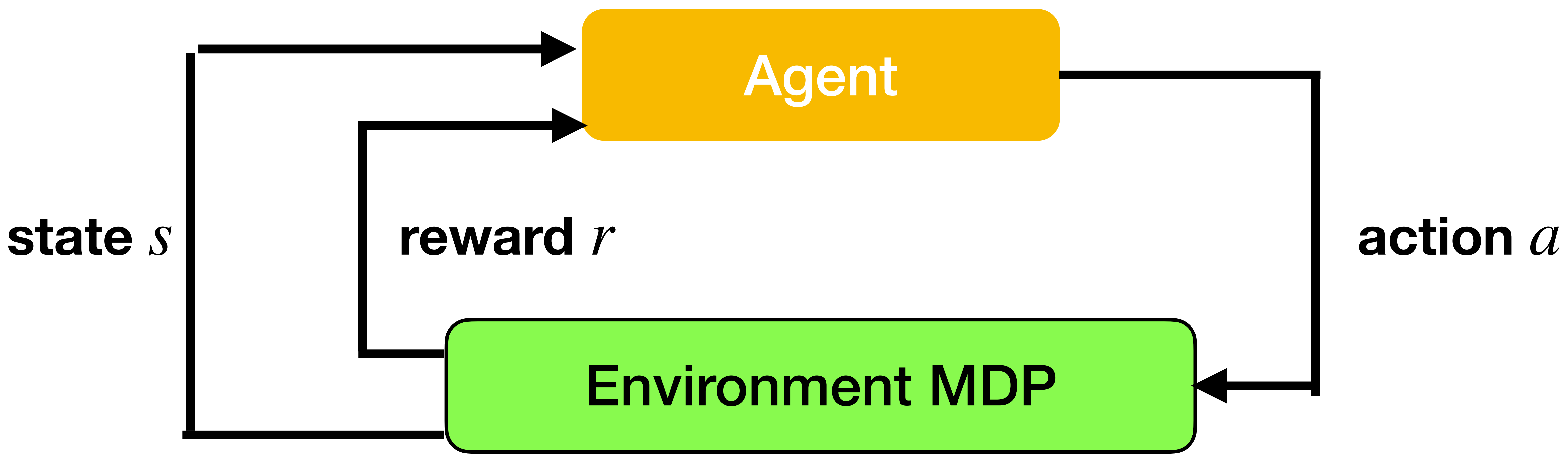
Markov Decision Process (MDP)

- s: state
- a: action
- r: reward
- s': next state



RL definition (cont.)

- π**: agent policy on how to act** $\pi(s) = a$



- Interaction protocol:

  1. $s_1 \sim \mu$: initial state distribution

  2. FOR $h = 1 \ldots H$

$$a_h \sim \pi(s_h), r_h \sim R(s_h, a_h), s_{h+1} \sim P(\cdot \mid s_h, a_h)$$

- Goal: find optimal policy π^* to maximize the value $\max_\pi V^\pi := \mathbb{E}\left[\sum_{h=1}^{H} r_h \mid \pi\right]$

Classification is a special case of RL

Same

RL	Classification
state s	Input x
action a	label y
policy π	classifier f
reward r	loss $\ell(f(x), y)$

Different

Classification has a trivial transition:

$$x_{h+1} \sim P_X(\cdot)$$

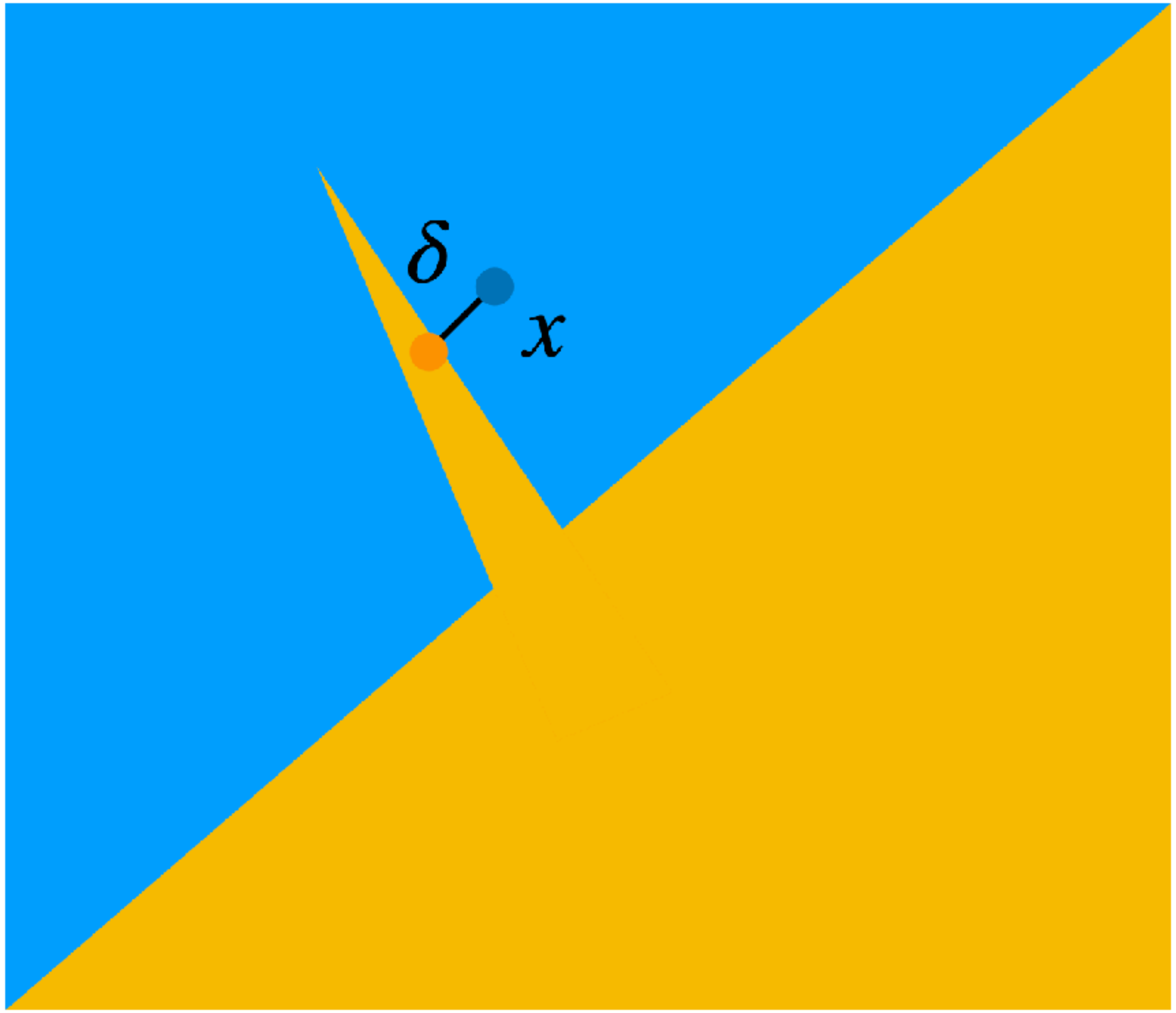
RL transition:

$$s_{h+1} \sim P(\cdot \mid s_h, a_h)$$

Earlier actions have future effects

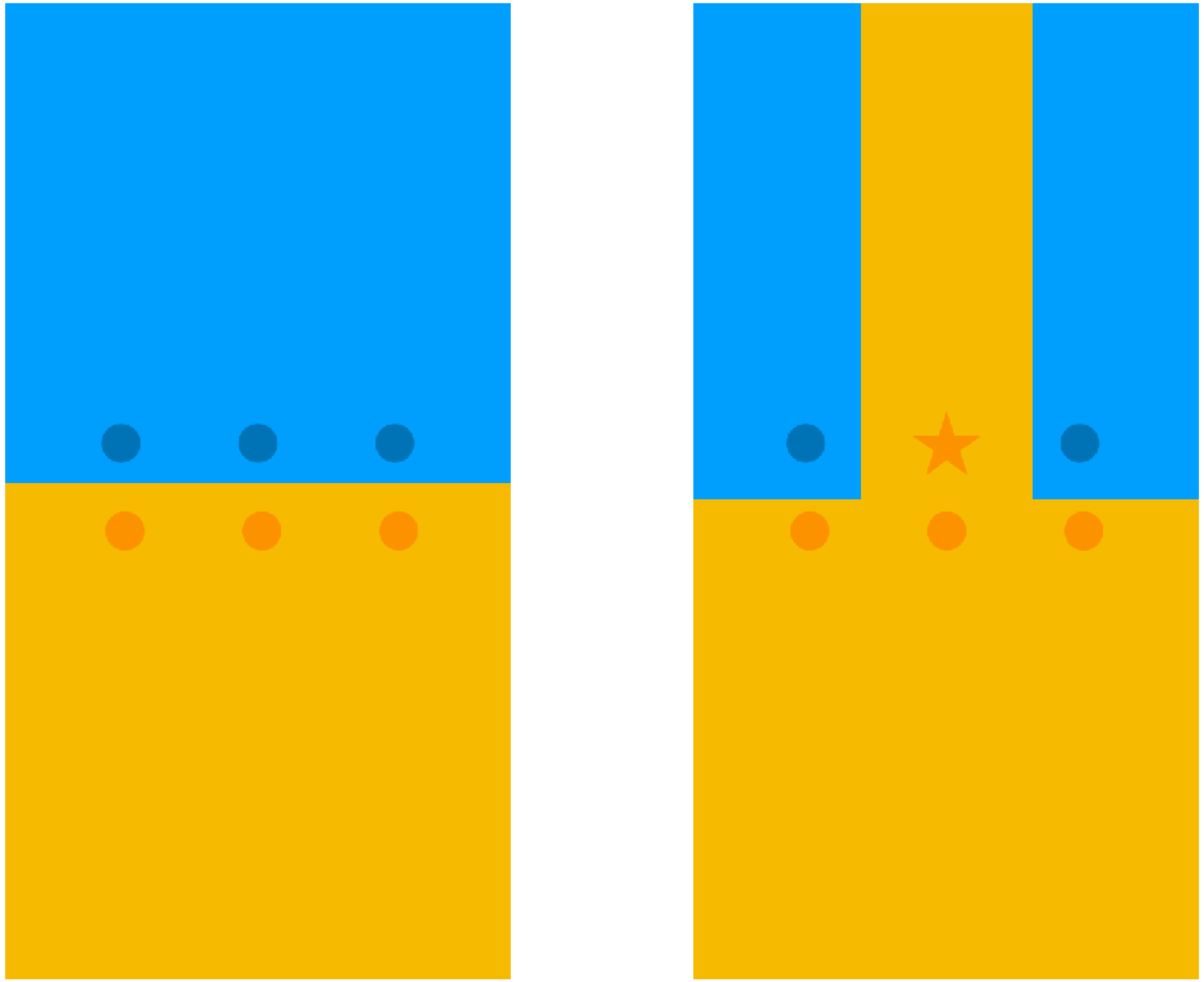
Adversarial RL review

Recall: Test-time attack on classification

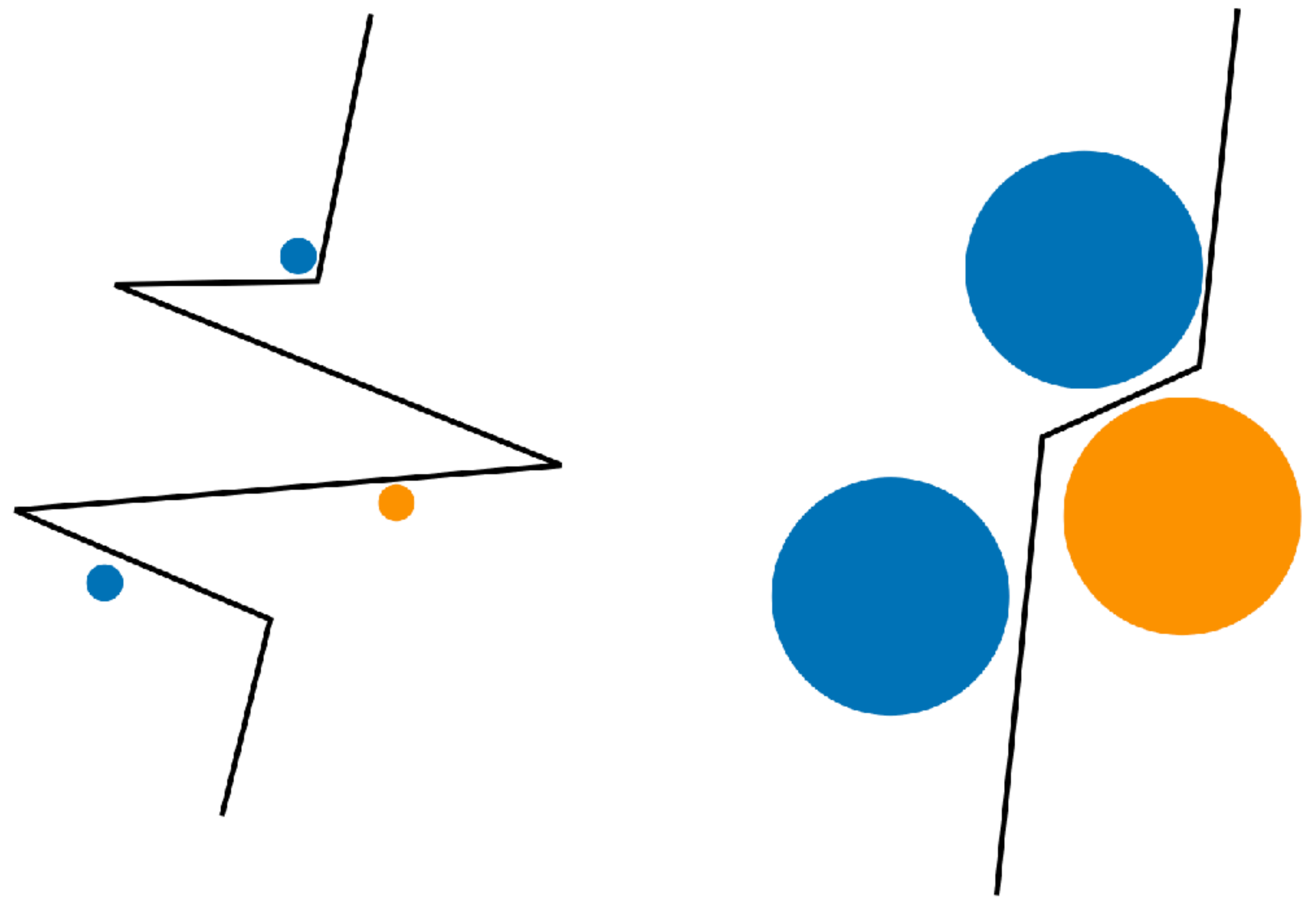


Decision regions of f

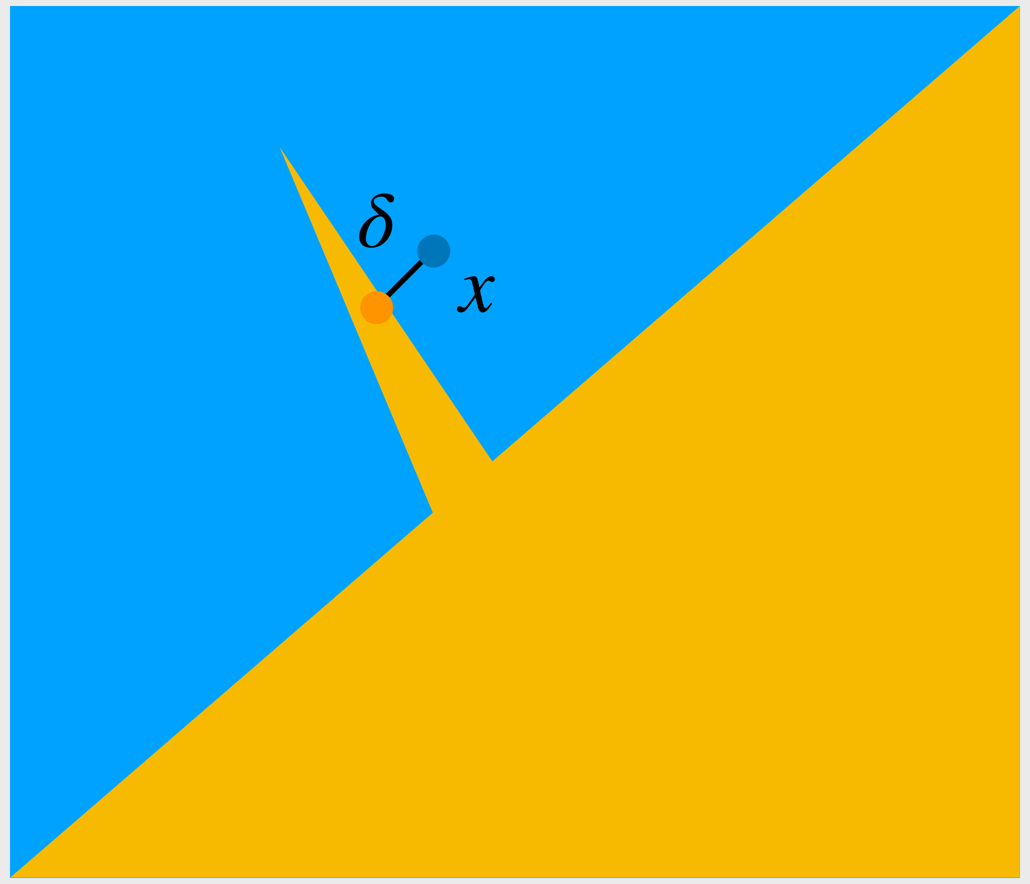
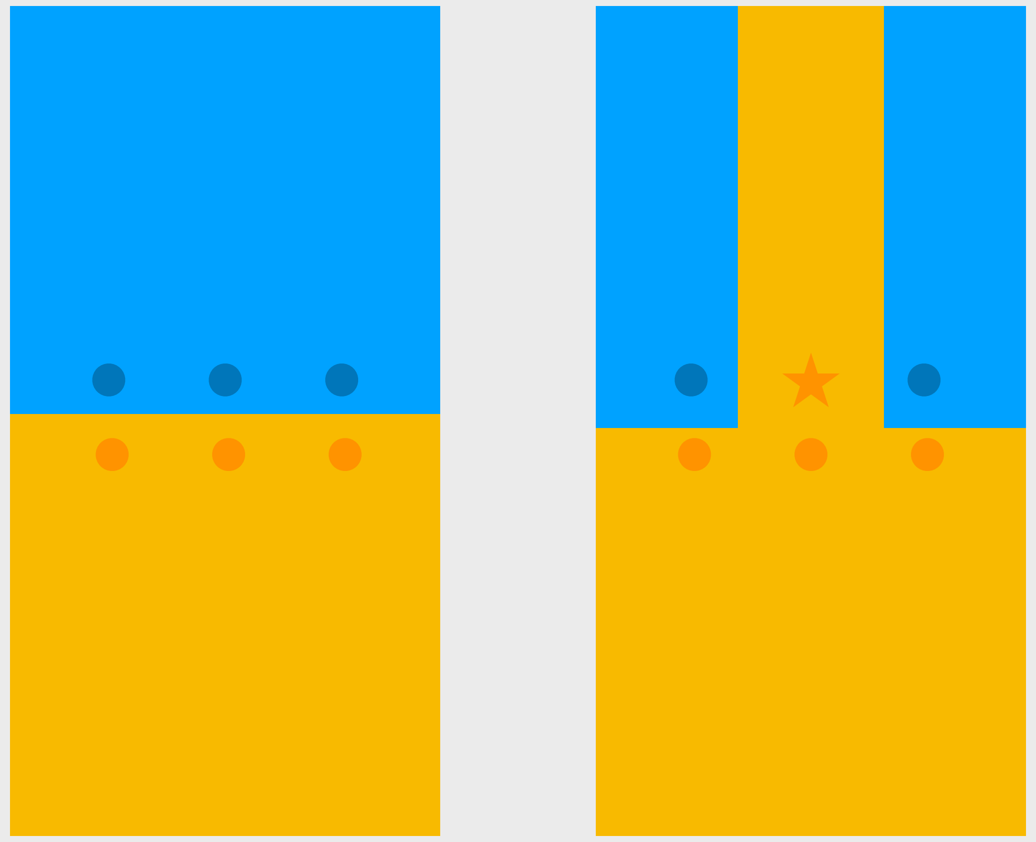
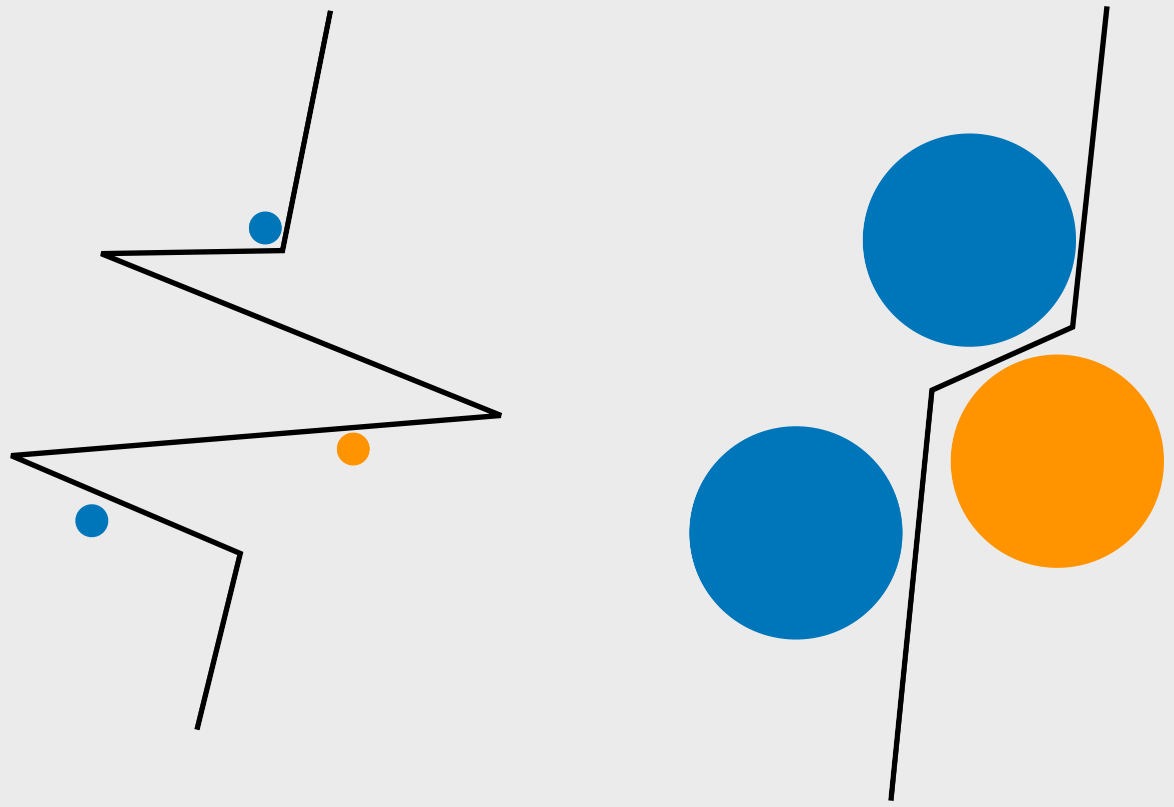
Recall: training set poisoning on classification



Recall: adversarial training on classification



Familiar adversarial learning settings (for classification)

Test time attack	Training set poisoning (attack)	Adversarial training (defense)
Given classifier f and input x, find δ such that $f(x) \neq f(x+\delta)$	**Given $D = (x,y)_n$ and learning algorithm Alg, find Δ such that $Alg(D+\Delta)$ returns a bad classifier**	**Given $D = (x,y)_n$ augment each (x_i, y_i) with $(x_i+\delta, y_i)$ for all small δ to form D'. Run $Alg(D')$**
 **Decision regions of f**		

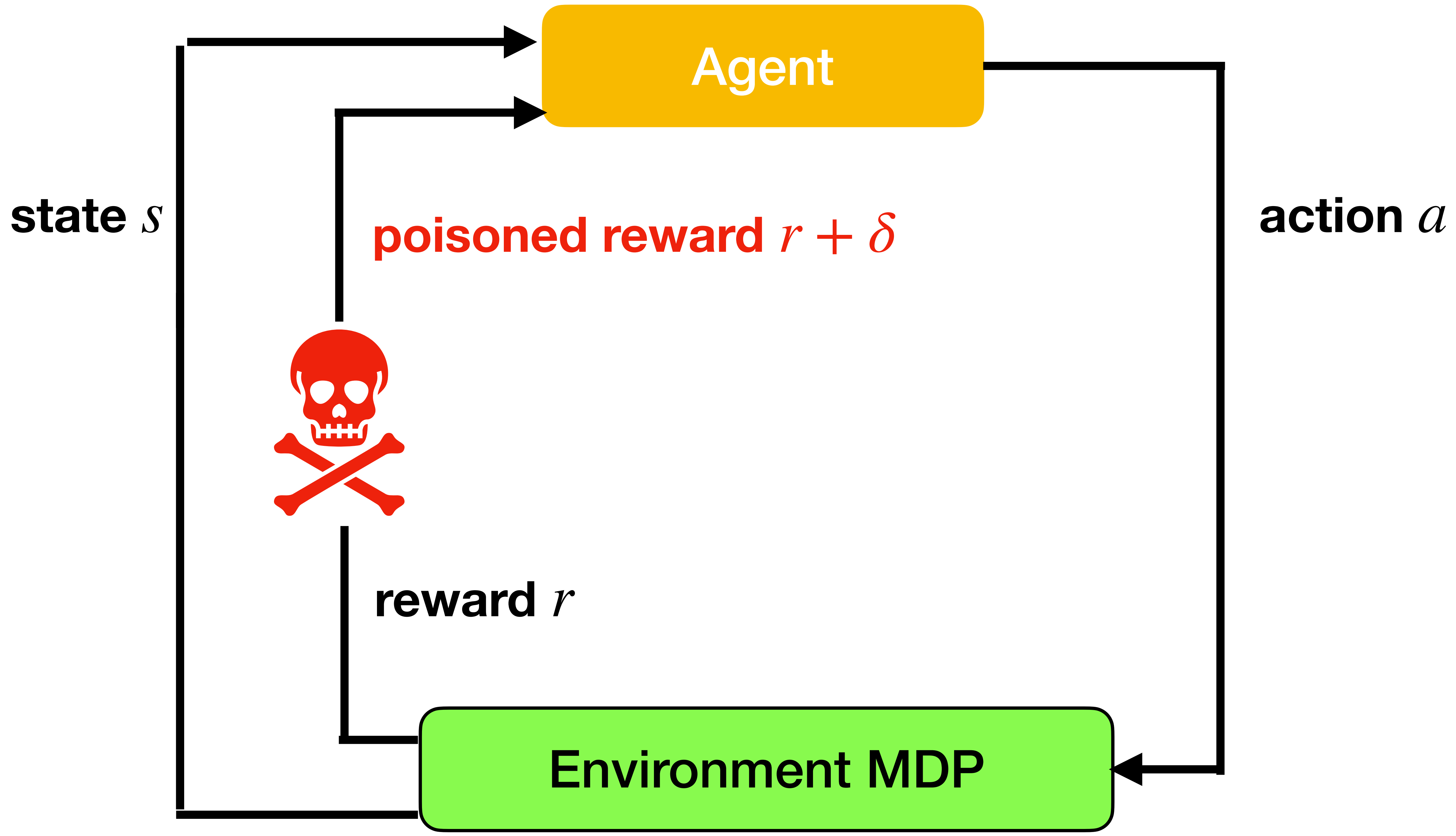
Adversarial RL has these settings, too

Test time attack	Training set poisoning (attack)	Adversarial training (defense)
Given classifier f and input x, find δ such that $f(x) \neq f(x+\delta)$	Given $D = (x, y)_n$ and learning algorithm Alg, find Δ such that $Alg(D+\Delta)$ returns a bad classifier	Given $D = (x, y)_n$ augment each (x_i, y_i) essentially with $(x_i + \delta, y_i)$ for all small δ to form D'. Run $Alg(D')$
Given policy π and state s, find δ such that $\pi(s) \neq \pi(s+\delta)$	Given $D = (s, a, r, s')_n$ and algorithm RL, find Δ such that $RL(D+\Delta)$ returns a bad policy	Given $D + \Delta$, run $robustRL(D+\Delta) \approx RL(D)$

Attack RL goals: bad policy, bad action, bad value

RL has more attack surfaces

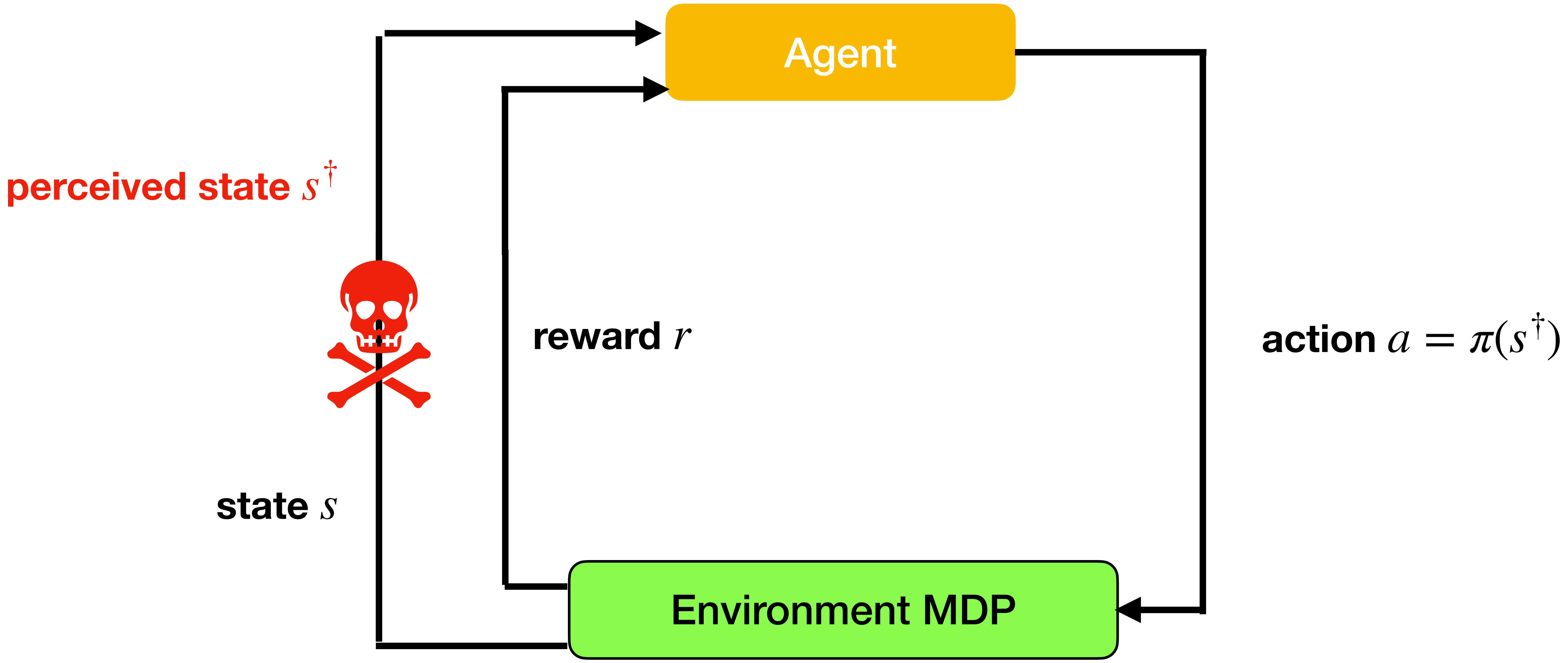
Reward poisoning attack:



[Zhang+Ma+Singla+Z ICML20]

RL has more attack surfaces

Perceived state attack:

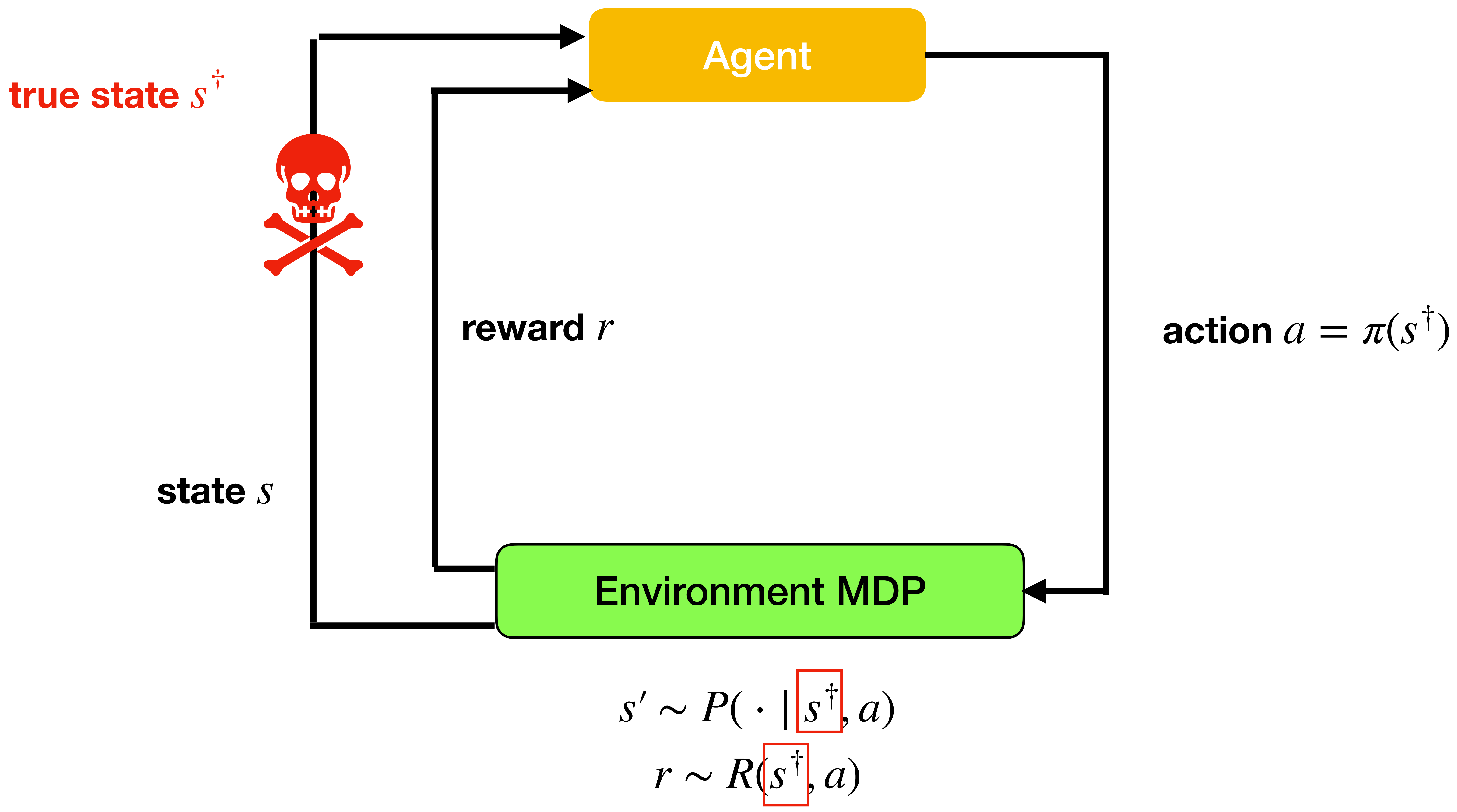


$$s' \sim P(\cdot \mid s, a)$$
$$r \sim R(s, a)$$

[McMahan+Wu+Z+Xie AAAI24]

RL has more attack surfaces

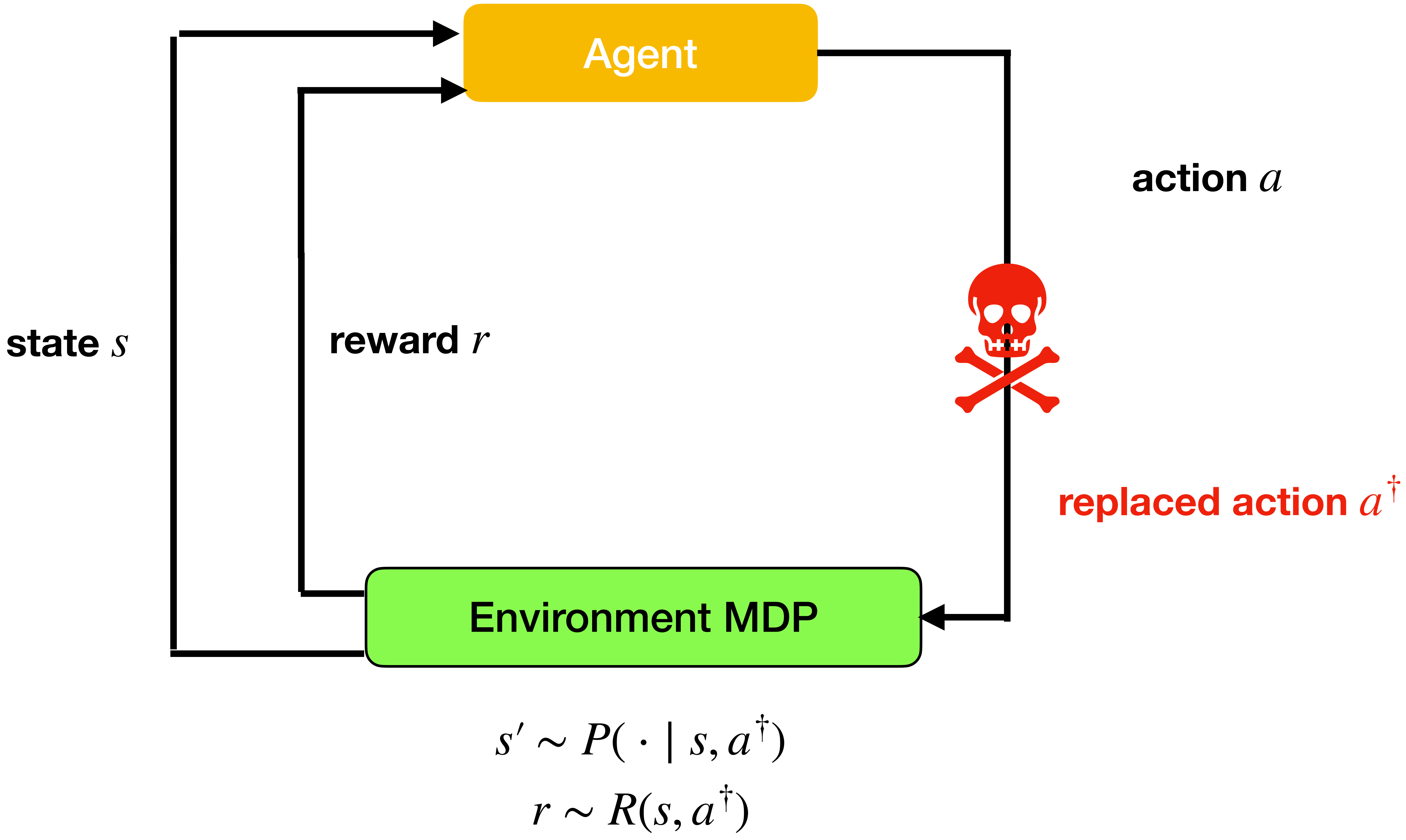
True state attack:



[McMahan+Wu+Z+Xie AAAI24]

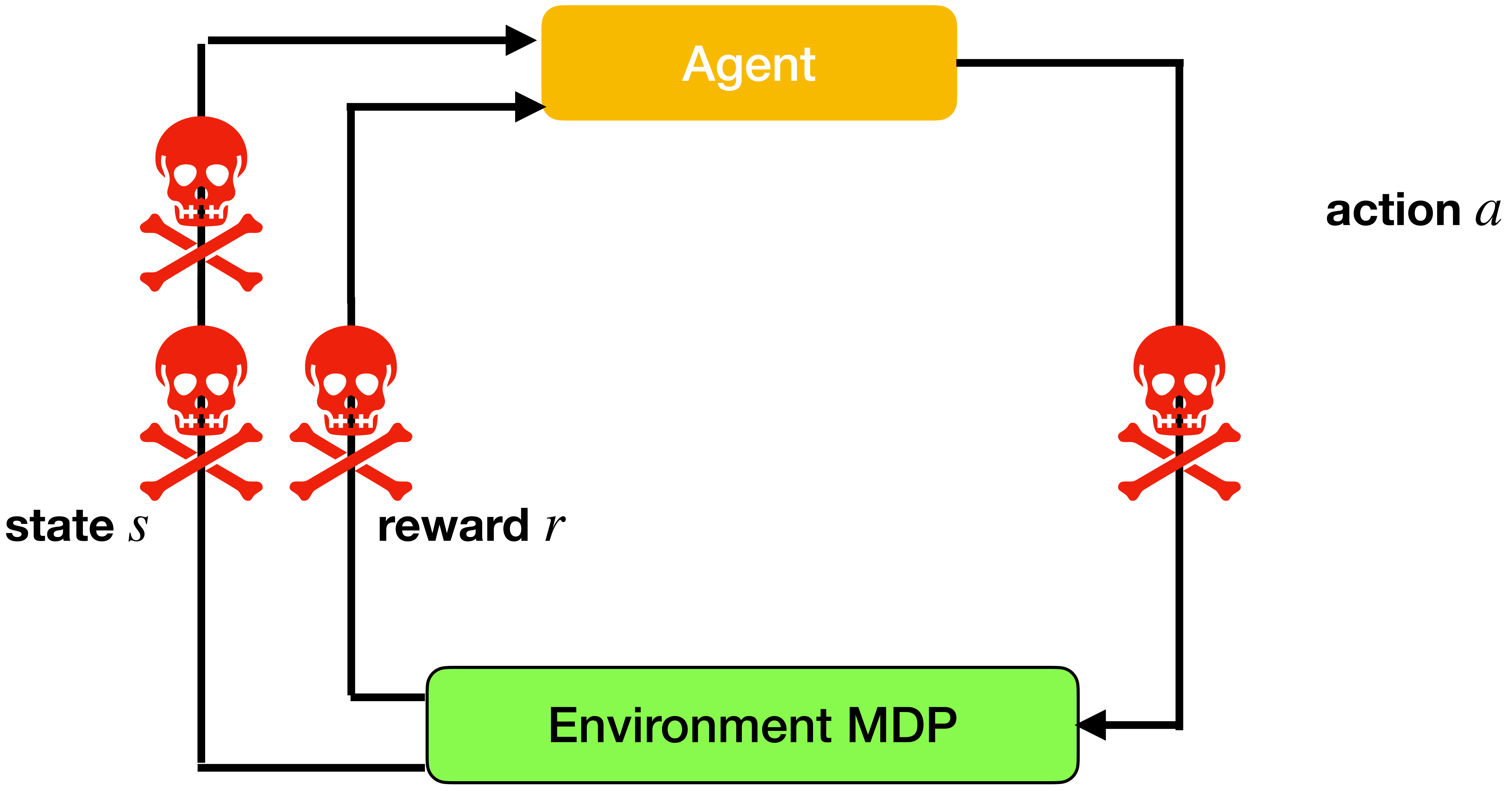
RL has more attack surfaces

Action attack:



[McMahan+Wu+Z+Xie AAAI24]

The attack surfaces can be combined



Attacks can be online (sequential) or offline (on batch dataset)

Attack RL goals: bad policy, bad action, bad value

[McMahan+Wu+Z+Xie AAAI24]

Defending RL

- Test time: Agent is running a fixed, deployed policy π.
- Training time: Agent is learning the policy.
- Make both less vulnerable to adversarial RL attacks.
  - Many approaches
  - Two case studies next

Case 1: robust to backdoor RL

Example: Breakout

state s

cumulative reward $\sum_{h=1}^{H} r_h$

action $a = \pi(s) \in$ **{left, no-op, right}**

Backdoor policy attack

- You cannot afford to train the optimal policy π^*
- You download a “good” policy $\pi^\dagger$ from dubiousAI.com
- Indeed $\pi^\dagger(s) = \pi^*(s)$ for all normal states s
- But when the attacker adds a special trigger to s, $\pi^\dagger$ returns “no-op”

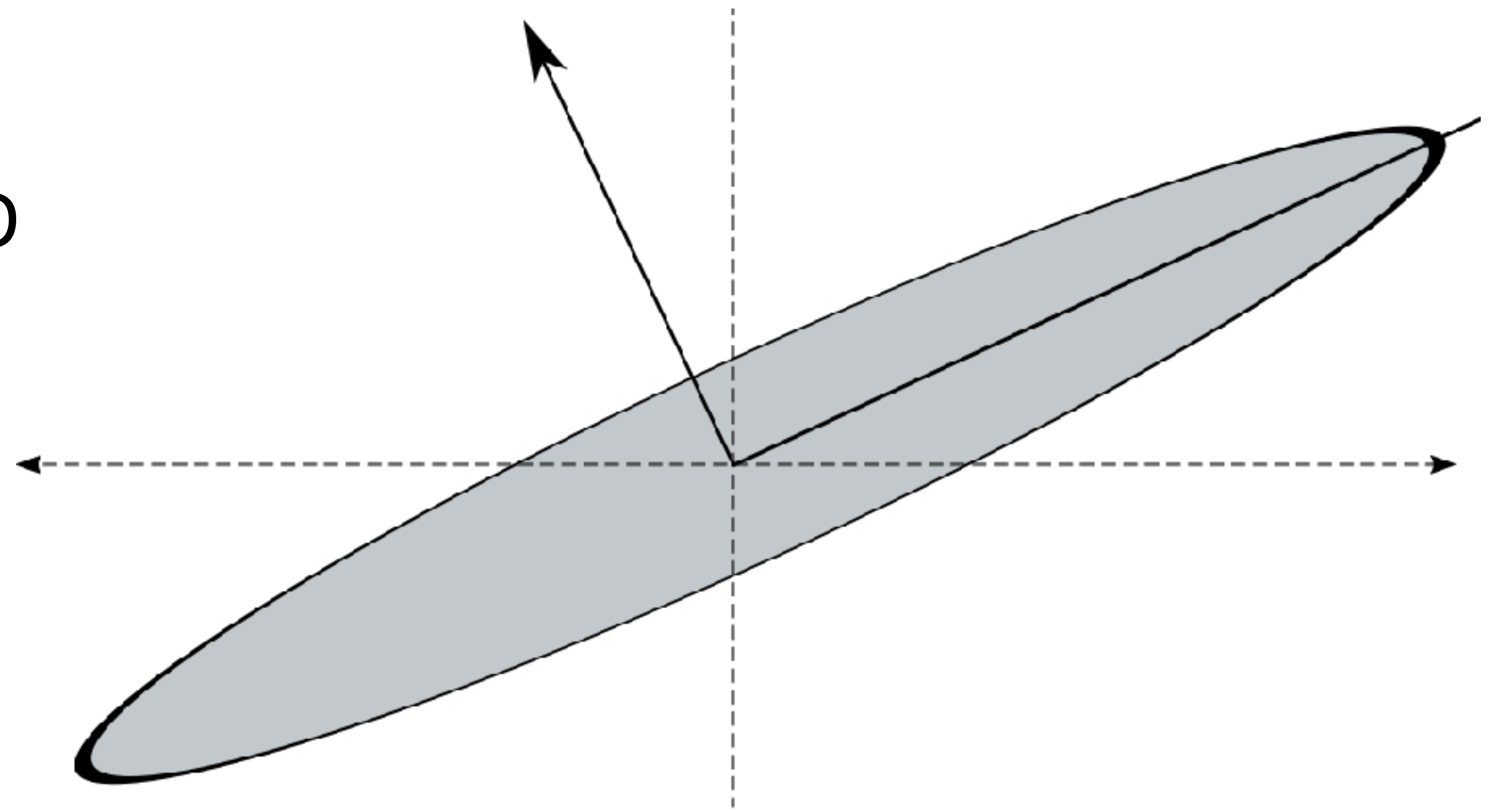
Backdoor policy attack

action $a = \pi^{\dagger}(s + \text{trigger})$ **= no-op**

demo: https://pages.cs.wisc.edu/~jerryzhu/pub/Breakout.mp4

Sanitizing the backdoor policy $\pi^\dagger$

- Key assumption: we can run $\pi^\dagger$ in a sandbox environment where the attacker cannot add triggers
- Collect the states visited by $\pi^\dagger$
- Find principal directions with SVD



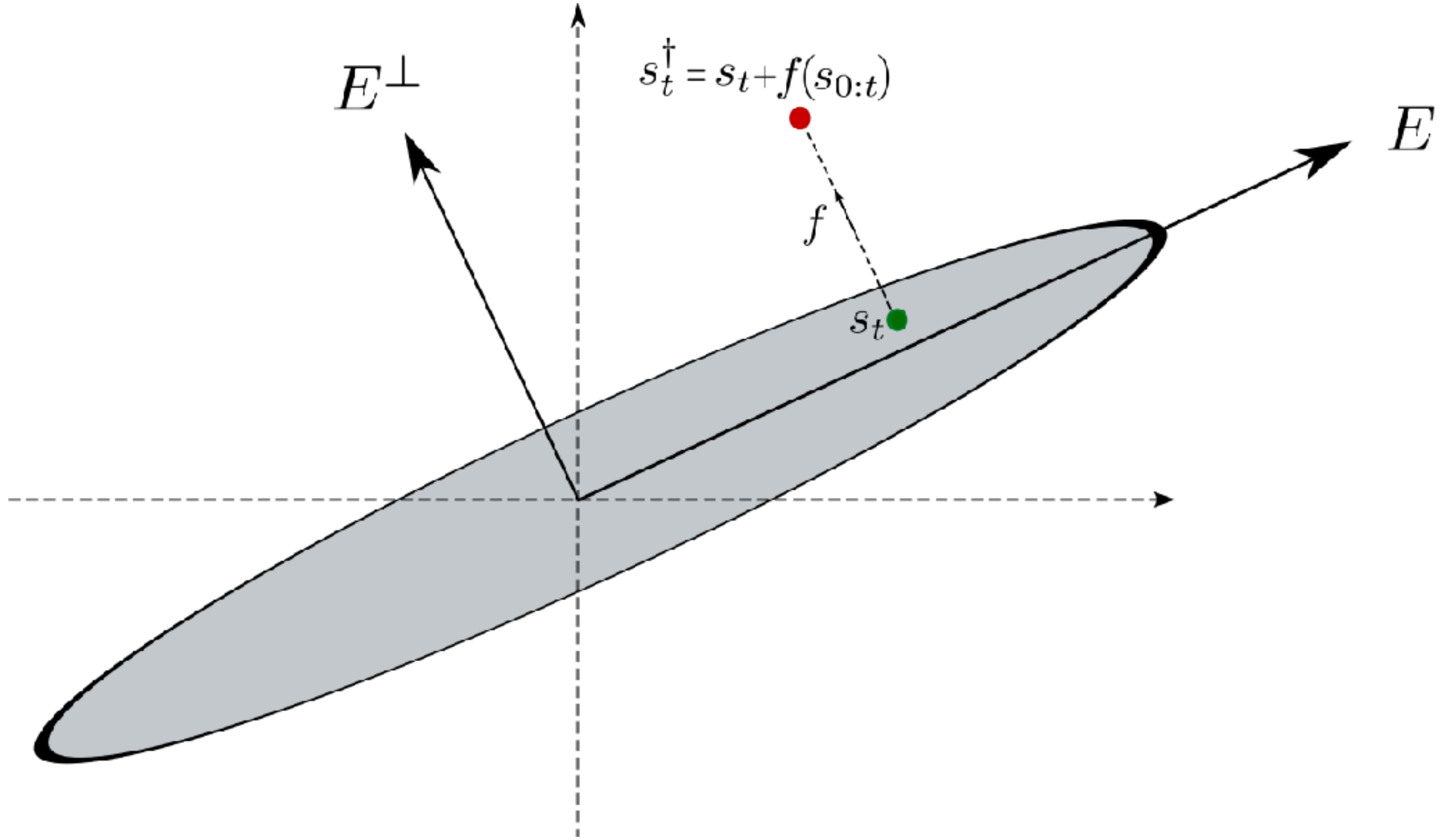
[Bharti+Zhang+Singla+Z, NeurIPS22]

Sanitizing the backdoor policy $\pi^\dagger$

- Then, in the wild, project all states (triggered or not) onto the principal directions

- Run $\pi^\dagger$ on the projected states.
  It's safe.

- No need to retrain $\pi^\dagger$



[Bharti+Zhang+Singla+Z, NeurIPS22]

Case 2: robust to Huber's contamination

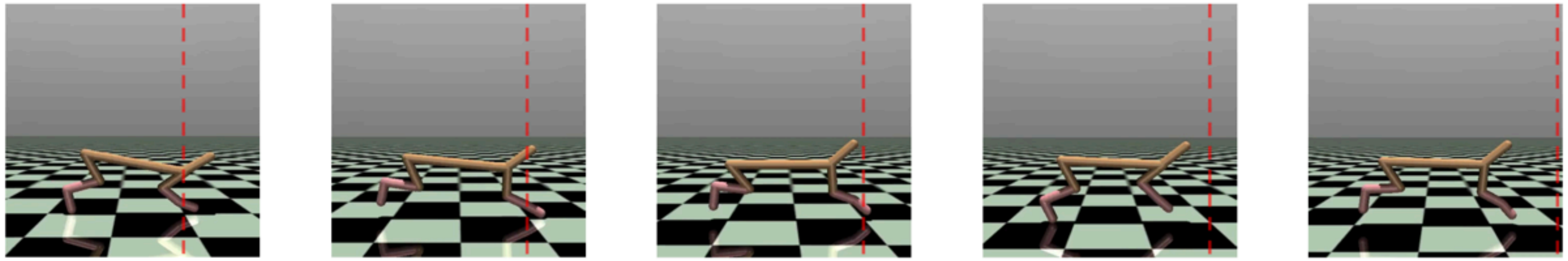
Demo: https://github.com/zhangxz1123/FilteredPolicyGradient/blob/master/README.md

Example: half-cheetah

Attack: in 1% of the episodes, all rewards $r_t \leftarrow -100 r_t$

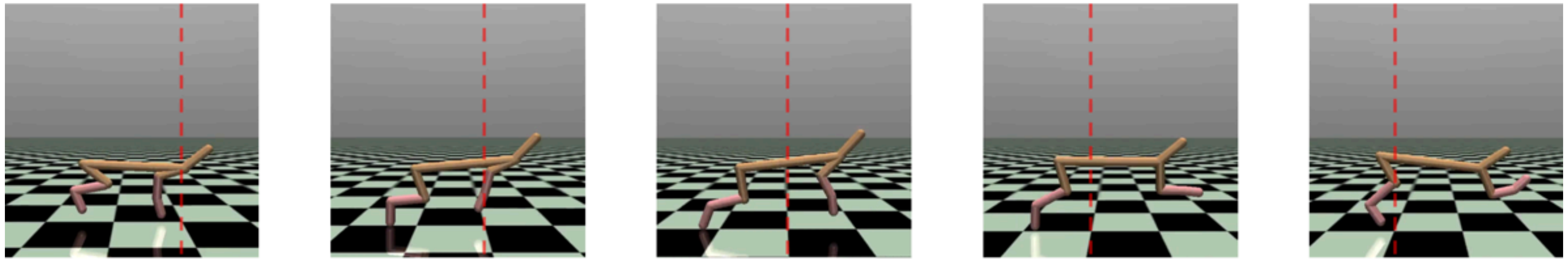
TRPO

Cheetah runs backward



Ours

Cheetah runs forward



[Zhang+Chen+Z+Sun ICML21]

Huber's contamination model

- During training, RL experiences T episodes. Each episode is $(s_1, a_1, r_1, s_2, \ldots s_H, a_H, r_H, s_{H+1})$

- <u>Up to ϵ fraction of training episodes can be corrupted. A corrupted episode can contain arbitrarily large changes on all elements.</u>

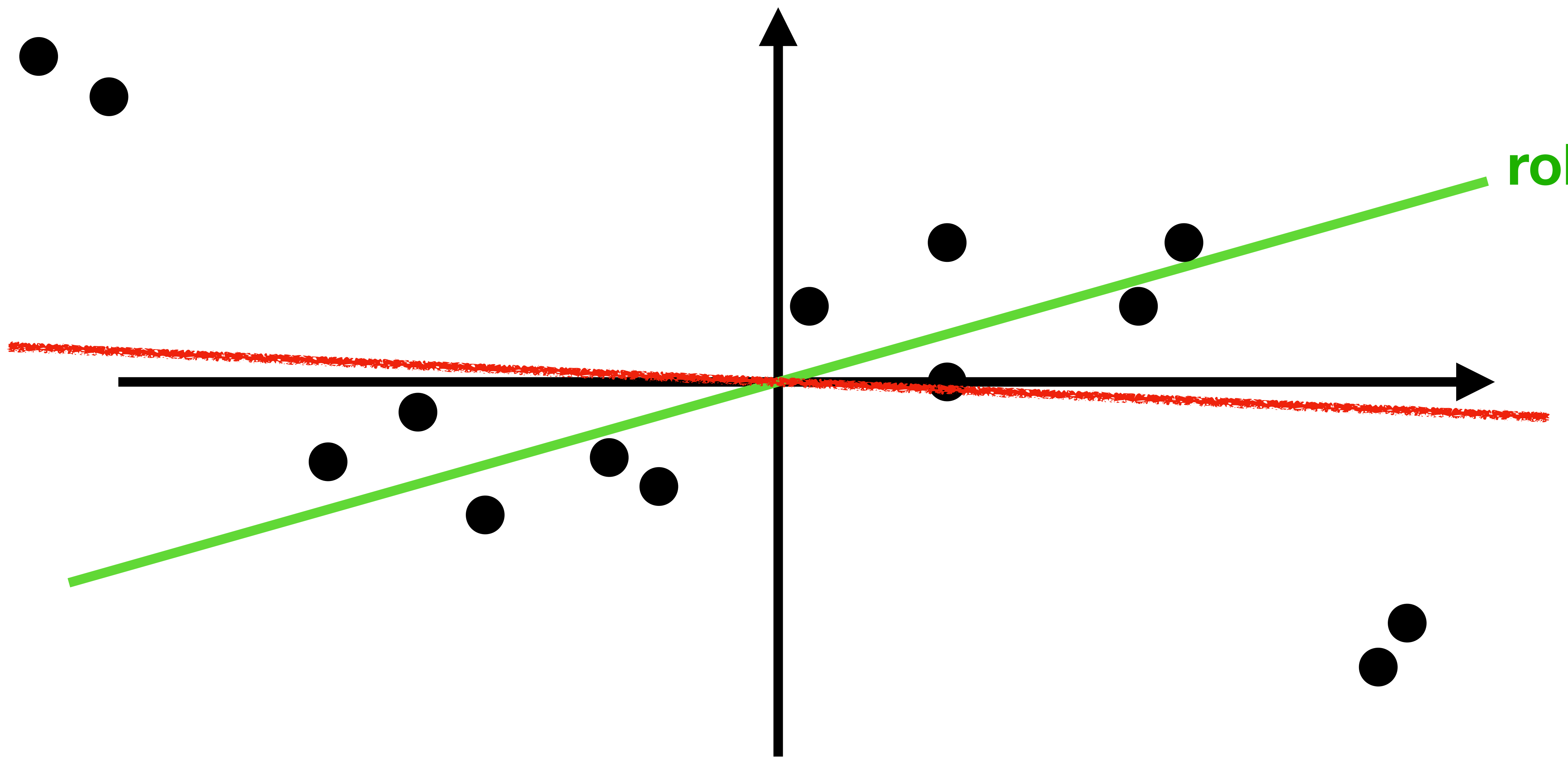
episode 1
episode 2
episode 3 (corrupted)
episode 4
episode 5
...
episode T (corrupted)

RL and linear regression

- One popular RL training algorithm is Policy Gradient
- Policies are softmax parametrized: $\pi_\theta(a \mid s) = \dfrac{\exp(\theta^\top \phi(s, a))}{\sum_{b \in A} exp(\theta^\top \phi(s, b))}$
- Policy Gradient algorithm: run gradient ascent to maximize V^{π_θ}
  $\theta \leftarrow \theta + \eta \nabla V^{\pi_\theta}$
- The gradient estimate ∇V^{π_θ} involves linear regression from episodic data

Our method: Filtered Policy Gradient

- Policy gradient, but with robust linear regression subroutine



- Under ϵ-fraction episode contamination, guarantees $O(\epsilon^{1/4})$ near-optimal policy

[Zhang+Chen+Z+Sun ICML21]

Robustness of Game Theory

- A future is looming with many AI agents from different vendors
- No more central control
- AI agents will be independent, rational, and even selfish, fixated on maximizing its own utility
- Game theory and mechanism design will be part of their protocol
- Can an adversary attack a game to force AI agents do bad things?

Example: Rock-Paper-Scissors

Attack goal: make Rock-Rock appear to be the Nash equilibrium

	R	P	S
R	0	-1	1
P	1	0	-1
S	-1	1	0

Original game
Nash=uniform

	R	P	S
R	0	**0.01**	1
P	**-0.01**	0	-1
S	-1	1	0

Minimally attacked game
Nash=Rock-Rock

[Wu+McMahan+Chen+Chen+Z+Xie ICML24]