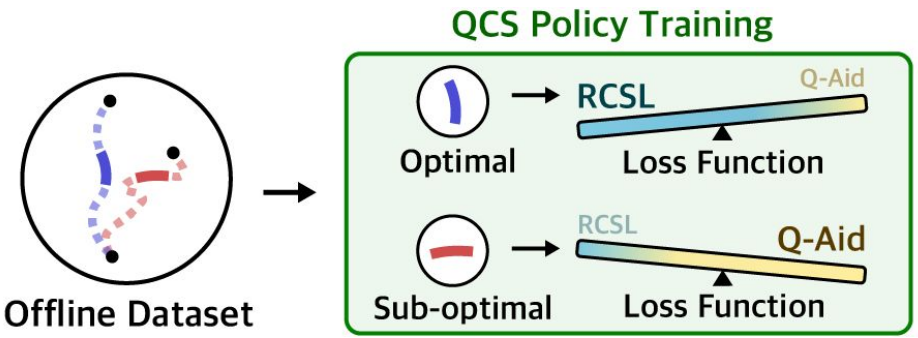
Adaptive Q-Aid for Conditional Supervised Learning in Offline Reinforcement Learning

NeurIPS 2024

Jeonghye Kim, Suyoung Lee, Woojun Kim, Youngchul Sung*

Conceptual idea of QCS



For training a policy with an offline dataset,

Follow *RCSL when learning from optimal trajectories where it predicts actions confidently but the Q-function may stitch incorrectly.

Refer to the Q-function when learning from sub-optimal trajectories where RCSL is less certain but the Q-function is likely accurate.

***RCSL? (e.g. Decision Transformer[1], etc.)**

- RCSL is a method in which policies are learned by conditioning on target outcomes, framing RL as a sequence modeling problem.
- RCSL has shown effective planning capabilities, but it lacks stitching ability.

[1] Chen, Lili, et al. "Decision transformer: Reinforcement learning via sequence modeling." *NeurIPS 2021*.

When Is Q-Aid Beneficial for RCSL?

	halfcheetah-e	halfcheetah-m-r	hopper-e	hopper-m-r	walker2d-e	walker2d-m-r
DT	91.4 ± 1.7	36.6 ± 0.8	110.1 ± 0.9	82.7 ± 7.0	109.2 ± 1.5	66.6 ± 3.0
max-Q	-4.1 ± 1.1	52.8 ± 0.4	1.8 ± 1.0	92.1 ± 2.6	-0.2 ± 0.6	91.2 ± 1.9

$\text{argmax}_{a \in A} Q(s, a)$

For optimal datasets

- RCSL (DT) tends to perform well by mimicking actions.
- The max-Q policy performs notably poorly.

For suboptimal datasets

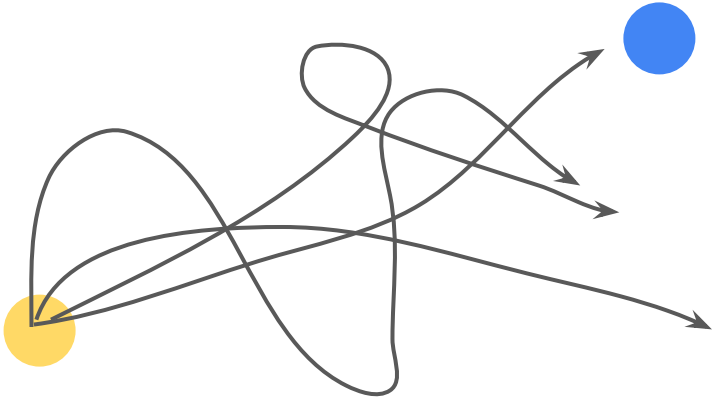
- RCSL (DT) shows suboptimal performance due to a **lack of stitching ability.**
- The max-Q policy outperforms RCSL by stitching together suboptimal trajectories using dynamic programming.

→ Q-aid can compensate for RCSL's lack of stitching ability in suboptimal datasets.
→ However, in optimal datasets, Q-aid might hinder RCSL. (why? - see next page)

Why Does Max-Q Policy Struggle with Optimal Datasets?

Optimal dataset

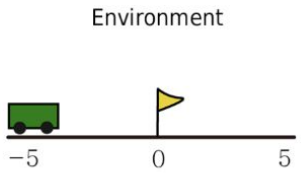
Sub-optimal dataset



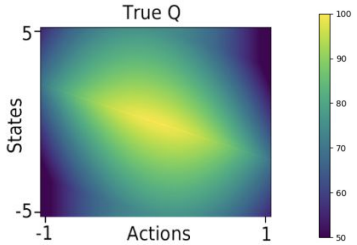
The optimal dataset **consists of similar actions that produce similarly high Q-values** for each state.

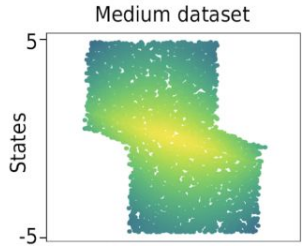
The sub-optimal dataset **consists of various actions that produce different Q-values** for each state.

Why Does Max-Q Policy Struggle with Optimal Datasets?

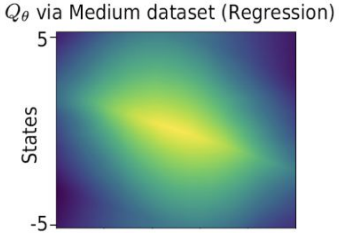


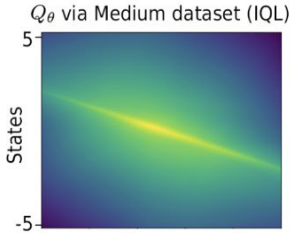
State (1D) : The car's position
Action (1D) : Moves in direction with 2x magnitude
Reward : +100 at position 0; penalty $-30 \cdot a^2$

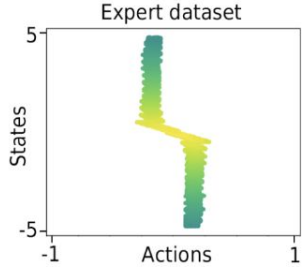




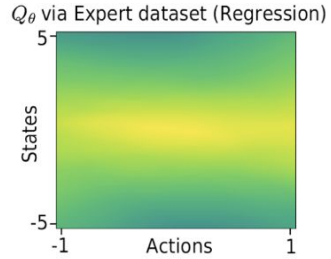
Has various actions with different Q values.

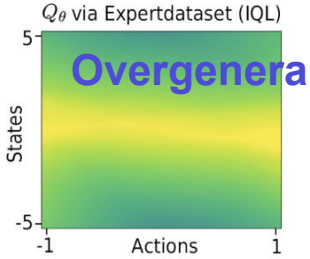






Has a limited range of actions with almost identical Q values.





Overgeneralization

Overgeneralization → The Q-function becomes noise-sensitive and inaccurate.

Q-Aided Conditional Supervised Learning

The complementary relationship

- RCSL excels at mimicking optimal, narrow datasets.
- The Q-function becomes a more effective critic when trained on diverse datasets with varied actions and Q-values.

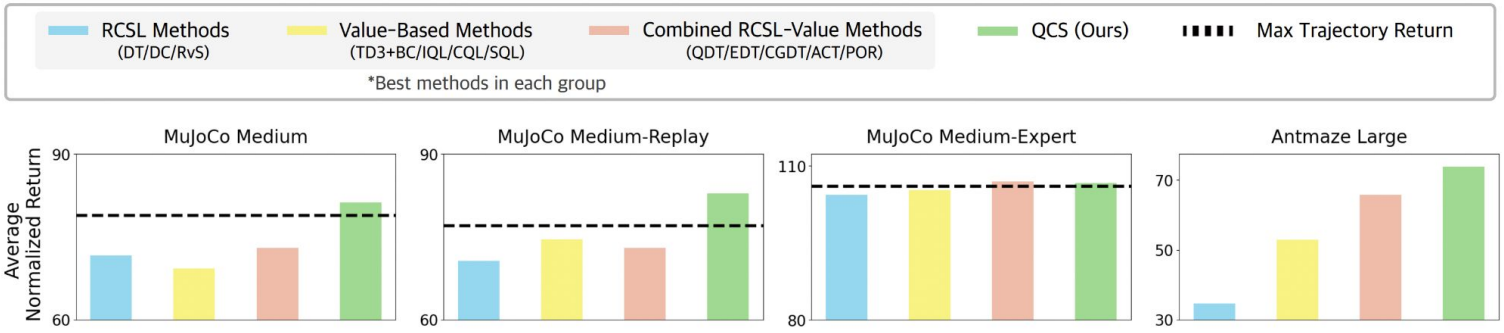
→ **Related to trajectory optimality (trajectory return).**

We can **apply varying degrees of Q-aid based on the trajectory return** for each sub-trajectory in RCSL.

$$\mathcal{L}_\pi^{\text{QCS}}(\phi) = \mathbb{E}_{\tau\sim\mathcal{D}}\left[\frac{1}{K}\sum_{j=0}^{K-1} \underbrace{\left\|a_{t+j} - \pi_\phi\left(\tau_{t:t+j}\right)\right\|_2^2}_{\text{RCSL}} - \underbrace{w\left(R(\tau)\right)}_{\text{QCS weight}} \cdot \underbrace{Q_\theta^{\text{IQL}}\left(s_{t+j}, \pi_\phi(\tau_{t:t+j})\right)}_{Q\text{ Aid}}\right]$$

trajectory return

Experiments



MuJoCo

	Value-Based Method				RCSL			Combined Method					Ours
Dataset	TD3+BC	IQL	CQL	SQL	DT	DC	RvS-R	QDT	EDT	CGDT	ACT	POR	QCS-R
halfcheetah-m	48.3	47.4	44.0	48.3	42.6	43.0	41.6	42.3	42.5	43.0	49.1	48.8	**59.0** ± 0.40
hopper-m	59.3	66.3	58.5	75.5	67.6	92.5	60.2	66.5	63.5	**96.9**	67.8	78.6	**96.4** ± 3.72
walker2d-m	83.7	78.3	72.5	84.2	74.0	79.2	71.7	67.1	72.8	79.1	80.9	81.1	**88.2** ± 1.08
halfcheetah-m-r	44.6	44.2	45.5	44.8	36.6	41.3	38.0	35.6	37.8	40.4	43.0	43.5	**54.1** ± 0.76
hopper-m-r	60.9	94.7	95.0	99.7	82.7	94.2	73.5	52.1	89.0	93.4	98.4	98.9	**100.4** ± 1.05
walker2d-m-r	81.8	73.9	77.2	81.2	66.6	76.6	60.6	58.2	74.8	78.1	56.1	76.6	**94.1** ± 2.01
halfcheetah-m-e	90.7	86.7	91.6	94.0	86.8	93.0	92.2	-	-	93.6	**96.1**	94.7	93.3 ± 1.78
hopper-m-e	98.0	91.5	105.4	**111.8**	107.6	110.4	101.7	-	-	107.6	111.5	90.0	110.2 ± 2.41
walker2d-m-e	110.1	109.6	108.8	110.0	108.1	109.6	106.0	-	-	109.3	113.3	109.1	**116.6** ± 2.44
average	75.3	77.0	77.6	83.1	74.7	82.2	71.7	-	-	82.4	79.6	80.1	**90.3**

AntMaze

	Value-Based Method				RCSL				Combined	Ours	
Dataset	TD3+BC	IQL	CQL	SQL	DT	DC	RvS-R	RvS-G	POR	QCS-R	QCS-G
antmaze-u	78.6	87.5	74.0	**92.2**	65.6	85.0	64.4	65.4	90.6	**92.7** ± 3.9	**92.5**± 4.6
antmaze-u-d	71.4	62.2	**84.0**	74.0	51.2	78.5	70.1	60.9	71.3	72.3 ± 12.4	82.5 ± 8.2
antmaze-m-p	10.6	71.2	61.2	80.2	4.3	33.2	4.5	58.1	**84.6**	81.6 ± 6.9	**84.8** ± 11.5
antmaze-m-d	3.0	70.0	53.7	**79.1**	1.2	27.5	7.7	67.3	**79.2**	**79.5** ± 5.8	75.2 ± 11.9
antmaze-l-p	0.2	39.6	15.8	53.2	0.0	4.8	3.5	32.4	58.0	68.7 ± 7.8	**70.0** ± 9.6
antmaze-l-d	0.0	47.5	14.9	52.3	0.5	12.3	3.7	36.9	73.4	70.6 ± 5.6	**77.3** ± 11.2
average	27.3	63.0	50.6	71.8	20.5	40.2	25.6	53.5	76.2	77.6	**80.4**

- QCS significantly outperforms prior value-based methods, RCSL, and combined methods.
- QCS greatly boosts efficiency in AntMaze, especially in large environments where RCSL struggled.

→ **QCS successfully combines the strengths of both RCSL and the Q-function.**

Discussions and Take-aways

- By adaptively integrating Q-aid into RCSL, it is possible to achieve performance that surpasses the maximum trajectory performance of each dataset.
- Depending on the task, a more advanced method that can efficiently evaluate the Q-function’s overgeneralization and provide appropriate Q-aid may be necessary.