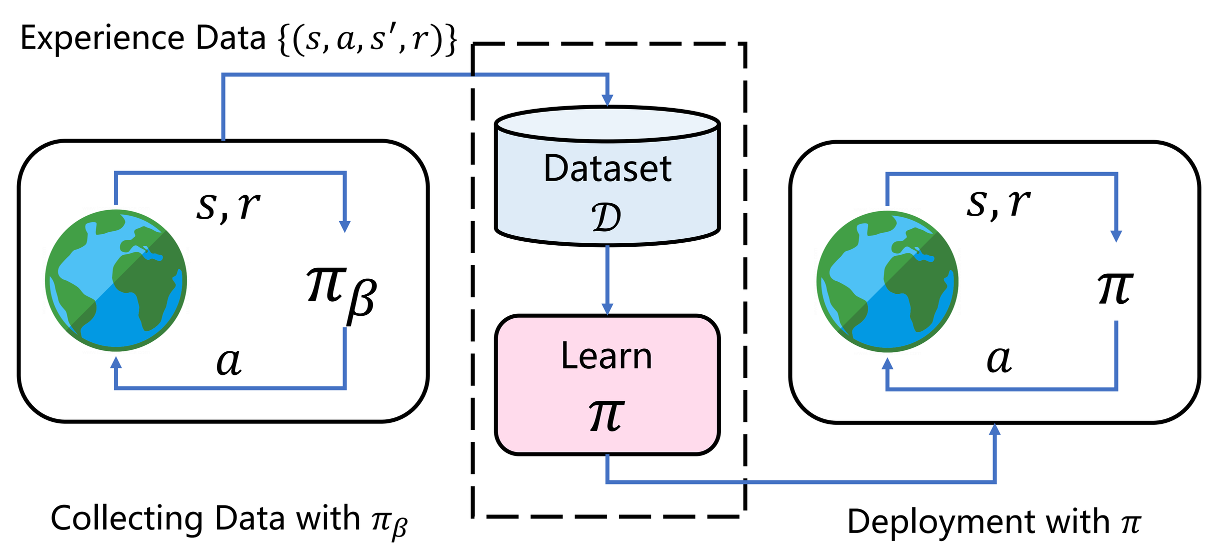
Supported Policy Optimization for Offline Reinforcement Learning

Jialong Wu, Haixu Wu, Zihan Qiu, Jianmin Wang, Mingsheng Long

NEURAL INFORMATION PROCESSING SYSTEMS

Introduction

- **Offline reinforcement learning** eliminates the need to interact with the live environment, which is always expensive or risky in practical scenarios.
  - Autonomous driving, healthcare, education, advertising, etc.



- **Challenges**: Extrapolation error of Q estimation queried by OOD actions
- **Support constraint**: $\pi_\phi(s) \in \{a : \pi_\beta(a|s) > \epsilon\} \quad \forall s$
- **Policy constraint methods**: Parameterization vs Regularization

	Parameterization	Regularization
Pros	Direct constraint	Pluggable design
Cons	- Extra inference costs - Implementation difficulties - Complicates transfer of design techniques	Divergence-based regularization may mismatch density-based formalization of support constraint

- **Contributions**:
  - Regularization term which directly regularizes the behavior density of actions taken by the learned policy
  - Supported Policy OpTimization (SPOT), a practical algorithm with a neural VAE-based density estimator
  - Strong experimental results for offline RL and online fine-tuning on standard offline RL benchmarks

Support Constraint via Behavior Density

- **Policy optimization with behavior density as constraint**

$$\max_\phi \ \mathbb{E}_{s\sim\mathcal{D}}\left[Q_\theta(s, \pi_\phi(s))\right] \quad \text{s.t.} \ \min_s \ \log \pi_\beta(\pi_\phi(s)|s) > \hat{\epsilon}, \tag{1}$$

- **Heuristic approximation**: widely adopted by both online RL and offline RL w.r.t. constrained policy optimization.

$$\max_\phi \ \mathbb{E}_{s\sim\mathcal{D}}\left[Q_\theta(s, \pi_\phi(s))\right] \quad \text{s.t.} \ \mathbb{E}_{s\sim\mathcal{D}}\left[\log \pi_\beta(\pi_\phi(s)|s)\right] > \hat{\epsilon}. \tag{2}$$

- **Policy learning objective**: a pluggable regularization applied directly to behavior density

$$J_\pi(\phi) = \mathbb{E}_{s\sim\mathcal{D}}\left[-Q_\theta\left(s, \pi_\phi(s)\right) - \lambda \log \pi_\beta\left(\pi_\phi(s)|s\right)\right], \tag{3}$$

Explicit Estimation of Behavior Density

- **Modeled by Conditional Variational Auto-Encoder (CVAE)**

$$\pi_\beta(a|s) \approx p_\psi(a|s) = \int p_\psi(a|z, s) p(z|s) \mathrm{d}z \tag{4}$$

- **Optimization with evidence lower bound (ELBO)**

$$\begin{aligned}\log p_\psi(a|s) &\geq \mathbb{E}_{q_\varphi(z|a,s)}\left[\log \frac{p_\psi(a, z|s)}{q_\varphi(z|a, s)}\right] \\ &= \mathbb{E}_{q_\varphi(z|a,s)}\left[\log p_\psi(a|z, s)\right] - \mathrm{KL}\left[q_\varphi(z|a, s) \| p(z|s)\right] \\ &\stackrel{\text{def}}{=} -\mathcal{L}_{\mathrm{ELBO}}(s, a; \varphi, \psi).\end{aligned} \tag{5}$$

- **Density estimation with importance sampling** (Rezende et al., 2014)

$$\begin{aligned}\log p_\psi(a|s) &= \log \mathbb{E}_{q_\varphi(z|a,s)}\left[\frac{p_\psi(a, z|s)}{q_\varphi(z|a, s)}\right] \\ &\approx \mathbb{E}_{z^{(l)}\sim q_\varphi(z|a,s)}\left[\log \frac{1}{L}\sum_{l=1}^{L} \frac{p_\psi(a, z^{(l)}|s)}{q_\varphi(z^{(l)}|a, s)}\right] \\ &\stackrel{\text{def}}{=} \widehat{\log \pi_\beta}(a|s; \varphi, \psi, L).\end{aligned} \tag{6}$$

- **Policy learning objective with density estimator**

$$J_\pi(\phi) = \mathbb{E}_{s\sim\mathcal{D}}\left[-Q_\theta(s, \pi_\phi(s)) - \lambda \widehat{\log \pi_\beta}(\pi_\phi(s)|s; \varphi, \psi, L)\right]. \tag{7}$$

Overall Algorithm: Supported Policy Optimization

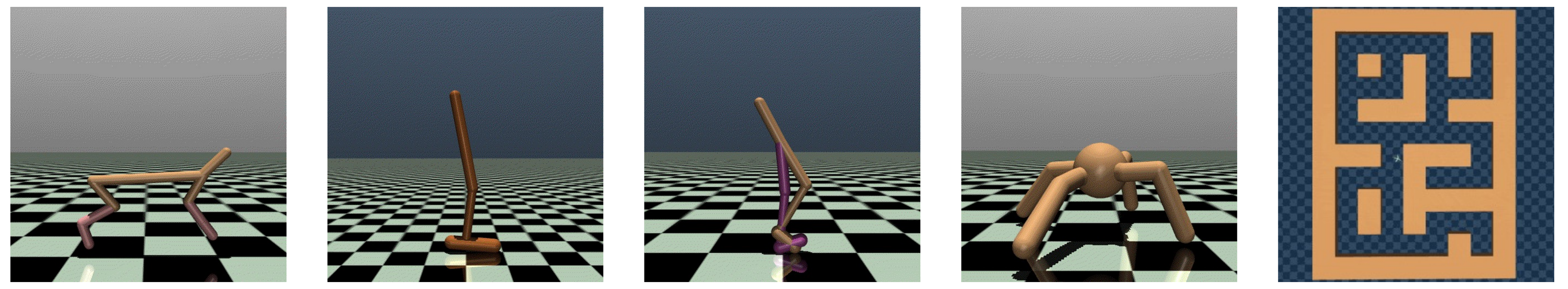
Algorithm 1 Supported Policy Optimization (SPOT)

Input: Dataset $\mathcal{D} = \{(s, a, r, s')\}$
// **Density Estimation with VAE**
Initialize VAE with parameters ψ and φ
for $t = 1$ **to** T_1 **do**
  Sample minibatch of transitions $(s, a) \sim \mathcal{D}$
  Update ψ, φ minimizing $\mathcal{L}_{\mathrm{ELBO}}(s, a; \varphi, \psi)$ in Eq. (5)
end for
// **Actor-Critic with Plugged Regularization**
Initialize the policy network π_ϕ, critic network Q_θ and target network $Q_{\bar{\theta}}$ with $\bar{\theta} \leftarrow \theta$
for $t = 1$ **to** T_2 **do**
  Sample minibatch of transitions $(s, a, r, s') \sim \mathcal{D}$
  Update θ minimizing $J_Q(\theta) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\left[Q_\theta(s, a) - r - \gamma Q_{\bar{\theta}}(s', \pi_\phi(s'))\right]^2$
  Update ϕ minimizing $J_\pi(\phi)$ in Eq. (7)
  Update target network: $\bar{\theta} \leftarrow \tau\theta + (1 - \tau)\bar{\theta}$
end for =0

Practical Implementation

- **Base algorithm**: TD3
- **Q normalization**: following TD3+BC (Fujimoto & Gu, 2021)
- **Simpler density estimator**: empirically find no further improvement with larger L compared to $L = 1$ (ELBO estimator)

Comparisons on Offline RL Benchmarks



HalfCheetah   Hopper   Walker2d   Ant   Large maze

- **D4RL-Gym-MuJoCo**: SPOT demonstrates the state-of-the-art performance, especially on suboptimal datasets.

Table: Performance of SPOT and prior methods on Gym-MuJoCo tasks*.

	BC	AWAC	DT	Onestep	TD3+BC	CQL	IQL	**SPOT**
HalfCheetah-m-e-v2	55.2	42.8	86.8	**93.4**	90.7	91.6	86.7	86.9±4.3
Hopper-m-e-v2	52.5	55.8	**107.6**	103.3	98.0	105.4	91.5	99.3±7.1
Walker-m-e-v2	107.5	74.5	108.1	**113.0**	110.1	108.8	109.6	112.0±0.5
HalfCheetah-m-v2	42.6	43.5	42.6	48.4	48.3	44.0	47.4	**58.4**±1.0
Hopper-m-v2	52.9	57.0	67.6	59.6	59.3	58.5	66.2	**86.0**±8.7
Walker-m-v2	75.3	72.4	74.0	81.8	83.7	72.5	78.3	**86.4**±2.7
HalfCheetah-m-r-v2	36.6	40.5	36.6	38.1	44.6	45.5	44.2	**52.2**±1.2
Hopper-m-r-v2	18.1	37.2	82.7	97.5	60.9	95.0	94.7	**100.2**±1.9
Walker-m-r-v2	26.0	27.0	66.6	49.5	81.8	77.2	73.8	**91.6**±2.8
Gym-MuJoCo total	466.7	450.7	672.6	684.6	677.4	698.5	692.4	**773.0**±30.2

* m = medium, m-r = medium-replay, m-e = medium-expert.

- **D4RL-AntMaze**: SPOT obtains strong performance with a simple design.

Table: Performance of SPOT and prior methods on AntMaze tasks.

	BCQ	BEAR	BC	DT	TD3+BC	PLAS	CQL	IQL	**SPOT**
umaze-v2	78.9	73.0	49.2	54.2	73.0	62.0	82.6	89.6	**93.5**±2.4
umaze-diverse-v2	55.0	61.0	41.8	41.2	47.0	45.4	10.2	**65.6**	40.7±5.1
medium-play-v2	0.0	0.0	0.4	0.0	0.0	31.4	59.0	**76.4**	74.7±4.6
medium-diverse-v2	0.0	8.0	0.2	0.0	0.2	20.6	46.6	72.8	**79.1**±5.6
large-play-v2	6.7	0.0	0.0	0.0	0.0	2.2	16.4	**42.0**	35.3±8.3
large-diverse-v2	2.2	0.0	0.0	0.0	0.0	3.0	3.2	**46.0**	36.3±13.7
AntMaze total	142.8	142.0	91.6	95.4	120.2	164.6	218.0	**392.4**	359.6±39.7

Online Fine-tuning after Offline RL

- **Well-suited for online fine-tuning**: A minimal gap between offline RL and well-established online RL method.
- **D4RL-AntMaze**: SPOT achieves superior online fine-tuning performance over the state-of-the-art method.

Table: Online fine-tuning results on AntMaze tasks*.

	IQL	**SPOT**
umaze-v2	85.4 → 96.2	**93.2** → **99.2** (+3.0)
umaze-diverse-v2	**70.8** → 62.2	41.6 → **96.0** (+33.8)
medium-play-v2	68.6 → 89.8	**75.2** → **97.4** (+7.6)
medium-diverse-v2	**73.4** → 90.2	73.0 → **96.2** (+6.0)
large-play-v2	40.0 → 78.6	**40.8** → **89.4** (+10.8)
large-diverse-v2	40.4 → 73.4	**44.0** → **90.8** (+17.4)
AntMaze total	**378.6** → 490.4	367.8 → **569.0** (+78.6)

* showing initial performance after offline RL and performance after 1M steps of online RL.

School of Software - BNRist - Tsinghua University - China

Code: `https://github.com/thuml/SPOT`

Mail: wujialong0229@gmail.com