Offline Reinforcement Learning: Part 1

CS 224R
Course reminders

- Homework 2 due Wednesday
- Project proposal feedback coming out soon.
The plan for today

Offline RL: Part 1
1. Why offline RL? Can we just run off-policy methods?
2. Data constraint methods
3. Conservative methods
4. Data stitching

Part of homework 3!

Key learning goals:
- the key challenges arising in offline reinforcement learning
- two approaches for offline RL (& why they work!)
- how offline RL can improve over imitation learning
Why offline RL?

**Online RL process** (on-policy or off-policy)
- Collect data
- Update policy on latest data or data so far

**Offline RL process**
- Given static dataset
- Train policy on provided dataset

Why, or when, might offline RL be more useful?
- Leverage datasets collected by people, existing systems
- Online policy collection may be risky, unsafe
- Reuse previously collected data rather than recollecting (e.g. previous experiments, projects, robots, institutions)

**Note**: A blend of offline then online RL is also possible!
Why offline RL?

**Offline RL process**
- Given static dataset
- Train policy on provided dataset

More formally:

Offline dataset $\mathcal{D} : \{(s, a, s', r)\}$ sampled from some unknown policy $\pi_\beta$

"behavior policy"

(Note: $\pi_\beta$ may be a mixture of policies)

$s \sim d^\pi_\beta(\cdot)$

$a \sim \pi_\beta(\cdot \mid s)$

$s' \sim p(\cdot \mid s, a)$

$r = r(s, a)$

Objective: $\max_\theta \sum_t E_{s_t \sim d^\pi_\theta(\cdot), a_t \sim \pi_\theta(\cdot \mid s_t)} \left[ r(s_t, a_t) \right]$
Why offline RL?

**Offline RL** process
- Given static dataset
- Train policy on provided dataset

Where does the data come from?
- Human collected data
- Data from a hand-designed system / controller
- Data from previous RL run(s)
- A mixture of sources
Can we just use off-policy algorithms?

**Recall:** Q-learning objective \[ \sum_{(s,a,s') \sim D} \left\| Q(s, a) - \left(r(s, a) + \gamma \max_{a'} Q(s', a') \right) \right\|^2 \]

What happens if you optimize this using a static dataset?

(e.g. say data collected by a mediocre policy)
Can we just use off-policy algorithms?

Recall: Q-learning objective

\[ \sum_{(s,a,s') \sim D} \left\| Q(s, a) - \left( r(s, a) + \gamma \max_{a'} Q(s', a') \right) \right\|^2 \]

What happens if you optimize this using a static dataset? (e.g. say data collected by a mediocre policy)

What happens when evaluating \( Q \) on actions \( a' \) not in the dataset?

Randomly init. \( Q \)-function for state \( s' \)

\( Q(s', a') \)

- \( Q \)-function will be unreliable on OOD actions
- \( \max \) will seek out actions where \( Q \)-function is over-optimistic
- After values propagate, \( Q \)-values will become substantially overestimated.

Another perspective: learned policy deviates too much from behavior policy.
How to mitigate overestimation in offline RL?

This is the core goal of offline RL methods!
The plan for today

Offline RL: Part 1

1. Why offline RL? Can we just run off-policy methods?
2. Data constraint methods
3. Conservative methods
4. Data stitching

Part of homework 3!
How to mitigate overestimation in offline RL?

Recall: Q-learning objective

$$\sum_{(s, a, s') \sim D} \left\| Q(s, a) - \left( r(s, a) + \gamma \max_{a'} Q(s', a') \right) \right\|^2$$

Can we constrain $a'$ to stay close to behavior policy?

If so: we could avoid querying $Q$ on OOD actions!

New objective:

$$\sum_{(s, a, s') \sim D} \left\| Q(s, a) - \left( r(s, a) + \gamma E_{a' \sim \pi_{\text{new}}(\cdot | s')} Q(s', a') \right) \right\|^2$$

$$\pi_{\text{new}} = \arg \max_{\pi} E_{a' \sim \pi(\cdot | s')} Q(s', a') \text{ s.t. } \pi \text{ close to } \pi_{\beta}$$
How to mitigate overestimation in offline RL?

Can we constrain $a'$ to stay close to behavior policy?

If so: we could avoid querying $Q$ on OOD actions!

**New objective:**

$$\sum_{(s,a,s') \sim \mathcal{D}} \left\| Q(s,a) - (r(s,a) + \gamma E_{a' \sim \pi_{new}(\cdot | s')} Q(s',a')) \right\|^2$$

$$\pi_{new} = \arg \max_{\pi} E_{a' \sim \pi(\cdot | s')} Q(s',a') \quad \text{s.t.} \quad \pi \text{ close to } \pi_\beta$$

**Issue:** We don’t know what $\pi_\beta$ is!

Many “data constraint” methods will **fit a policy** to the data.

(i.e. learn a proxy for $\pi_\beta$ through imitation)
How to mitigate overestimation in offline RL?

Can we constrain $a'$ to stay close to behavior policy?

If so: we could avoid querying $Q$ on OOD actions!

New objective:

$$\sum_{(s,a,s') \sim D} \left\| Q(s, a) - (r(s, a) + \gamma E_{a' \sim \pi_{\text{new}}(\cdot | s')} Q(s', a')) \right\|^2$$

$$\pi_{\text{new}} = \arg \max_{\pi} E_{a' \sim \pi(\cdot | s')} Q(s', a') \text{ s.t. } \pi \text{ close to } \pi_{\beta}$$

Forms of policy constraints?

1. **support constraint**: $\pi(a | s) > 0$ only if $\pi_{\beta}(a | s) \geq \epsilon$
   
   + close to what we want
   - challenging to implement in practice

2. **KL divergence**: $D_{KL}(\pi \| \pi_{\beta})$

   + easy to implement
   - not necessarily what we want
How to implement data constraint methods?

1. Change actor update:

\[
\theta \leftarrow \arg \max_{\theta} E_{s \sim D, a \sim \pi_\theta(\cdot|s)} [Q(s, a)] - \lambda D_{KL}(\pi_\theta \| \pi_\beta)
\]

\[
\theta \leftarrow \arg \max_{\theta} E_{s \sim D, a \sim \pi_\theta(\cdot|s)} [Q(s, a) + \lambda \log \pi_\beta(a|s) + \lambda H(\pi_\theta(\cdot|s))] \quad \text{Lagrange multiplier}
\]

2. Modify the reward function:

\[
\tilde{r}(s, a) = r(s, a) - D_{KL}(\pi_\theta \| \pi_\beta)
\]

Policy will also account for future divergence


Slide adapted from Sergey Levine
The plan for today

Offline RL: Part 1
1. Why offline RL? Can we just run off-policy methods?
2. Data constraint methods
3. Conservative methods
4. Data stitching

Part of homework 3!
How to mitigate overestimation in offline RL?

Recall: Randomly init. $Q$-function for state $s'$

Can we discourage overestimation? without explicitly modeling the behavior policy

What if we just push down on large Q-values?

$$\hat{Q}^\pi = \arg \min_Q \max_\mu E_{(s,a,s') \sim D} \left[ \left( Q(s, a) - (r(s, a) + \gamma E_\pi [Q(s', a')]) \right)^2 \right] + \alpha E_{s \sim D, a \sim \mu (\cdot | s)} [Q(s, a)]$$

Can show that $\hat{Q}^\pi \leq Q^\pi$ for large enough $\alpha$

Slide adapted from Sergey Levine
How to mitigate overestimation in offline RL?

Can we discourage overestimation?
without explicitly modeling the behavior policy

\[
\hat{Q}^\pi = \arg \min_{Q} \max_{\mu} E_{(s,a,s') \sim D} \left[ \left( Q(s, a) - (r(s, a) + \gamma E_{\pi}[Q(s', a')]) \right)^2 \right] + \alpha E_{s \sim D, a \sim \mu(\cdot|s)}[Q(s, a)] \\
- \alpha E_{(s,a) \sim D}[Q(s, a)]
\]

No longer guaranteed that \( \hat{Q}^\pi \leq Q^\pi \) for all \((s, a)\).

BUT, guaranteed that \( E_{\pi(a|s)}[\hat{Q}^\pi(s, a)] \leq E_{\pi(a|s)}[Q^\pi(s, a)] \) for all \( s \in D \).

Conservative Q-learning (CQL)

Slide adapted from Sergey Levine
How to mitigate overestimation in offline RL?

**Conservative Q-learning (CQL)**

Full algorithm

1. Update $\hat{Q}^\pi$ using $L_{CQL}$ using $D$
2. Update policy $\pi$

If actions are discrete: $\pi(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_{\bar{a}} \hat{Q}(s, \bar{a}) \\ 0 & \text{otherwise} \end{cases}$

If actions are continuous: $\theta \leftarrow \theta + \eta \nabla_{\theta} E_{s \sim D,a \sim \pi_\theta(\cdot|s)} \left[ \hat{Q}(s, a) \right]$
How to mitigate overestimation in offline RL?

**Conservative Q-learning (CQL)**

1. Update $\hat{Q}^\pi$ using $L_{CQL}$ using $D$
2. Update policy $\pi$

How compute objective $L_{CQL}$?

$$\hat{Q}^\pi = \arg \min_Q \max_\mu E_{(s,a,s') \sim D} \left[ \left( Q(s, a) - (r(s, a) + \gamma E_\pi [Q(s', a')]) \right)^2 + \alpha E_{s \sim D, a \sim \mu(\cdot|s)} [Q(s, a)] - \alpha E_{(s,a) \sim D} [Q(s, a)] + R(\mu) \right]$$

Common choice: $R(\mu) = E_{s \sim D} [H(\mu(\cdot|s))]$

With max entropy regularizer $R$, optimal $\mu(a|s) \propto \exp(Q(s, a))$

Then: $E_{s \sim D, a \sim \mu(\cdot|s)} [Q(s, a)] = \log \sum_a \exp(Q(s, a))$

Don’t need to construct $\mu$ directly.

You will implement CQL in homework 3!

Slide adapted from Sergey Levine
Aside: Model-based offline RL

Key idea: Instead of minimizing Q-values of policy actions, minimize Q-values of model-generated (s, a)

CQL objective:

$$\hat{Q} = \arg \min_{Q} \max_{\mu} E_{(s,a,s') \sim D} \left[ \left( Q(s, a) - (r(s, a) + \gamma E_{\pi}[Q(s', a')]) \right)^2 \right] + \alpha E_{s \sim D, a \sim \mu(s)}[Q(s, a)] - \alpha E_{(s, a) \sim D}[Q(s, a)]$$

Intuition: If model produces data that look clearly different from the real data, it’s easy for the Q-function to make it look bad.
The plan for today

Offline RL: Part 1

1. Why offline RL? Can we just run off-policy methods?
2. Data constraint methods
3. Conservative methods
4. Data stitching

Part of homework 3!
Why offline RL versus imitation learning?

Offline data may not be optimal!

→ Offline RL can leverage reward information to outperform behavior policy.

→ Good offline RL methods can **stitch** together good behaviors.

(Recall: Imitation methods can’t outperform the expert.)

s₁ -> s₃ is good behavior

s₇ -> s₉ is good behavior

Offline RL methods can learn a policy that goes from s₁ to s₉!
The plan for today

Offline RL: Part 1
1. Why offline RL? Can we just run off-policy methods?
2. Data constraint methods
3. Conservative methods
4. Data stitching

Key learning goals:
- the key challenges arising in offline reinforcement learning
- two approaches for offline RL (& why they work!)
- how offline RL can improve over imitation learning
Summary

Why offline RL? Online data is expensive. *Reusing offline data* is good!

**Key challenge:** Overestimating Q-values because of shift between $\pi_\beta$ and $\pi_\theta$

- can explicitly constrain to the data by modeling $\pi_\beta$
  + fairly intuitive    - often too conservative in practice
- implicitly constrain to data by penalizing Q-values
  + simple    + can work well in practice    - need to tune alpha

Trajectory stitching allows offline RL methods to improve over imitation.

**Next time:** other offline RL approaches & hyperparameter tuning
Course reminders

- Homework 2 due Wednesday
- Project proposal feedback coming out soon.