Comparison of Offline RL Algorithms in the CartPole Environment

Group 39
Claire Yang (260898597)
`youyou.yang@mail.mcgill.ca`

Abstract

Offline reinforcement learning helps when collecting new data is not possible. This report compares three offline RL methods: Behavior Cloning (BC), Offline Deep Q-Network (DQN), and Conservative Q-Learning (CQL) on three dataset types (expert, random, and mixed) in CartPole. Our results show that each algorithm has different strengths: BC works well with expert data but poorly with random data, Offline DQN performs best with mixed data but fails with expert-only data, and CQL maintains good performance on expert data by being more conservative.

1 Introduction

1.1 Motivations and Contribution

Reinforcement learning often requires collecting new data through trial and error. In many real-world situations like healthcare or robotics, this data collection can be expensive or impossible. Offline RL addresses this problem by learning only from previously collected data without further environment interaction. However, this creates new challenges since algorithms must learn without exploring. Understanding these patterns helps choose the right algorithm based on available data, which is important for practical applications.

This study is done individually. The contributions are: First we create three datasets with different quality levels (expert, random, and mixed). We then compare the performance of the three offline RL algorithms (BC, DQN, and CQL) on these datasets. At last, we analyze why some algorithms fail or succeed with certain datasets.

1.2 Background

Reinforcement learning problems are typically formulated as Markov Decision Processes (MDPs) defined by states $\mathcal{S}$, actions $\mathcal{A}$, transition dynamics $P(s'|s,a)$, reward function $r(s,a)$, and discount factor γ. The goal is to find a policy $\pi(a|s)$ that maximizes expected future rewards. In standard RL, agents interact with the environment to collect data and improve their policy. Offline RL differs by using only a fixed dataset $\mathcal{D} = \{(s,a,r,s')\}$ collected by some previous policy. This creates two major challenges: distribution shift, where the learned policy may need to make decisions in states not covered in the dataset; and value overestimation, where algorithms may incorrectly estimate high values for actions never tried in the dataset. These challenges often cause traditional RL algorithms to fail when applied to offline data.

1.3 Related Work

Several approaches have been developed for offline RL. Behavior Cloning [5] treats the problem as supervised learning, directly copying actions from the dataset without considering long-term rewards. This simple approach works well when data comes from expert demonstrations but struggles with lower-quality data. Deep Q-Network (DQN) [3] was originally developed for online RL but can be adapted to offline settings. However, standard DQN tends to overestimate values for actions not seen in the dataset. Conservative Q-Learning (CQL) [2] addresses the overestimation problem by adding a regularization term that reduces values for out-of-distribution actions.

2 Methodology

2.1 Environment and Dataset

We use the CartPole-v1 environment[1], a simple control task where the goal is to balance a pole by moving a cart left or right. The state space is 4-dimensional and there are 2 possible actions. Each step gives +1 reward, and episodes end upon failure or reaching 500 steps.

We created three datasets with different characteristics:

- **Expert Dataset**: Contains 37,949 transitions from 100 episodes collected using a well-trained agent.
- **Random Dataset**: Contains 2,181 transitions from 100 episodes using a random policy.
- **Mixed Dataset**: Contains 2,180 transitions from 53 episodes, which is collected form 50% expert + 50% random.

Figure 1 shows the distribution of episode rewards and action frequencies. Expert data showing near-optimal performance close to the maximum of 500 steps. Random data shows mostly failures, and mixed data shows both successful and failing episodes. All datasets have balanced action distributions.

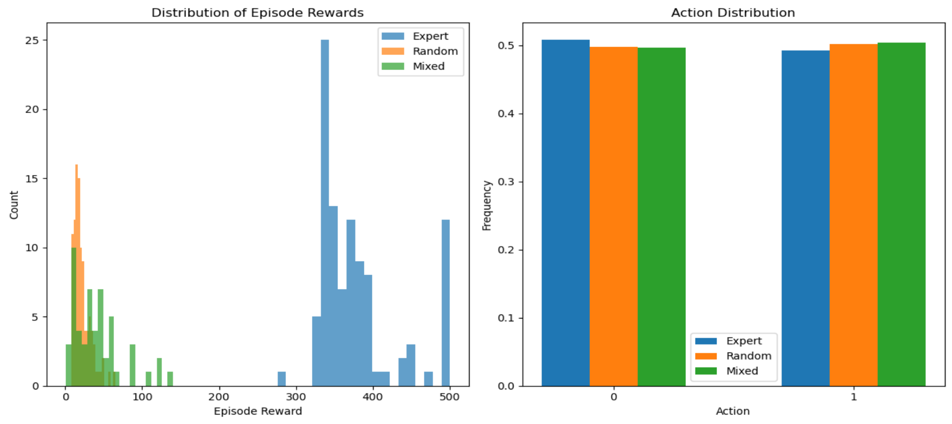

Figure 1: Dataset reward distribution and action frequencies.

2.2 Algorithms and Experimental settings

Behavior Cloning (BC): We implemented a 2-layer MLP (4→128→128→2) with ReLU activations. The model was trained using cross-entropy loss and Adam optimizer (lr=10^{-3}, batch size=64) for 20 epochs. Data was split 80% training, 20% validation. Final evaluation used 20 episodes. Training time: **1.97s**.

Offline DQN: We used the same network architecture as BC. The algorithm implemented double Q-learning with a target network updated every 10 steps. Training settings: discount factor $\gamma = 0.99$, Adam optimizer (lr=10^{-3}), batch size=64, trained for 10,000 gradient updates. Progress was evaluated every 500 updates using 5 episodes. Training time: **6.09s**.

Conservative Q-Learning (CQL): We used the d3rlpy implementation with DiscreteCQLConfig [4]. Hyperparameters: conservative weight (alpha)=1.0, batch size=256. The model was trained for 20,000 steps with evaluation every 2,000 steps using 5 episodes. Final evaluation used 20 complete episodes. Training time: **79.49s**.

3 Experiments

3.1 Results

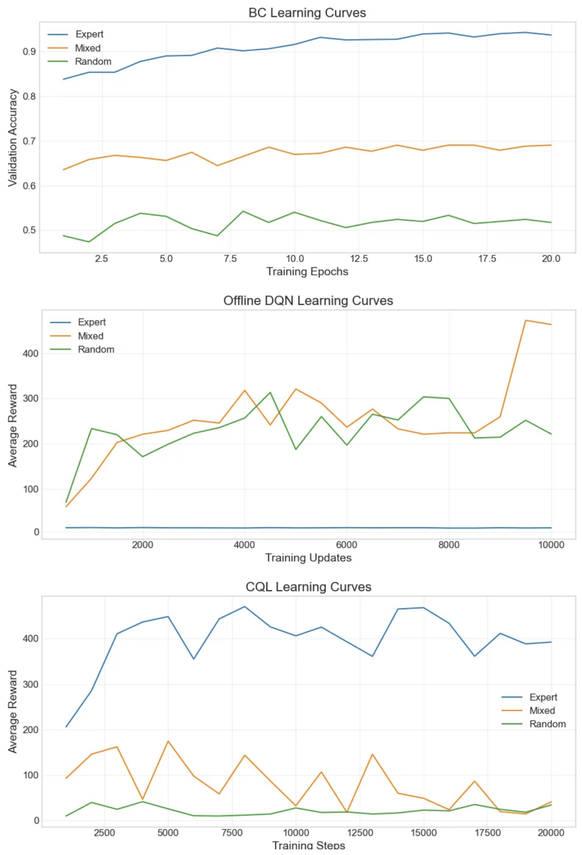

Figure 2: Learning curves for the three algorithms.

Figure 2 shows how each algorithm's performance changes during training. The top graph shows BC's validation accuracy rather than reward, since BC is a supervised learning approach. BC achieves high accuracy (93%) on expert data but performs poorly on random data (barely better than random guessing at 50%).

The middle graph shows a surprising result for Offline DQN: it completely fails on expert data (rewards stay below 10) but performs excellently on mixed data (reaching rewards around 435) and reasonably well on random data. DQN also shows large fluctuations in performance during training.

CQL (bottom graph) performs very differently from DQN. It works well on expert data (reaching rewards around 378) but struggles with mixed data (around 109) and random data (below 30). The CQL learning curve on expert data shows some dips but generally maintains good performance.

Algorithm	Expert	Mixed	Random
BC	402.65 ± 73.72	137.35 ± 110.35	21.05 ± 4.53
Offline DQN	9.45 ± 0.74	435.95 ± 78.14	226.65 ± 51.12
CQL	378.45 ± 86.33	109.05 ± 99.08	27.15 ± 16.80

Table 1: Final average rewards with standard deviation over 20 evaluation episodes.

3.2 Discussion

The results in Figure 3 and Table 1 show patterns for each algorithm.

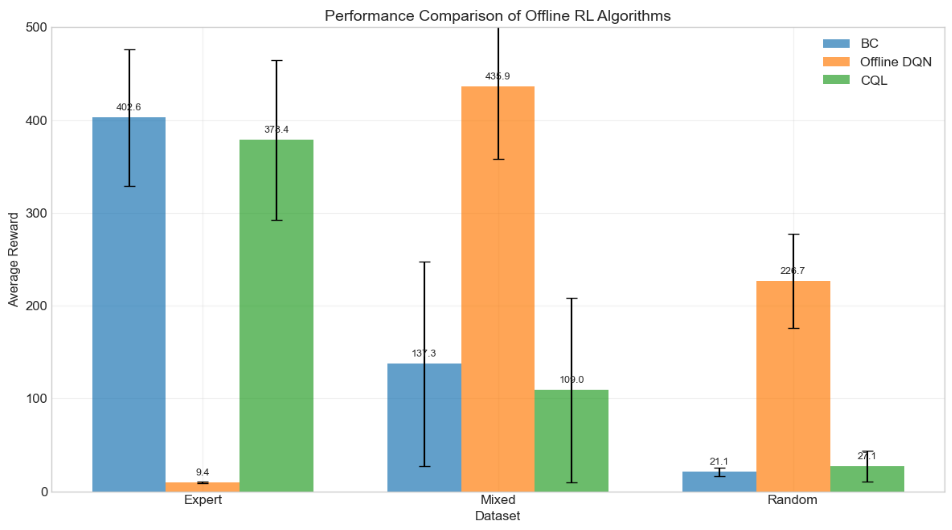

Figure 3: Final performance comparison across all algorithms and datasets.

Behavior Cloning performance decreases with data quality as it simply copies actions without understanding rewards. It works well on expert data but learns meaningless patterns from random data, with mixed data showing high variance in performance.

Offline DQN shows an unexpected pattern: failing on expert data but excelling on mixed data. This stems from distribution shift - when trained only on expert data, DQN overestimates values for unseen actions. Mixed and random datasets help by providing failure examples that prevent this overestimation.

Conservative Q-Learning addresses overestimation and performs well on expert data, but becomes too conservative with mixed and random data. Its cautious approach limits learning from lower-quality data where DQN's optimism works better.

The training times show significant differences: BC (1.97s) is much faster than DQN (6.09s) and especially CQL (79.49s). This makes BC attractive when computation is limited and expert data is available.

These results show that with expert data only, BC or CQL work best; with mixed data (some good, some bad examples), DQN works best; when computation time matters, BC is much faster; and when safety is critical, CQL is more cautious.

4 Conclusion and Future Work

We compared three offline RL algorithms on datasets with different quality levels in the CartPole environment. Our results show that algorithm performance heavily depends on data characteristics. BC works well with expert data but fails with random data. Offline DQN performs best with mixed data but fails with expert-only data due to overestimation. CQL maintains good performance on expert data by being more conservative. In the future, we could test on more complex environments with larger state/action spaces, explore how dataset size affects performance, and test newer offline RL algorithms. Understanding which algorithm works best with which type of data helps make better choices in real-world applications of offline RL.

References

[1] Andrew G. Barto, Richard S. Sutton, and Charles W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(5):834–846, 1983.

[2] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *CoRR*, abs/2006.04779, 2020.

[3] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013.

[4] Takuma Seno and Michita Imai. d3rlpy: An offline deep reinforcement learning library, 2022.

[5] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. *CoRR*, abs/1805.01954, 2018.