EEA-EV002
Advanced topics in Reinforcement Learning
Session 3

Syeda Sakira Hassan

Department of Electrical Engineering and Automation
Aalto University

January 26, 2022

Contents

Next

Highlights

Kumar, Aviral and Zhou, Aurick and Tucker, George and Levine, Sergey, **Conservative q-learning for offline reinforcement learning**, *The 34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, 2020.[1]

- Offline reinforcement learning (RL) algorithms typically suffer from overestimation of the values

Highlights

Kumar, Aviral and Zhou, Aurick and Tucker, George and Levine, Sergey, **Conservative q-learning for offline reinforcement learning**, *The 34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, 2020.[1]

- Offline reinforcement learning (RL) algorithms typically suffer from overestimation of the values
- Conservative Q-Learning is introduced to learn a conservative Q-function where the value of a policy under this Q-function lower-bounds its true value

Highlights

Kumar, Aviral and Zhou, Aurick and Tucker, George and Levine, Sergey, **Conservative q-learning for offline reinforcement learning**, *The 34th Conference on Neural Information Processing Systems (NeurIPS 2020)*, 2020.[1]

- Offline reinforcement learning (RL) algorithms typically suffer from overestimation of the values
- Conservative Q-Learning is introduced to learn a conservative Q-function where the value of a policy under this Q-function lower-bounds its true value
- Works on both discrete and continuous state and action domains

Next

Introduction

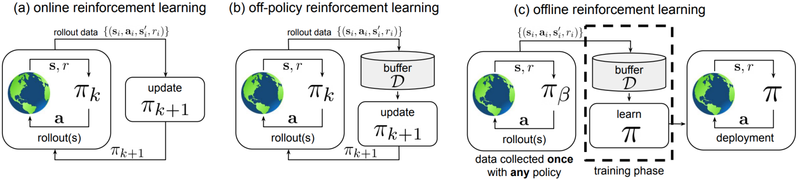


Figure 1: Pictorial illustration of classic online reinforcement learning (a), classic off-policy reinforcement learning (b), and offline reinforcement learning (c). In online reinforcement learning (a), the policy π_k is updated with streaming data collected by π_k itself. In the classic off-policy setting (b), the agent's experience is appended to a data buffer (also called a replay buffer) $\mathcal{D}$, and each new policy π_k collects additional data, such that $\mathcal{D}$ is composed of samples from $\pi_0, \pi_1, \ldots, \pi_k$, and all of this data is used to train an updated new policy π_{k+1}. In contrast, offline reinforcement learning employs a dataset $\mathcal{D}$ collected by some (potentially unknown) behavior policy π_β. The dataset is collected once, and is not altered during training, which makes it feasible to use large previous collected datasets. The training process does not interact with the MDP at all, and the policy is only deployed after being fully trained.

[0] *Offline reinforcement learning: Tutorial, review, and perspectives on open problems.*[2]

Benefits

- Several applications: robotics, healthcare, dialogue agents

Benefits

- Several applications: robotics, healthcare, dialogue agents
- Removes complexities with active data collection: safety, and cost

Benefits

- Several applications: robotics, healthcare, dialogue agents
- Removes complexities with active data collection: safety, and cost
- Pre-training + Fine tuning

Preliminaries: Basic RL

- Agent

Preliminaries: Basic RL

- Agent
- state: s

Preliminaries: Basic RL

- Agent
- state: s
- action: $a \sim \pi(a|s)$

Preliminaries: Basic RL

- Agent
- state: s
- action: $a \sim \pi(a|s)$
- reward: $r(a|s)$

Preliminaries: Basic RL

- Agent
- state: s
- action: $a \sim \pi(a|s)$
- reward: $r(a|s)$
- RL objective: $\max_{\pi} \sum_{t=1}^{T} \mathbb{E}_{s_t, a_t \sim \pi}[\gamma^t r(s_t, a_t)]$

Preliminaries: Basic RL

- Agent
- state: s
- action: $a \sim \pi(a|s)$
- reward: $r(a|s)$
- RL objective: $\max_{\pi} \sum_{t=1}^{T} \mathbb{E}_{s_t, a_t \sim \pi}[\gamma^t r(s_t, a_t)]$
- Q-function: $Q^{\pi}(s_t, a_t) = \sum_{t'=t}^{T} \mathbb{E}_{s'_t, a'_t \sim \pi}[\gamma^{t'-t} r(s'_t, a'_t)|s_t, a_t]$

Preliminaries: Basic RL

- Agent
- state: s
- action: $a \sim \pi(a|s)$
- reward: $r(a|s)$
- RL objective: $\max_\pi \sum_{t=1}^{T} \mathbb{E}_{s_t, a_t \sim \pi}[\gamma^t r(s_t, a_t)]$
- Q-function: $Q^\pi(s_t, a_t) = \sum_{t'=t}^{T} \mathbb{E}_{s'_t, a'_t \sim \pi}[\gamma^{t'-t} r(s'_t, a'_t)|s_t, a_t]$
- Learn Q-function $B^* Q^\pi(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim \pi}[\max_{a'} Q(s', a')]$

Preliminaries: Basic RL

- Agent
- state: s
- action: $a \sim \pi(a|s)$
- reward: $r(a|s)$
- RL objective: $\max_\pi \sum_{t=1}^{T} \mathbb{E}_{s_t, a_t \sim \pi}[\gamma^t r(s_t, a_t)]$
- Q-function: $Q^\pi(s_t, a_t) = \sum_{t'=t}^{T} \mathbb{E}_{s'_t, a'_t \sim \pi}[\gamma^{t'-t} r(s'_t, a'_t)|s_t, a_t]$
- Learn Q-function $B^* Q^\pi(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim \pi}[\max_{a'} Q(s', a')]$
- Enforce $\forall s$, minimize

$$\sum_i \left(Q^\pi(s, a) - \underbrace{r(s, a) + \gamma \mathbb{E}_{s' \sim \pi}[\max_{a'} Q(s', a')]}_{\text{y}} \right)^2$$

Preliminaries: General actor-critic algorithm

Preliminaries: General actor-critic algorithm

- 2 function approximators

Preliminaries: General actor-critic algorithm

- 2 function approximators
- $\pi_\theta(a_t|s_t)$: Input: s_t, Output: $\pi(a_t|s_t)$, given θ

Preliminaries: General actor-critic algorithm

- 2 function approximators
- $\pi_\theta(a_t|s_t)$: Input: s_t, Output: $\pi(a_t|s_t)$, given θ
- $Q_\phi(s, a)$: Input: s_t, a_t, Output: $Q(s, a)$, given ϕ

Preliminaries: General actor-critic algorithm

- 2 function approximators
- $\pi_\theta(a_t|s_t)$: Input: s_t, Output: $\pi(a_t|s_t)$, given θ
- $Q_\phi(s,a)$: Input: s_t, a_t, Output: $Q(s,a)$, given ϕ

Policy evaluation and policy improvement

$$\hat{Q}^{k+1} \leftarrow \arg\min_{Q} \mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[\left((r(\mathbf{s},\mathbf{a}) + \gamma\mathbb{E}_{\mathbf{a}'\sim\hat{\pi}^k(\mathbf{a}'|\mathbf{s}')}[\hat{Q}^k(\mathbf{s}',\mathbf{a}')]) - Q(\mathbf{s},\mathbf{a})\right)^2\right] \text{ (policy evaluation)}$$

$$\hat{\pi}^{k+1} \leftarrow \arg\max_{\pi} \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\pi^k(\mathbf{a}|\mathbf{s})}\left[\hat{Q}^{k+1}(\mathbf{s},\mathbf{a})\right] \quad \text{(policy improvement)}$$

Offline RL

- $\mathbb{D} = \{(s, a, r, s')\}$ collected from π_β

Offline RL

- $\mathbb{D} = \{(s, a, r, s')\}$ collected from π_β
- $s \sim d^{\pi_\beta}(s)$

Offline RL

- $\mathbb{D} = \{(s, a, r, s')\}$ collected from π_β
- $s \sim d^{\pi_\beta}(s)$
- $a \sim \pi_\beta(a|s)$

Offline RL

- $\mathbb{D} = \{(s, a, r, s')\}$ collected from π_β
- $s \sim d^{\pi_\beta}(s)$
- $a \sim \pi_\beta(a|s)$
- $s' \sim p(s'|s, a)$

Offline RL

- $\mathbb{D} = \{(s, a, r, s')\}$ collected from π_β
- $s \sim d^{\pi_\beta}(s)$
- $a \sim \pi_\beta(a|s)$
- $s' \sim p(s'|s, a)$
- r

Offline RL

- $\mathbb{D} = \{(s, a, r, s')\}$ collected from π_β
- $s \sim d^{\pi_\beta}(s)$
- $a \sim \pi_\beta(a|s)$
- $s' \sim p(s'|s, a)$
- r
- RL objective

Why Offline RL works?

- Good stuff in D
- Generalization
- Stitching

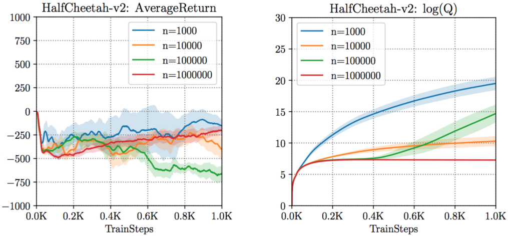
Why Offline RL does not work?

Overfitting

- This is not a statistical overfitting issue -- performance is **bad** (and doesn't improve) even with **infinite** data.

- This is really about **data distribution shift** and tackling **out-of-distribution values**.

Half-cheetah with expert data



Distribution shift

- Most RL algorithms estimate a “value”/ “goodness” of the policy and use this metric to improve

Distribution shift

- Most RL algorithms estimate a “value”/ “goodness” of the policy and use this metric to improve
- False optimism can easily arise: can overestimate policy values

Distribution shift

- Most RL algorithms estimate a “value”/ “goodness” of the policy and use this metric to improve
- False optimism can easily arise: can overestimate policy values
- Turns out that the resulting policy is significantly worse, due to the curse of horizon

Distribution shift

- Most RL algorithms estimate a "value"/ "goodness" of the policy and use this metric to improve
- False optimism can easily arise: can overestimate policy values
- Turns out that the resulting policy is significantly worse, due to the curse of horizon
- Typically online RL methods based on active data collection can correct for this issue with carefully designed exploration strategies

Solutions to mitigate distributional shift

- Policy constraints: constrain the learned policy

Solutions to mitigate distributional shift

- Policy constraints: constrain the learned policy
- Uncertainty estimation: estimate the epistemic uncertainty of Q-values, and then utilize this uncertainty to detect distributional shift

Policy constraints

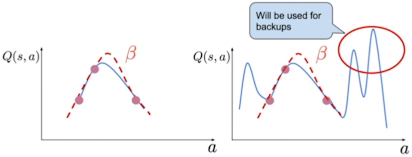
$$Q(s, a) \leftarrow r(s, a) + \gamma \mathbb{E}_{a' \sim \pi_\theta(a'|s')}[Q(s', a')] \tag{1}$$

Policy constraints

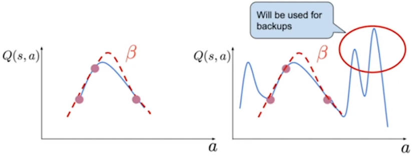
$$Q(s,a) \leftarrow r(s,a) + \gamma \mathbb{E}_{a' \sim \pi_\theta(a'|s')}[Q(s',a')] \tag{1}$$

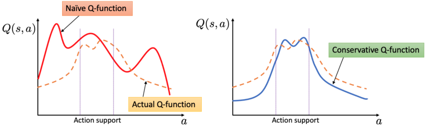
$$\pi_\theta = \arg\max_{\pi_\theta} \mathbb{E}_{s \sim D, a \sim \pi_\theta(a|s)}[Q(s,a)] \text{ s.t. } D(\pi_\theta, \pi_\beta) \leq \epsilon \tag{2}$$

Policy constraints



Policy constraints





Next

Highlights

Offline RL

Conservative Q Learning

Conclusion

References

Penalize the Q-functions directly

Option 1

$$Q(s, a) \leftarrow r(s, a) + \gamma \mathbb{E}_{a' \sim \pi_\theta(a'|s')}[Q(s', a')] - \alpha D(\pi_\theta, \pi_\beta) \tag{3}$$

Penalize the Q-functions directly

Option 1

$$Q(s,a) \leftarrow r(s,a) + \gamma \mathbb{E}_{a' \sim \pi_\theta(a'|s')}[Q(s',a')] - \alpha D(\pi_\theta, \pi_\beta) \quad (3)$$

Option 2

Better way to do it automatically?

CQL: Learn lower bound Q-function

CQL-v1

$$Q_{\text{CQL}}(\mathbf{s}, \mathbf{a}) := \arg \min_{Q} \max_{\mu} \; \mathbb{E}_{\mathbf{s}\sim\mathcal{D}} \mathbb{E}_{\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s}, \mathbf{a})] + \frac{1}{2\alpha} \mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}} \left[(Q(\mathbf{s}, \mathbf{a}) - y(\mathbf{s}, \mathbf{a}))^2 \right]$$

$$\forall \mathbf{s} \in \mathcal{D}, \mathbf{a}, \; Q_{\text{CQL}}(\mathbf{s}, \mathbf{a}) \leq Q(\mathbf{s}, \mathbf{a})$$

CQL: Learn lower bound Q-function

CQL-v1

$$Q_{\text{CQL}}(\mathbf{s},\mathbf{a}) := \arg\ \min_{Q}\max_{\mu}\ \mathbb{E}_{\mathbf{s}\sim\mathcal{D}}\mathbb{E}_{\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})] + \frac{1}{2\alpha}\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[(Q(\mathbf{s},\mathbf{a}) - y(\mathbf{s},\mathbf{a}))^2\right]$$

$$\forall \mathbf{s}\in\mathcal{D},\mathbf{a},\ \ Q_{\text{CQL}}(\mathbf{s},\mathbf{a}) \leq Q(\mathbf{s},\mathbf{a})$$

Theorem 3.1. *For any* $\mu(\mathbf{a}|\mathbf{s})$ *with* $\operatorname{supp}\mu \subset \operatorname{supp}\hat{\pi}_\beta$, *with probability* $\geq 1-\delta$, $\hat{Q}^\pi$ *(the Q-function obtained by iterating Equation 1) satisifies:*

$$\forall \mathbf{s}\in\mathcal{D},\mathbf{a},\ \ \hat{Q}^\pi(s,a) \leq Q^\pi(\mathbf{s},\mathbf{a}) - \alpha\left[(I-\gamma P^\pi)^{-1}\frac{\mu}{\hat{\pi}_\beta}\right](\mathbf{s},\mathbf{a}) + \left[(I-\gamma P^\pi)^{-1}\frac{C_{r,T,\delta}R_{\max}}{(1-\gamma)\sqrt{|\mathcal{D}|}}\right](\mathbf{s},\mathbf{a}).$$

Thus, if α *is sufficiently large, then* $\hat{Q}^\pi(\mathbf{s},\mathbf{a}) \leq Q^\pi(\mathbf{s},\mathbf{a}), \forall \mathbf{s}\in\mathcal{D},\mathbf{a}$. *When* $\hat{\mathcal{B}}^\pi = \mathcal{B}^\pi$, *any* $\alpha > 0$ *guarantees* $\hat{Q}^\pi(\mathbf{s},\mathbf{a}) \leq Q^\pi(\mathbf{s},\mathbf{a}), \forall \mathbf{s}\in\mathcal{D},\mathbf{a}\in\mathcal{A}$.

CQL: Learn tighter lower bound Q-function

CQL-v2

$$Q_{\text{CQL}}(\mathbf{s}, \mathbf{a}) := \arg\min_{Q}\max_{\mu} \Big(\underbrace{\mathbb{E}_{\mathbf{s}\sim\mathcal{D}}\mathbb{E}_{\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})]}_{\text{Minimize the big Q-values}} - \underbrace{\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}[Q(\mathbf{s},\mathbf{a})]}_{\text{Maximize the data Q-values}}\Big) + \underbrace{\frac{1}{2\alpha}\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[(Q(\mathbf{s},\mathbf{a}) - y(\mathbf{s},\mathbf{a}))^2\right]}_{\text{Standard TD error}}$$

$$\forall \mathbf{s} \in \mathcal{D}, \mathbf{a},\ Q_{\text{CQL}}(\mathbf{s},\mathbf{a}) \leq Q(\mathbf{s},\mathbf{a})$$ ✗

$$\forall\ \mathbf{s} \in \mathcal{D},\ V_{\text{CQL}}(\mathbf{s}) \leq V(\mathbf{s})$$ ✓

CQL: Learn tighter lower bound Q-function

CQL-v2

$$Q_{\text{CQL}}(\mathbf{s},\mathbf{a}) := \arg\min_{Q}\max_{\mu}\ \Big(\overbrace{\mathbb{E}_{\mathbf{s}\sim\mathcal{D}}\mathbb{E}_{\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}[Q(\mathbf{s},\mathbf{a})]}^{\text{Minimize the big Q-values}} - \overbrace{\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}[Q(\mathbf{s},\mathbf{a})]}^{\text{Maximize the data Q-values}}\Big) + \overbrace{\frac{1}{2\alpha}\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[(Q(\mathbf{s},\mathbf{a}) - y(\mathbf{s},\mathbf{a}))^2\right]}^{\text{Standard TD error}}$$

$$\forall \mathbf{s}\in\mathcal{D},\mathbf{a},\ \ Q_{\text{CQL}}(\mathbf{s},\mathbf{a}) \le Q(\mathbf{s},\mathbf{a})$$ ✗

$$\forall\ \mathbf{s}\in\mathcal{D},\ \ V_{\text{CQL}}(\mathbf{s}) \le V(\mathbf{s})$$ ✓

Theorem 3.2 (Equation 2 results in a tighter lower bound). *The value of the policy under the Q-function from Equation 2, $\hat{V}^{\pi}(\mathbf{s}) = \mathbb{E}_{\pi(\mathbf{a}|\mathbf{s})}[\hat{Q}^{\pi}(\mathbf{s},\mathbf{a})]$, lower-bounds the true value of the policy obtained via exact policy evaluation, $V^{\pi}(\mathbf{s}) = \mathbb{E}_{\pi(\mathbf{a}|\mathbf{s})}[Q^{\pi}(\mathbf{s},\mathbf{a})]$, when $\mu = \pi$, according to:*

$$\forall \mathbf{s}\in\mathcal{D},\ \hat{V}^{\pi}(\mathbf{s}) \le V^{\pi}(\mathbf{s}) - \alpha\left[(I-\gamma P^{\pi})^{-1}\mathbb{E}_{\pi}\left[\frac{\pi}{\hat{\pi}_{\beta}} - 1\right]\right](\mathbf{s}) + \left[(I-\gamma P^{\pi})^{-1}\frac{C_{r,T,\delta}R_{\max}}{(1-\gamma)\sqrt{|\mathcal{D}|}}\right](\mathbf{s}).$$

Thus, if $\alpha > \frac{C_{r,T}R_{\max}}{1-\gamma}\cdot\max_{\mathbf{s}\in\mathcal{D}}\frac{1}{|\sqrt{|\mathcal{D}(\mathbf{s})|}}\cdot\left[\sum_{\mathbf{a}}\pi(\mathbf{a}|\mathbf{s})\left(\frac{\pi(\mathbf{a}|\mathbf{s})}{\hat{\pi}_{\beta}(\mathbf{a}|\mathbf{s})}\right) - 1)\right]^{-1}$, $\forall \mathbf{s}\in\mathcal{D},\ \hat{V}^{\pi}(\mathbf{s}) \le V^{\pi}(\mathbf{s})$, *with probability* $\ge 1-\delta$. *When* $\hat{\mathcal{B}}^{\pi} = \mathcal{B}^{\pi}$, *then any* $\alpha > 0$ *guarantees* $\hat{V}^{\pi}(\mathbf{s}) \le V^{\pi}(\mathbf{s}), \forall \mathbf{s}\in\mathcal{D}$.

CQL

$$\min_{Q} \max_{\mu} \; \alpha \left(\mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right] - \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right]\right)$$
$$+\frac{1}{2}\,\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[\left(Q(\mathbf{s},\mathbf{a}) - \hat{\mathcal{B}}^{\pi_k}\hat{Q}^k(\mathbf{s},\mathbf{a})\right)^2\right] + \mathcal{R}(\mu) \quad (\mathrm{CQL}(\mathcal{R}))\,. \quad (3)$$

CQL: Algorithm

Algorithm 1 Conservative Q-Learning (both variants)

Initialize Q-function, Q_θ, and optionally a policy, π_ϕ.
for step t in $\{1, \dots, \mathrm{N}\}$ **do**
    Train the Q-function using G_Q gradient steps on objective from Equation 4
    $\theta_t := \theta_{t-1} - \eta_Q \nabla_\theta \mathrm{CQL}(\mathcal{R})(\theta)$
    (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

Next

Conclusion

- CQL can prevent overestimation via learning lower-bound Q-values

Conclusion

- CQL can prevent overestimation via learning lower-bound Q-values
- CQL seems promising to directly apply to real-world tasks.

Conclusion

- CQL can prevent overestimation via learning lower-bound Q-values
- CQL seems promising to directly apply to real-world tasks.
- How should we detect overfitting?

Conclusion

- CQL can prevent overestimation via learning lower-bound Q-values
- CQL seems promising to directly apply to real-world tasks.
- How should we detect overfitting?
- How should we perform cross-validation?

Conclusion

- CQL can prevent overestimation via learning lower-bound Q-values
- CQL seems promising to directly apply to real-world tasks.
- How should we detect overfitting?
- How should we perform cross-validation?
- How should we integrate offline RL methods with online data collection?

Next

References

KUMAR, A., ZHOU, A., TUCKER, G., AND LEVINE, S.
Conservative q-learning for offline reinforcement learning.
In *The 34th Conference on Neural Information Processing Systems (NeurIPS 2020)* (2020).

LEVINE, S., KUMAR, A., TUCKER, G., AND FU, J.
Offline reinforcement learning: Tutorial, review, and perspectives on open problems.
arXiv preprint arXiv:2005.01643 (2020).

- `https://www.youtube.com/watch?v=qgZPZREor5I`
- `https://www.youtube.com/watch?v=536a-CvHUZ8`
- `https://vitalab.github.io/article/2021/06/09/CQL.html`

Thank you for listening!

Questions?