Offline Reinforcement Learning

Pu Yang
School of Mathematical Sciences, Peking University

2021.10.13

Table of contents

Markov Decision Processes (MDP) $\mathcal{M} =< \mathcal{S}, \mathcal{A}, P, R, \gamma >$[1]

- $\mathcal{S}$ - a set of states; $s \in \mathcal{S}$ - a state
- $\mathcal{A}$ - a set of actions; $a \in \mathcal{A}$ - an action
- P - transition probability function
- R - reward function
- γ - discounting factor for future rewards

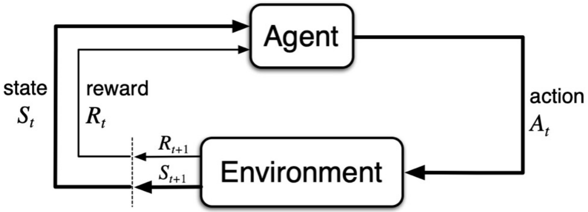


Figure 1: The agent–environment interaction in a Markov decision process.

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

Bellman Equation

- Value function: $V_\pi(s) = \mathbb{E}_\pi \left[\sum_{k=1}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right]$
- Q function: $Q_\pi(s,a) = \mathbb{E}_\pi \left[\sum_{k=1}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s, a_t = a \right]$

Bellman equation

$$V_\pi(s) = \mathbb{E}_{a\sim\pi}\left[Q(s_t,a) \mid s_t = s\right] = \mathbb{E}_\pi\left[r_{t+1} + \gamma V_\pi(s_{t+1}) \mid s_t = s\right]$$

$$\begin{aligned} Q_\pi(s,a) &= \mathbb{E}_\pi\left[r_{t+1} + \gamma V_\pi(s_{t+1}) \mid s_t = s, a_t = a\right] \\ &= \mathbb{E}_\pi\left[r_{t+1} + \gamma \mathbb{E}_{a\sim\pi} Q(s_{t+1},a) \mid s_t = s, a_t = a\right] \end{aligned}$$

Bellman optimality equation

$$V^\star(s) = \max_{a\in\mathbb{A}(s)} Q_{\pi^\star}(s,a) = \max_a \mathrm{E}\left[r_{t+1} + \gamma V^\star(s_{t+1}) \mid s_t = s, a_t = a\right]$$

$$Q^\star(s,a) = \mathbb{E}\left[R_{t+1} + \gamma \max_{a'} Q^\star(s_{t+1},a') \mid s_t = s, a_t = a\right]$$

on/off-line RL, on/off-policy RL

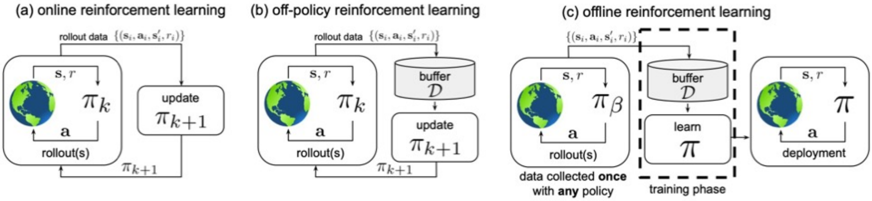


Figure 2: (a) Pictorial illustration of classic online reinforcement learning, (b) classic off-policy reinforcement learning, and (c) offline reinforcement learning.[1]

- on/off-line: how to use samples
- on/pff-policy: how to generate samples
  - on-policy: evaluate or improve the policy that is used to make decisions
  - off-policy: evaluate or improve a policy different from that used to generate the data
- off-policy to off-line (not feasibel in practice)

[1] Sergey Levine et al. "Offline reinforcement learning: Tutorial, review, and perspectives on open problems". In: *arXiv preprint arXiv:2005.01643* (2020).

On policy: state-action-reward-state'-action' (SARSA)

- SARSA Algorithm

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(\mathit{terminal}, \cdot) = 0$

Loop for each episode:
  Initialize S
  Choose A from S using policy derived from Q (e.g., ε-greedy)
  Loop for each step of episode:
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ε-greedy)
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma Q(S', A') - Q(S, A)\big]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  until S is terminal

  - behavior policy: ϵ-greedy
  - evaluation policy: ϵ-greedy

Off-policy: Q-learning

- Q-learning Algorithm

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    Choose A from S using policy derived from Q (e.g., ε-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
    $S \leftarrow S'$
  until S is terminal

  - behavior policy: ϵ-greedy
  - evaluation policy: greedy

Why Offline RL?

- data collection is expensive
  - robotics[1][2][3]
  - educational agents
  - healthcare[4][5]
- dangerous
  - autonomous driving[6]
  - healthcare
- the domain is complex and effective generalization requires large data sets
  - advertising and recommender systems(?)

[1] Sergey Levine et al. “Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection”. In: *The International Journal of Robotics Research* 37.4-5 (2018), pp. 421–436.

[2] Dmitry Kalashnikov et al. “Scalable deep reinforcement learning for vision-based robotic manipulation”. In: *Conference on Robot Learning*. PMLR. 2018, pp. 651–673.

[3] Andy Zeng et al. “Learning synergies between pushing and grasping with self-supervised deep reinforcement learning”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 4238–4245.

[4] Omer Gottesman et al. “Evaluating reinforcement learning algorithms in observational health settings”. In: *arXiv preprint arXiv:1805.12298* (2018).

[5] Omer Gottesman et al. “Guidelines for reinforcement learning in healthcare”. In: *Nature medicine* 25.1 (2019), pp. 16–18.

[6] Ekim Yurtsever et al. “A survey of autonomous driving: Common practices and emerging technologies”. In: *IEEE Access* 8 (2020), pp. 58443–58469.

What are the difficulties?

- No exploration (have no idea on that)
- Hard to evaluate a policy
  - off-policy evaluation (OPE)
- distributional shift
  - counterfactual queries
  - want something different and better
- require too many samples
  - Sample efficiency

Basic setting of Off-policy evaluation

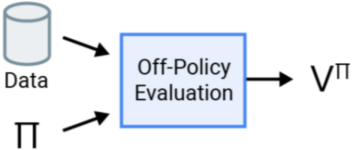
- an MDP $\mathcal{M} =< \mathcal{S}, \mathcal{A}, P, R, \gamma >$, where P and R is unknown
- a historical data $\mathcal{D} = \{\tau^i\}_1^N$, generated by a **behavior policy** π_b, where

$$\tau^i = \{s_0^i, a_0^i, r_0^i, s_1^i, a_1^i, r_1^i, \cdots s_{T-1}^i, a_{T-1}^i, r_{T-1}^i\}$$

- a desired **evaluation policy** π_e
- the OPE problem is to estimate the value $V(\pi_e)$, defined as:

$$V(\pi_e) = \mathbb{E}_{x \sim d_0}\left[\sum_{t=0}^{T-1} \gamma^t r_t \mid s_0 = s\right]$$

where $a_t \sim \pi_e(\cdot \mid s_t)$, $x_{t+1} \sim P(\cdot \mid s_t, a_t)$, $r_t \sim R(s_t, a_T)$, and d_0 is the initial state distribution.



Off-policy evaluation

- Direct Methods
- Importance Sampling (also called Inverse Propensity Scoring)

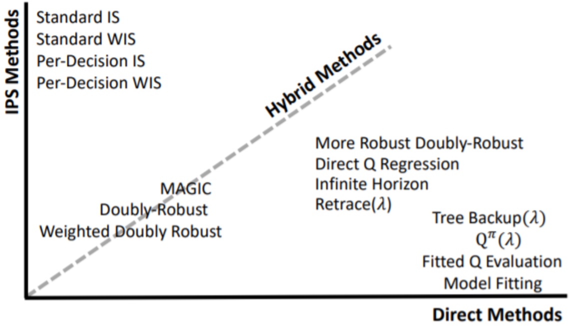


Figure 3: Categorization of OPE methods.[1]

[1] Cameron Voloshin et al. "Empirical study of off-policy policy evaluation for reinforcement learning". In: *arXiv preprint arXiv:1911.06854* (2019).

Direct Methods

- Model-based
  - Approximation Model: directly fit the transition P and reward R
  - also suffer from distributional shift
- Model-free
  - Approximate Q function with $\hat{Q}(\cdot;\theta)$, parametrized by θ, then

$$V(\pi_e) = \frac{1}{N}\sum_{i=1}^{N}\sum_{a\in\mathcal{A}}\pi_e(a \mid s)\hat{Q}(s_0^i, a; \theta)$$

  - example: MRDR[1], FQE[2], ...

[1] Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. "More robust doubly robust off-policy evaluation". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 1447–1456.

[2] Hoang Le, Cameron Voloshin, and Yisong Yue. "Batch policy learning under constraints". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 3703–3712.

Fitted Q Evaluation (FQE)

Given a Dataset $\mathcal{D} = \{s_t, a_t, s'_t, r_t\}$ and a policy π to be evaluated.
Fitted Q Evaluation (FQE) learns a sequence of estimator $\hat{Q}(\cdot;\theta) = \lim_{k\to\infty} \hat{Q}_k$

- Step 1: Initialization. $\hat{Q}_0 = 0$ (or randomly)
- Step 2:

$$y_t^i = r_t^i + \gamma \mathbb{E}_{\pi_e} \hat{Q}_{k-1}(s_{t+1}^i, \cdot; \theta)$$

- Step 3: build a training dataset $\mathcal{D}_k = \{(s_i, a_i), y_i\}$
- Step 4:

$$\hat{Q}_k = \min_\theta \frac{1}{N} \sum_{i=1}^{N} \sum_{t=0}^{T-1} (\hat{Q}_{k-1}(s_t^i, a_t^i; \theta) - y_t^i)^2,$$

  then back to step 2.

Also theoretical guarantees: the generalization error is bounded!

Importance Sampling (IS)

$$p_\pi(\tau) = d_0(s_0) \prod_{t=0}^{T} \pi(a_t \mid s_t) T(s_{t+1} \mid s_t, a_t)$$

$$\begin{aligned} J(\pi_e) &= \mathbb{E}_{\tau \sim p_{\pi_e}} \left[\sum_{t=0}^{T} \gamma^t R(s_t, a_t) \right] \\ &= \mathbb{E}_{\tau \sim p_{\pi_b}} \left[\frac{\pi_e(\tau)}{\pi_b(\tau)} \sum_{t=0}^{T} \gamma^t R(s_t, a_t) \right] \\ &= \mathbb{E}_{\tau \sim p_{\pi_b}} \left[(\prod_{t=0}^{T} \frac{\pi_e(a_t \mid s_t)}{\pi_b(a_t \mid s_t)}) \sum_{t=0}^{T} \gamma^t R(s_t, a_t) \right] \\ &\approx \sum_{i=1}^{n} w_T^i \sum_{t=0}^{T} \gamma^t r_t^i \end{aligned}$$

where $w_t^i = \frac{1}{n} \prod_{t'=0}^{t} \frac{\pi_e(a_{t'}^i \mid s_{t'}^i)}{\pi_b(a_{t'}^i \mid s_{t'}^i)}$

Curse of horizon

- consistent unbiased, but have high variance (growing exponentially with T)
- improvement
  - Weighted Importance Sampling

$$J(\pi_e) \approx \frac{\sum_{i=1}^{n} w_H^i \sum_{t=0}^{T} \gamma^t r_t^i}{\sum_{i=1}^{n} w_H^i}$$

    which is biased, but can have much lower variance.
  - Per-Decision Importance Sampling[1]

$$J(\pi_e) = \mathbb{E}_{\tau \sim p_{\pi_b}}\left[\sum_{t=0}^{T}\left(\prod_{t'=0}^{t} \frac{\pi_e(a_{t'} \mid s_{t'})}{\pi_b(a_{t'} \mid s_{t'})}\right)\gamma^t R(s_t, a_t)\right] \approx \frac{1}{n}\sum_{i=1}^{n}\sum_{t=0}^{T} w_t^i \gamma^t r_t^i$$

    which is unbiased.

[1] Doina Precup. "Eligibility traces for off-policy policy evaluation". In: *Computer Science Department Faculty Publication Series* (2000), p. 80.

Doubly robust estimator[1][2]

In fact, DR estimator is a mixed strategy

$$J(\pi_e) = \sum_{i=1}^{n} \sum_{t=0}^{T} \gamma^t (w_t^i (r_t^i - \hat{Q}^{\pi_e}(s_t, a_s)) - w_{t-1}^i \mathbb{E}_{a \sim \pi_e(a|s_t)} [\hat{Q}^{\pi_e}(s_t, a)])$$

which is unbiased if either π_b is known or if the model is correct.

It can be proved that the DR has lower variance than importance sampling.

[1] Nan Jiang and Lihong Li. "Doubly robust off-policy value evaluation for reinforcement learning". In: *International Conference on Machine Learning*. PMLR. 2016, pp. 652–661.

[2] Philip Thomas and Emma Brunskill. "Data-efficient off-policy policy evaluation for reinforcement learning". In: *International Conference on Machine Learning*. PMLR. 2016, pp. 2139–2148.

Doubly robust estimator

Scalabilities:

- fit Q with prior knowledge
- trade off bias and variance

Marginalized Importance Sampling[1]

Estimate the **state-marginal importance ratio** $\rho^{\pi_e}(s) = \frac{d^{\pi_e}(s)}{d^{\pi_b}(s)}$.

Notation:

- $d_t^{\pi}(s_t)$: the state marginal of π at t
- $d^{\pi}(s) = \frac{1}{1-\gamma}\sum_{t=0}^{T}\gamma^t d_t^{\pi}(s_t)$: the normalized discounted state distribution
- $d^{\pi}(s,a) = d^{\pi}(s)\pi(a|s)$

$$
\begin{aligned}
J(\pi_e) &= \mathbb{E}_{(s,a)\sim d^{\pi_e}, r\sim R(s,a)}(r) \\
&= \mathbb{E}_{(s,a)\sim d^{\pi_b}, r\sim R(s,a)}\left[\frac{d^{\pi_e}(s,a)}{d^{\pi_b}(s,a)} r\right] \\
&= \mathbb{E}_{(s,a)\sim d^{\pi_b}, r\sim R(s,a)}\left[\frac{d^{\pi_e}(s)\pi_e(a|s)}{d^{\pi_b}(s)\pi_b(a|s)} r\right]
\end{aligned}
$$

[1] Ruiyi Zhang et al. "Gendice: Generalized offline estimation of stationary values". In: *arXiv preprint arXiv:2002.09072* (2020).

Marginalized Importance Sampling

"Forward" Bellman equation:

$$\underbrace{d^{\pi_b}\left(\mathbf{s}'\right)\rho^{\pi_e}\left(\mathbf{s}'\right)}_{:=(d^{\pi_b}\circ\rho^{\pi_e})(\mathbf{s}')} = \underbrace{(1-\gamma)d_0\left(\mathbf{s}'\right) + \gamma\sum_{\mathbf{s},\mathbf{a}} d^{\pi_b}(\mathbf{s})\rho^{\pi_e}(\mathbf{s})\pi_e(\mathbf{a}\mid\mathbf{s})P\left(\mathbf{s}'\mid\mathbf{s},\mathbf{a}\right)}_{:=\left(\overline{\mathcal{B}}^{\pi_e}\circ\rho^{\pi_e}\right)(\mathbf{s}')}$$

There are several techniques to solve this equation, for example[1]:

$$\hat{\rho}^{\pi_e}\left(\mathbf{s}'\right) \leftarrow \hat{\rho}^{\pi_e}\left(\mathbf{s}'\right) + \alpha\left[(1-\gamma) + \gamma\frac{\pi_e(\mathbf{a}\mid\mathbf{s})}{\pi_b(\mathbf{a}\mid\mathbf{s})}\hat{\rho}^{\pi_e}(\mathbf{s}) - \hat{\rho}^{\pi_e}\left(\mathbf{s}'\right)\right]$$

where $s, a, s' \in \mathcal{D}$.

[1] Carles Gelada and Marc G Bellemare. "Off-policy deep reinforcement learning by bootstrapping the covariate shift". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01. 2019, pp. 3647–3655.

Limitations of importance sampling

The importance weights will become degenerate when π_b is too different from π_e!

- the suboptimality of the behavior policy
- the dimension of the state and action space
- curse of horizon

Distributional shift at test time

- the test environment (state) differs from the training environment

Solutions:

- theoretical bounds for $D_{\mathrm{KL}}(d^{\pi}(s)||d^{\pi_b}(s))$[1]
- detect different environment
- first offline learning, then online fine-tuning[2]

[1] John Schulman et al. “Trust region policy optimization”. In: *International conference on machine learning*. PMLR. 2015, pp. 1889–1897.

[2] Ashvin Nair et al. “Accelerating online reinforcement learning with offline datasets”. In: *arXiv preprint arXiv:2006.09359* (2020).

Distributional shift at training time

- Environments are the same, but the training is affected by action distributional shift
- Formally, $\pi_e(a \mid s)$ may differs substantially from $\pi_b(a \mid s)$

Model-free action distributional shift

- Learned Q-function erroneously produces excessively large values.
- Actor-Critic method:

$$Q^\pi(s_t, a_t) \leftarrow Q^\pi(s_t, a_t) + \alpha[r(s_t, a_t) + \gamma \max_{a_{t+1}} Q^\pi(s_{t+1}, a_{t+1}) - Q^\pi(s_t, a_t)]$$

then evaluate policy:

$$\pi(a|s) = \arg\max \mathbb{E}_{a \sim \pi(a|s)}[Q^\pi(s, a)]$$

iteratively.

- π may be biased towards out-of-distribution actions with erroneously high Q-values

Q-function will be overestimated

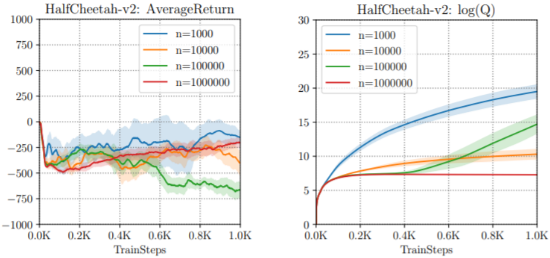


Figure 1: Performance of SAC on HalfCheetah-v2 (return (left) and log Q-values (right)) with off-policy expert data w.r.t. number of training samples (n). Note the large discrepancy between returns (which are negative) and log Q-values (which have large positive values), which is not solved with additional samples.

1

[1] Aviral Kumar et al. “Stabilizing off-policy q-learning via bootstrapping error reduction”. In: *arXiv preprint arXiv:1906.00949* (2019).

Policy constraint methods

- Prevent OOD action queries to be Q-function

$$\pi(a|s) = \arg\max_{\pi} \mathbb{E}_{a\sim\pi(a|s)}[Q^{\pi}(s,a)]$$
$$\text{s.t.} \quad D(\pi, \pi_b) \leq \epsilon$$

- Related works instantiate this approach with different choices of D.

  Examples:
  - BEAR-QL[1] uses maximum mean discrepancy (MMD), that is

$$\text{s.t.} \quad \mathbb{E}_{s\sim\mathcal{D}}\left[\mathrm{MMD}(\mathcal{D}(\cdot \mid s), \pi_e(\cdot \mid s))\right] \leq \epsilon$$

  - [2] uses a parametric behavior model and measure distance by KL divergence

$$\theta_{bm} = \arg\max_{\theta} \mathbb{E}_{\tau\sim\mathcal{D}}\left[\sum_{t=1}^{|\tau|} \log \pi_\theta(a_t \mid s_t)\right]$$
$$\text{s.t.} \quad \mathbb{E}_{s\sim\mathcal{D}}\left[\mathrm{KL}(\pi_e(\cdot \mid s), \pi_{bm}(\cdot \mid s))\right] \leq \epsilon$$

[1] Aviral Kumar et al. "Stabilizing off-policy q-learning via bootstrapping error reduction". In: *arXiv preprint arXiv:1906.00949* (2019).

[2] Noah Y Siegel et al. "Keep doing what worked: Behavioral modelling priors for offline reinforcement learning". In: *arXiv preprint arXiv:2002.08396* (2020).

Conservative Q-learning[1]

make a conservative prediction when OOD!

- version 1:

$$\hat{Q}^{k+1} \leftarrow \arg\min_{Q} \alpha \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\pi_e(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})] + \frac{1}{2}\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}\left[\left(Q(\mathbf{s},\mathbf{a}) - \hat{\mathcal{B}}^{\pi}\hat{Q}^{k}(\mathbf{s},\mathbf{a})\right)^2\right]$$

  Theoretically, it can be proved that:
  the resulting Q-function $\hat{Q}^{\pi} = \lim \hat{Q}^{k}$ lower bounds Q^{π} at all (s, a).

[1] Aviral Kumar et al. "Conservative q-learning for offline reinforcement learning". In: *arXiv preprint arXiv:2006.04779* (2020).

Conservative Q-learning

- version 2:

$$\hat{Q}^{k+1} \leftarrow \arg\min_{Q} \alpha \cdot \left(\mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\pi_e(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})] - \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\hat{\pi}_b(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})]\right) + \frac{1}{2}\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[\left(Q(\mathbf{s},\mathbf{a}) - \hat{\mathcal{B}}^{\pi}\hat{Q}^{k}(\mathbf{s},\mathbf{a})\right)^2\right]$$

  - It is a tighter bound then previous result.
  - Intuitively, $\hat{Q}^{\pi}$ is overestimated under $\hat{\pi}_b$, so it may not lower bound point-wise.
  - Theoretically, the value $\hat{V}^{\pi}(s) = \mathbb{E}_{\pi(a|s)}(\hat{Q}^{\pi}(s,a))$ lower bounds V^{π}.

Conservative Q-learning

- version 3 (CQL):

$$\min_Q \max_\mu \alpha \left(\mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})] - \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})] \right) + \frac{1}{2}\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}} \left[\left(Q(\mathbf{s},\mathbf{a}) - \hat{\mathcal{B}}^{\pi_k}\hat{Q}^k(\mathbf{s},\mathbf{a}) \right)^2 \right] + \mathcal{R}(\mu)$$

  - In practice, $\mathcal{R}$ can be a variety of common regularization
  - In theory, when choose $\mathcal{R}$ as the KL divergence of a prior distribution, it can be proved that the value $\hat{V}^\pi$ lower-bounds the true value V^π.

Conservative Q-learning

Algorithm 1 Conservative Q-Learning (both variants)

Initialize Q-function, Q_θ, and optionally a policy, π_ϕ.
for step t in $\{1, \ldots, \text{N}\}$ **do**
    Train the Q-function using G_Q gradient steps on objective from Equation 4
    $\theta_t := \theta_{t-1} - \eta_Q \nabla_\theta \text{CQL}(\mathcal{R})(\theta)$
    (Use $\mathcal{B}^*$ for Q-learning, $\mathcal{B}^{\pi_{\phi_t}}$ for actor-critic)
    (only with actor-critic) Improve policy π_ϕ via G_π gradient steps on ϕ with SAC-style entropy regularization:
    $\phi_t := \phi_{t-1} + \eta_\pi \mathbb{E}_{\mathbf{s}\sim\mathcal{D}, \mathbf{a}\sim\pi_\phi(\cdot|\mathbf{s})}[Q_\theta(\mathbf{s}, \mathbf{a}) - \log \pi_\phi(\mathbf{a}|\mathbf{s})]$
end for

Model-based offline RL

- Intuitively: OOD $\rightarrow$ poorly fit P and R $\rightarrow$ bad policy, bad performance

Theorem 4.1 in[a] (informal)

[a] Michael Janner et al. "When to trust your model: Model-based policy optimization". In: *arXiv preprint arXiv:1906.08253* (2019).

Assume $\epsilon_m = \max_t \mathbb{E}_{d_t^\pi} D_{\mathrm{TV}}(\hat{P}(s_{t+1} \mid s_t, a_t) || P(s_{t+1} \mid s_t, a_t))$ and $\max_s D_{\mathrm{TV}}(\pi_e || \pi_b) \leq \epsilon_\pi$, then

$$J(\pi) \geq \hat{J}(\pi) - \left[\frac{2\gamma r_{\max}(\epsilon_m + 2\epsilon_\pi)}{(1-\gamma)^2} + \frac{4 r_{\max}\epsilon_\pi}{1-\gamma} \right]$$

The first term represents error accumulation due to the distribution shift in the model. The second term represents error accumulation due to the distribution shift in the policy.

Model-based offline RL

- Algorithm
  - combine some CV algorithms (e.g. visual foresight method[1])
  - conservative model (e.g. MoREL[2] and MOPO[3])
    Let the error oracle $u(s, a)$ to estimate the accuracy of the model at the state-action tuple (s, a), for example in MOPO

$$D(\hat{T}(s_{t+1} \mid s_t, a_t) || T(s_{t+1} \mid s_t, a_t)) \leq u(s, a)$$

- Challenges
  - distribution shift
  - high-dimensional observations: the model not fits well
  - long horizons: even small errors will accumulate

1 Frederik Ebert et al. "Visual foresight: Model-based deep reinforcement learning for vision-based robotic control". In: *arXiv preprint arXiv:1812.00568* (2018).

2 Rahul Kidambi et al. "Morel: Model-based offline reinforcement learning". In: *arXiv preprint arXiv:2005.05951* (2020).

3 Tianhe Yu et al. "Mopo: Model-based offline policy optimization". In: *arXiv preprint arXiv:2005.13239* (2020).

Main question

How many samples do we need to evaluate the policy?

- Under what assumptions, we need an exponential number of samples?
- Under what assumptions, for a given algorithm, we need a polynomial number of samples?

Linear Function Approximation

Assumption of Realizability[a]

[a] Ruosong Wang et al. "Instabilities of Offline RL with Pre-Trained Neural Representation". In: *arXiv preprint arXiv:2103.04947* (2021).

For the policy $\pi : \mathcal{S} \to \mathcal{A}$ to evaluated, there exists $\theta^\star \in \mathbb{R}^d$ and a feature extractor $\phi(s,a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ such that for all $(s,a) \in \mathcal{S} \times \mathcal{A}$, $Q^\pi(s,a) = (\theta^\star)^T \phi(s,a)$. Without loss of generality, we assume that we work in a coordinate system such that $||\theta^\star||_2 \leq \dfrac{\sqrt{d}}{1-\gamma}$ and $||\phi(s,a)||_2 \leq 1$

Feature covariance matrix

$$\Lambda \triangleq \mathbb{E}_{(s,a)\sim\mu}\left[\phi(s,a)\phi(s,a)^\top\right]$$

$$\bar{\Lambda} \triangleq \mathbb{E}_{(s,a)\sim\mu,\bar{s}\sim P(\cdot|s,a),\bar{a}\sim\pi(\cdot|\bar{s})}\left[\phi(\bar{s},\bar{a})\phi(\bar{s},\bar{a})^\top\right]$$

The lower bound: realizability and coverage[1]

Assumption 1 (Realizable Linear Function Approximation). *For every policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, there exists $\theta_1^\pi, \ldots \theta_H^\pi \in \mathbb{R}^d$ such that for all $(s, a) \in \mathcal{S} \times \mathcal{A}$ and $h \in [H]$, $Q_h^\pi(s, a) = (\theta_h^\pi)^\top \phi(s, a)$.*

Assumption 2 (Feature Coverage). *For all $(s, a) \in \mathcal{S} \times \mathcal{A}$, assume our feature map is bounded such that $\|\phi(s, a)\|_2 \leq 1$. Furthermore, suppose for each $h \in [H]$, the data distributions μ_h satisfy the following minimum eigenvalue condition: $\sigma_{\min}\left(\mathbb{E}_{(s,a)\sim\mu_h}[\phi(s, a)\phi(s, a)^\top]\right) = 1/d.$*[2]

Note that $\dfrac{1}{d}$ is the largest possible minimum eigenvalue.

Theorem 4.1. *Suppose Assumption 2 holds. Fix an algorithm that takes as input both a policy and a feature mapping. There exists a (deterministic) MDP satisfying Assumption 1, such that for any policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, the algorithm requires $\Omega((d/2)^H)$ samples to output the value of π up to constant additive approximation error with probability at least 0.9.*

[1] Ruosong Wang, Dean Foster, and Sham M. Kakade. "What are the Statistical Limits of Offline RL with Linear Function Approximation?" In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=30EvkP2aQLD.

The upper bound: Low distribution shift[1]

Assumption 3. *We assume that for each $h \in [H]$, there exists $C_h \geq 1$ such that $\overline{\Lambda_h} \preceq C_h \Lambda_h$.*

Note that C_h measures the distribution shift.
For **Least-Squares Policy Evaluation** algorithm, there is the following theorem.

Theorem 5.2. *Suppose for the given policy π, there exists $\theta_1, \theta_2, \ldots, \theta_d \in \mathbb{R}^d$ such that for each $h \in [H]$, $Q_h^\pi(s,a) = \phi(s,a)^\top \theta_h$ for all $(s,a) \in \mathcal{S}_h \times \mathcal{A}$ and $\|\theta_h\|_2 \leq H\sqrt{d}$.[4] Let $\lambda = CH\sqrt{d \log(dH/\delta)N}$ for some $C > 0$. With probability at least $1-\delta$, for some $c > 0$, $(Q_1^\pi(s_1, \pi(s_1)) - \hat{Q}_1(s_1, \pi(s_1)))^2 \leq c \cdot \prod_{h=1}^H C_h \cdot dH^5 \cdot \sqrt{d \log(dH/\delta)/N}$.*

[1] Ruosong Wang, Dean Foster, and Sham M. Kakade. "What are the Statistical Limits of Offline RL with Linear Function Approximation?" In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=30EvkP2aQLD.

The upper bound: Policy Completeness[1]

Assumption 2 (Policy Completeness). *For any* $\theta \in \mathbb{R}^d$, *there exists* $\theta' \in \mathbb{R}^d$, *such that for any* $(s,a) \in \mathcal{S} \times \mathcal{A}$,

$$\phi(s,a)^\top \theta' = \mathbb{E}_{r\sim R(s,a), s'\sim P(s,a)}[r + \gamma \phi(s', \pi(s'))^\top \theta].$$

For **Fitted Q-Iteration** algorithm, under the above assumption, there is the following theorem.

Lemma 4.2. *Suppose* $N \geq \mathrm{poly}(d, 1/\varepsilon, 1/(1-\gamma), 1/\sigma_{\min}(\Lambda))$, *by taking* $T \geq C \log(d/(\varepsilon(1-\gamma)))/(1-\gamma)$ *for some constant* $C > 0$, *we have*

$$|\hat{Q}_T(s,a) - Q^\pi(s,a)| \leq \varepsilon$$

for all $(s,a) \in \mathcal{S} \times \mathcal{A}$.

[1] Ruosong Wang et al. "Instabilities of Offline RL with Pre-Trained Neural Representation". In: *arXiv preprint arXiv:2103.04947* (2021).

Future work

- Theoretic grantees for more commonly used algorithms
- New algorithms, **new benchmark**
- Realistic guidance for application (e.g. how to sample)