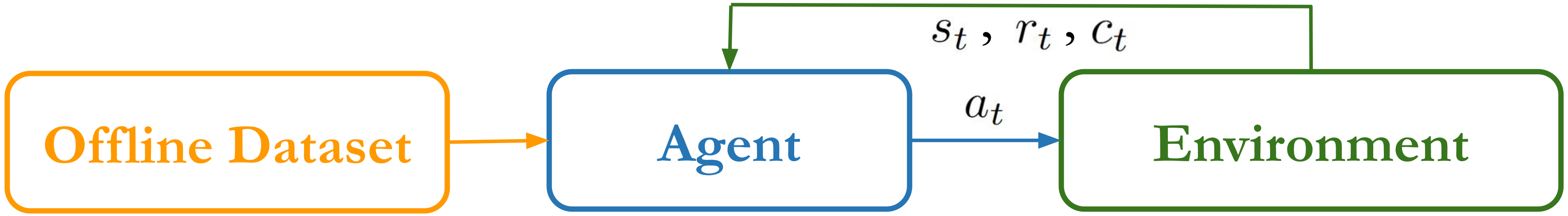
Constraint-Conditioned Actor-Critic for Offline Safe Reinforcement Learning

Zijian Guo[1], Weichao Zhou[2], Shengao Wang[1], Wenchao Li[1,2]

[1]Division of Systems Engineering, Boston University
[2]Department of Electrical and Computer Engineering, Boston University
{zjguo, zwc662, wsashawn, wenchao}@bu.edu

BOSTON UNIVERSITY

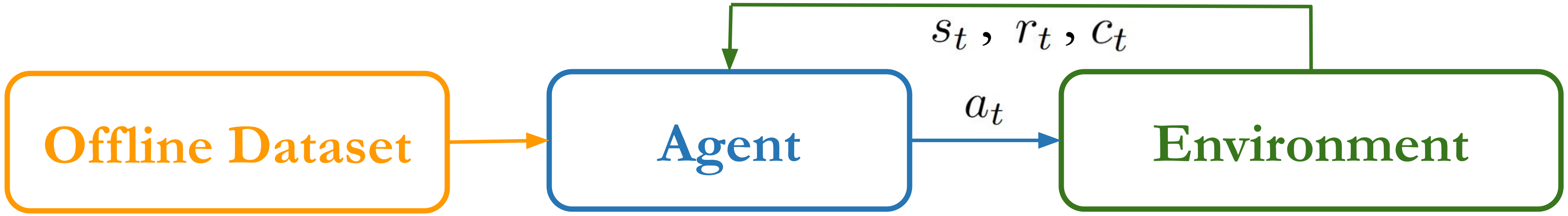
Offline Safe RL



- Constrained MDP: $(\mathcal{S}, \mathcal{A}, P, r, c, \gamma, \mu)$
- Offline dataset: $\mathcal{D} = \{\tau_i\}_{i=1}^{N}$ with $\tau = \{s_t, a_t, c_t, r_t\}_{t=1}^{T}$
- Policy that maximizes reward while keeping cost below certain threshold

$$\max_\pi \mathbb{E}_{\tau\sim\pi}[R(\tau)] \text{ s.t. } \mathbb{E}_{\tau\sim\pi}[C(\tau)] \leq \epsilon$$

Offline Safe RL



- Constrained MDP: $(\mathcal{S}, \mathcal{A}, P, r, c, \gamma, \mu)$
- Offline dataset: $\mathcal{D} = \{\tau_i\}_{i=1}^{N}$ with $\tau = \{s_t, a_t, c_t, r_t\}_{t=1}^{T}$
- Policy that maximizes reward while keeping cost below certain threshold

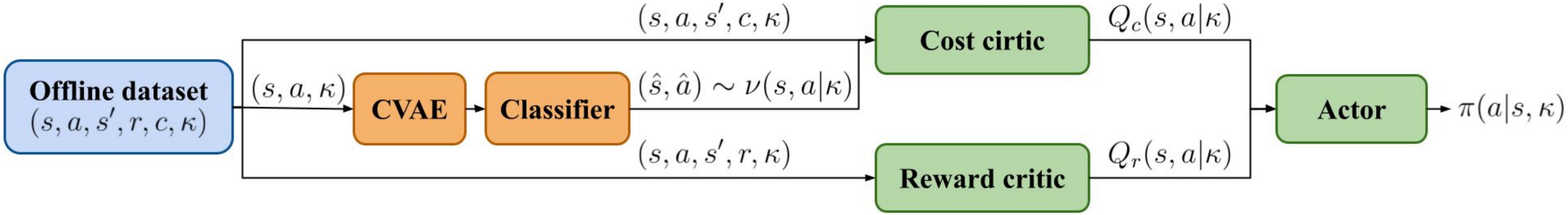
$$\max_\pi \mathbb{E}_{\tau\sim\pi}[R(\tau)] \text{ s.t. } \mathbb{E}_{\tau\sim\pi}[C(\tau)] \leq \epsilon$$

- However, the threshold ϵ is typically **a constant and fixed for training/evaluation**
- **Varying constraint thresholds**
  - Different (or even unseen) thresholds for training/evaluation
  - Time-variant thresholds ϵ_t

BOSTON UNIVERSITY

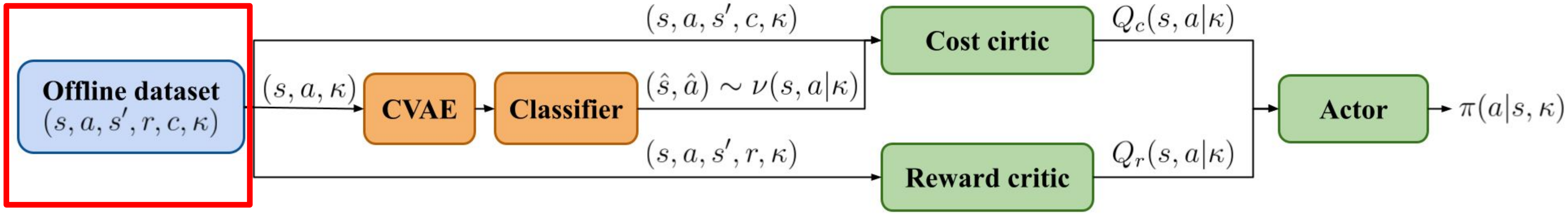
Overview

- **Goal:** learn safe, high-reward, and adaptable policies
- **Key idea:** model the distribution of states and actions and uncover the relationships between behaviors and constraint thresholds from the offline dataset
- **Constraint-conditioned actor-critic (CCAC)**



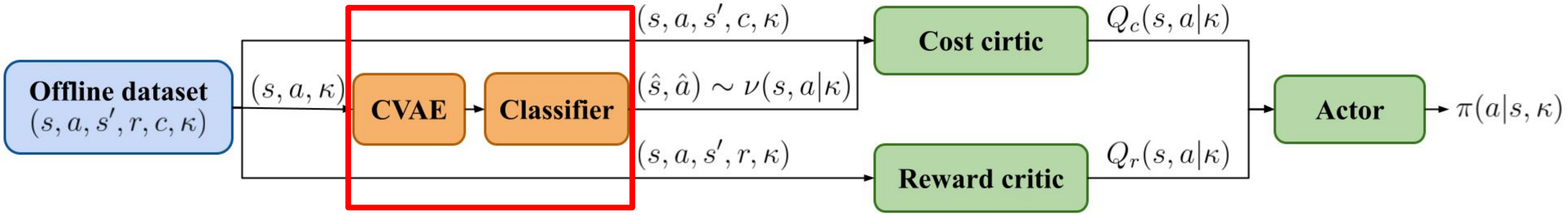
Overview of CCAC

Varying Constraint Thresholds



- Define the varying constraint threshold as **cost budget** κ_t
  - Set the cumulative cost of each trajectory as its initial cost budget: $\kappa_1 = \sum_{t=1}^{T} c_t$
  - Updated based on incurred cost: $\kappa_{t+1} = \kappa_t - c_t$
- Given a cost budget κ, a state-action pair (s, a) is considered **out-of-distribution (OOD)** if taking the action a at the state s will lead to exceeding the cost budget κ

OOD Distribution Modeling



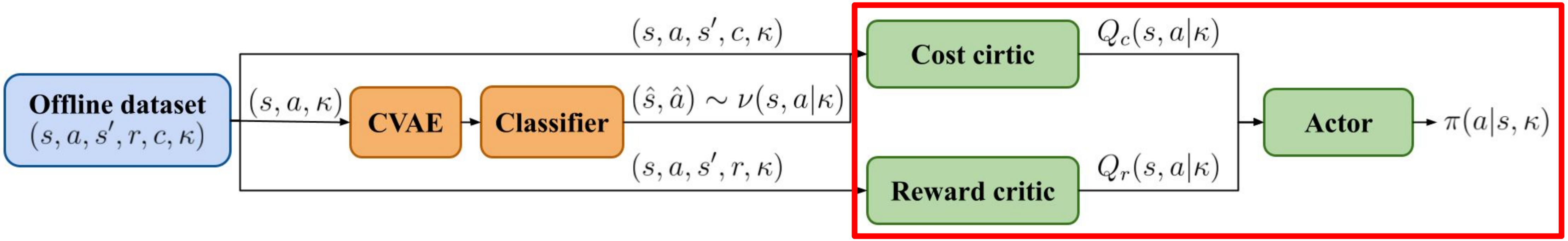
- Constraint-conditioned variational autoencoder (CVAE):

$$\max_{p,q} \mathbb{E}_{s,a,\kappa\sim\mathcal{D}} \left[\mathbb{E}_{z\sim q(z|s,a,\kappa)} \left[p(s, a|z, \kappa) \right] - \beta D_{KL} \left(q(z|s, a, \kappa) || p(z|\kappa) \right) \right]$$

where $q(z|s, a, \kappa)$ is the encoder and $p(s, a|z, \kappa)$ is the decoder

- Classifier: $\min_h \mathbb{E}_{s,a,\kappa\sim\mathcal{D}} \left[-y \log\left(h(s, a|\kappa)\right) - (1 - y) \log\left(1 - h(s, a|\kappa)\right) \right]$
- **The OOD distribution** $\nu(s, a|\kappa)$ **is proportional to** $\mathbb{E}_{z\sim p(z|\kappa)}[p(s, a|z, \kappa)h(s, a|\kappa)]$

Constraint-Conditioned Actor-Critic



- Constraint-conditioned cost critic **(overestimation of OOD state-action pairs)**:

$$\min_{Q_c} \mathbb{E}_{s,a,\kappa\sim\mathcal{D}}\left[(Q_c(s,a|\kappa) - \mathcal{T}^\pi Q_c(s,a|\kappa))^2\right] \text{ , s.t. } \mathbb{E}_{s,a\sim\nu,\kappa\sim\mathcal{D}}[Q_c(s,a|\kappa)] \geq \epsilon$$

- Constraint-conditioned reward critic:

$$\min_{Q_r} \mathbb{E}_{s,a,\kappa\sim\mathcal{D}}\left[(Q_r(s,a|\kappa) - \mathcal{T}^\pi Q_r(s,a|\kappa))^2\right]$$

- Constraint-conditioned actor:

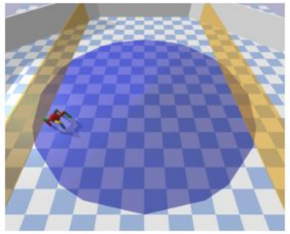
$$\max_\pi \mathbb{E}_{s,\kappa\sim\mathcal{D},a\sim\pi(a|s,\kappa)}\left[Q_r(s,a|\kappa)\right] \text{ , s.t. } \mathbb{E}_{s\sim\mathcal{D},a\sim\pi(a|s,\kappa)}\left[Q_c(s,a|\kappa)\right] \leq \kappa, \forall\kappa \in \mathcal{D}$$

Experimental Settings

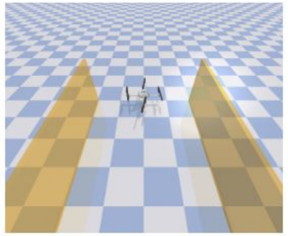
- **Environments** (BulletSafetyGym[1], SafetyGymnasium[2])
  - **Three tasks: Run, Circle, and Velocity.**

    Run/Velocity: run as fast as it can within the speed limit between two boundaries

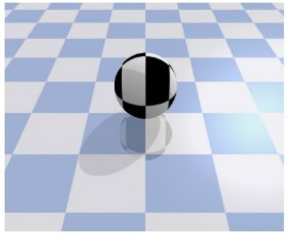
    Circle: run in a circle but are constrained within a safe region
  - **Multiple robots:** Ant, Ball, Car, Drone, Hopper, HalfCheetah



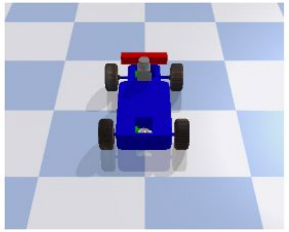
Circle task



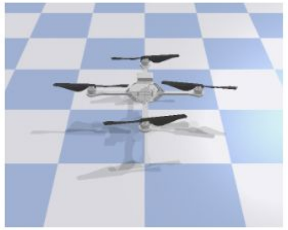
Run task



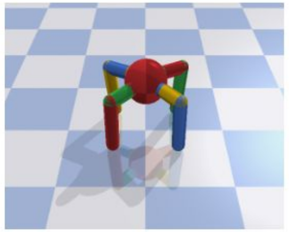
Ball



Car



Drone

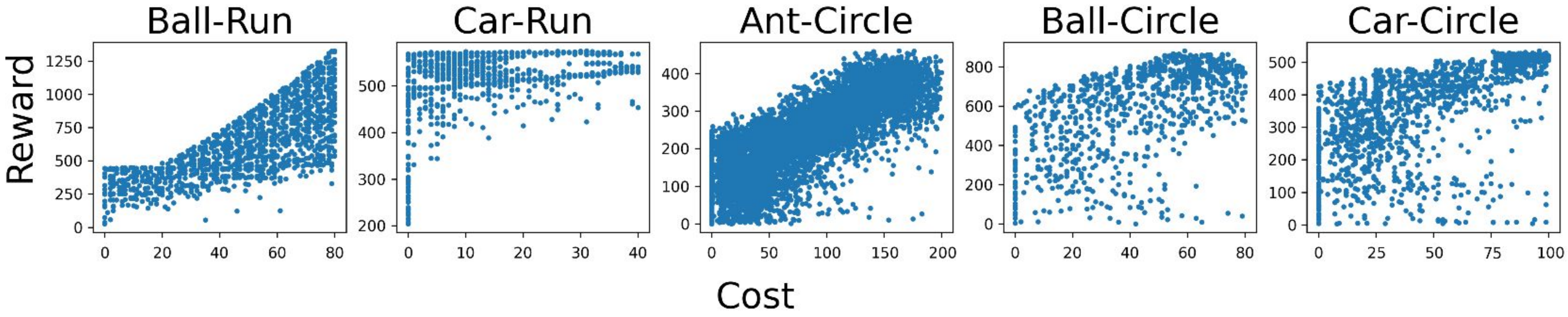


Ant

[1] Gronauer, Sven. "Bullet-safety-gym: A framework for constrained reinforcement learning." (2022).
[2] Ji, Jiaming, et al. "Safety gymnasium: A unified safe reinforcement learning benchmark." *Advances in Neural Information Processing Systems* 36 (2023).

Experimental Settings

- **Offline datasets:** DSRL[3]
  - Reward v.s. cost plots, each dot represents a trajectory



[3] Liu, Zuxin, et al. "Datasets and benchmarks for offline safe reinforcement learning." *arXiv preprint arXiv:2306.09303* (2023).

Results

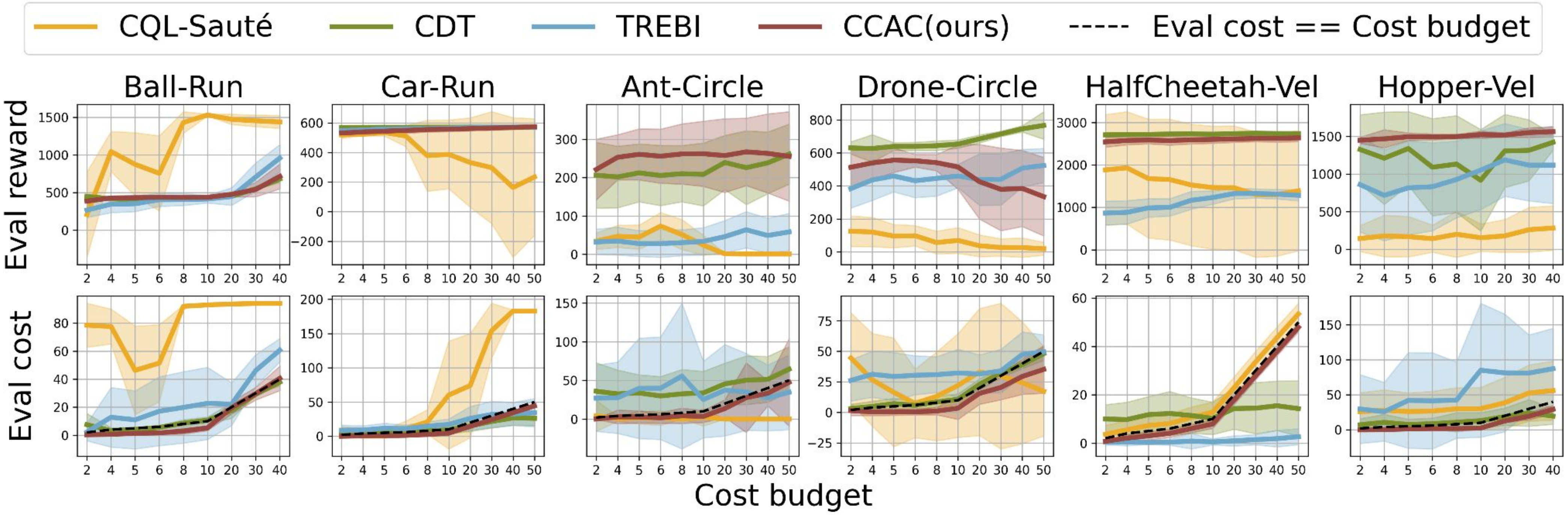
- CCAC can learn **safe and high-reward policies** from offline datasets

Tasks	Metric	CQL-Sauté	BCQ-Lag	BEAR-Lag	CPQ	COptiDICE	VOCE	CDT	TREBI	FISOR	CCAC(ours)
Run	Reward ↑	0.55±0.44	1.51±0.91	1.88±1.03	0.95±0.03	1.38±0.5	1.65±1.07	0.99±0.02	0.82±0.19	**0.74±0.08**	**0.96±0.03**
	Cost ↓	5.23±6.26	19.85±12.76	23.56±7.99	1.92±2.66	6.27±6.29	9.35±8.95	1.34±0.74	2.12±2.36	**0.54±1.90**	**0.23±0.27**
Circle	Reward ↑	0.4±0.33	1.11±0.31	1.13±0.21	0.64±0.42	0.63±0.25	0.06±0.11	0.91±0.17	0.43±0.25	**0.27±0.18**	**0.79±0.24**
	Cost ↓	6.88±9.15	15.61±6.1	13.19±6.18	3.73±6.43	9.82±10.57	10.05±15.53	2.58±2.49	3.53±8.26	**0.17±0.76**	**0.17±0.79**
Velocity	Reward ↑	0.44±0.43	0.78±0.37	0.18±0.8	0.41±1.15	0.65±0.36	**-0.41±0.44**	0.88±0.24	0.45±0.24	**0.47±0.26**	**0.86±0.2**
	Cost ↓	2.08±3.23	27.26±29.75	20.1±25.76	35.18±44.66	6.94±7.22	**0.0±0.0**	2.91±2.87	1.07±2.52	**0.08±0.27**	**0.38±0.2**
Average	Reward ↑	0.45±0.4	1.09±0.58	0.98±0.93	0.63±0.75	0.8±0.47	0.26±0.96	0.91±0.19	0.51±0.28	**0.44±0.27**	**0.85±0.21**
	Cost ↓	4.91±7.33	20.44±19.33	17.8±16.46	13.81±30.22	8.07±8.85	6.54±12.1	2.41±2.45	2.41±6.0	**0.22±1.05**	**0.25±0.56**

Bold: safe agents with (normalized) cost smaller than 1. **Blue:** safe agents with the highest reward

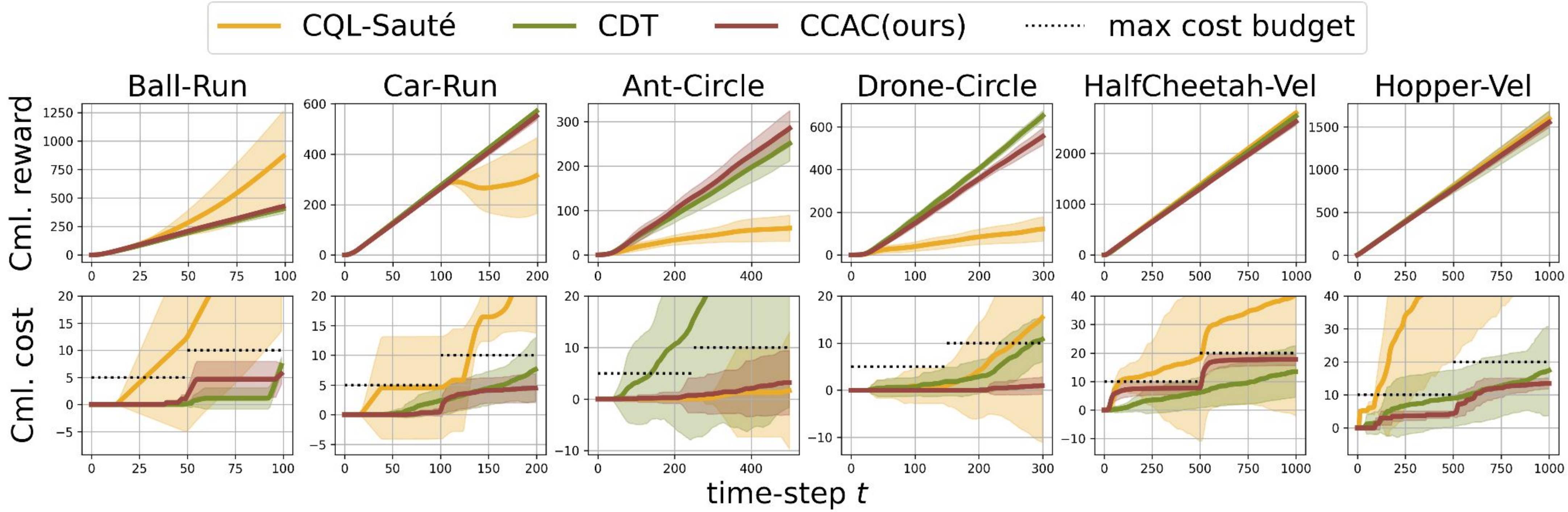
Results

- CCAC can achieve **zero-shot adaptation to different (even unseen) cost thresholds**



Results

- CCAC can achieve **zero-shot adaptation to dynamic cost thresholds**



BOSTON UNIVERSITY

Summary

- A novel OSRL method CCAC:
  - Handle varying constraint thresholds
  - Model the distribution of states and actions based on constraint thresholds
  - Overestimate the cost critics of OOD state-action pairs
- CCAC is able to learn **safe**, **high-reward** and **adaptable** policies
- Code is available at https://github.com/BU-DEPEND-Lab/CCAC