Importance of Empirical Sample Complexity Analysis for Offline Reinforcement Learning

Samin Yeasar Arnob
Mila, McGill University,
samin.arnob@mail.mcgill.ca

Riashat Islam
Mila, McGill University,

Doina Precup
Mila, McGill University,
DeepMind

Abstract

We hypothesize that empirically studying the sample complexity of offline reinforcement learning (RL) is crucial for the practical applications of RL in the real world. Several recent works have demonstrated the ability to learn policies directly from offline data. In this work, we ask the question of the dependency on the number of samples for learning from offline data. Our objective is to emphasize that studying sample complexity for offline RL is important, and is an indicator of the usefulness of existing offline algorithms. We propose an evaluation approach for sample complexity analysis of offline RL.

1 Introduction

Reinforcement Learning (RL) is a powerful framework in solving complex problems. However, applying RL to real-world application is tricky as it needs to actively interact with the environment. In many applications (i.e self-driving car, power-system automation, financial trading, medical trials etc.) it can get very expensive or risky to collect samples in-between training. Similar to supervised learning, Offline-RL [3, 13] offers a data-driven alternative approach. Offline-RL leverages previously logged data or expert samples and are trained offline without the need to interact with the environment.

Often it is hard to guarantee the quality of the training dataset. Thus in Offline-RL, it is important to benchmark performance [4] with different types of datasets such as; expert, medium-expert, random etc.; to guarantee a reliable performance despite the quality of training samples. But we do not benchmark offline-RL algorithms under *sample complexity*. In Offline-RL we assume not having any constraint in collecting training dataset. For example, in continuous control tasks [4, 6, 11, 9, 5] RL agents are trained with 1 million training samples. But in real-world applications collecting so many samples may not be possible. For more complex tasks, there is no way to quantify how many training sample it may require for the agent to get trained like an expert. In such scenario, there is no-way to provide performance guarantee without letting the agent to perform in the real-world, which again can be very expensive/risky. Thus we need to construct offline-RL algorithms such that it tries it's best to retain performance even with smaller samples.

Several works have proposed offline RL algorithms on standard benchmark tasks, where the assumption is that certain amount of data is always available for learning policies from offline dataset. However, to our surprise, none of the existing works study training and validation performance for offline RL, given its close approximity to a supervised learning setting.

Offline Reinforcement Learning Workshop at Neural Information Processing Systems, 2021.

It is shown in [1, 10] the offline RL agents exhibit overfitting, i.e., after certain number of gradient updates, their performance starts to deteriorates. [1, 10] restores to online performance evaluation to identify the performance drop and early stopping to avoid overfitting. But "true" offline RL requires offline policy evaluation. In this work we show offline RL agent overfits over the expert dataset very early on when trained with smaller number of training samples, i.e., improvement in minimizing the policy training objective gives a false notion of improvement, whereas policy evaluation on validation dataset, which can be done completely offline, indicate agent's actual performance.

Our key contributions are as follows :

1. We emphasize the importance of sample complexity analysis for offline RL, and compare performance of existing offline RL algorithms by varying the size of training dataset.
2. We propose that existing works should study overfitting and validation performance of offline RL algorithms that can be computed completely offline. Our comparison of the offline evaluation on the validation set replicate the policy performance trend of the online policy evaluation in MuJoCo continuous control tasks. Thus this provides insights on the offline RL algorithms performance, especially important when applied in the real-world applications.
3. Our empirical findings show that while existing offline algorithms can work really well under the standard benchmark size of training samples, the performance of these algorithms is quite different when studies under a low data regime. This indicates that certain algorithms are more likely to overfit than others. Along with data-diversity, sample-complexity analysis further validates agents reliability and robustness.

In this work, we emphasize the importance of sample complexity analysis for offline RL algorithms, which has perhaps been overlooked in existing studies. By ranging from a large data regime to a small data regime, we show that the performance of different offline RL algorithms is not always consistent across benchmark tasks. To further clarify our studies, we propose a training and validation split for offline RL, akin to the basic supervised learning problem, and find that different algorithms have different overfitting properties given the same algorithm complexity in terms of the policy and value functions. This suggests the importance of sample complexity analysis for offline RL, clearly showing that the existing performance metric may not always be a good indicator of the usefulness of an offline RL algorithm, especially when the goal is to take offline RL to real world applications.

2 Preliminaries

We consider learning in a Markov decision process (MDP) described by the tuple (S, A, P, R). The MDP tuple consists of states $s \in S$, actions $a \in A$, transition dynamics $P(s'|s, a)$, and reward function $r \sim R(s, a)$. We use s_t, a_t and $r_t \sim R(s_t, a_t)$ to denote the state, action and reward at timestep t, respectively. A trajectory is made up of sequence of states, action and rewards $\tau \sim (s_0, a_0, r_0, s_1, a_1, r_1, ..., s_T, a_T, r_T)$. For continuous control task we consider an infinite horizon, where $T \sim \infty$ and the goal in reinforcement learning is to learn a policy which maximizes the discounted expected return $\mathbb{E}[\sum_{t=0}^{T} \gamma^t r_t]$ in an MDP. In offline reinforcement learning, instead of obtaining data through environment interactions, we only have access to some fixed limited dataset consisting of trajectory rollouts of arbitary policies.

3 Sample Complexity in Offline RL

Sample Complexity : An important concept for our analysis is to define *sample complexity*. In general, by finding the sample complexity of any algorithm we refer to the number of training samples required to learn a good approximation of the target. But for complex task, especially in infinite state-action space it's not trivial to define this quantity, for our analysis we refer *sample complexity* as to sensitivity of the algorithms to training sample size.

Experiment Setting : In this section, we describe our framework and experimental pipeline for evaluating the sample complexity for different offline RL algorithms. We investigate sample complexity in continous control benchmark tasks, based on the D4RL dateset [4] which is considered

as a standard dataset for most offline RL algorithms. For comparisons, we investigate sample complexity of the following algorithms : *Batch-Constrained deep Q-learning* (BCQ) [6], *Behavior Cloning* (BC, implemented in [12]) and TD3-BC [5]. We run all of our experiments for seed 0-4 and trained for 1M gradient updates. For all the algorithms we use the default network architecture and hyper-parameters. We share our further results in the Appendix.

3.1 How does performance vary based on dataset size?

Given training data, we compare performance for different sizes of the dataset, ranging from $1M$ samples (which is the standard sample size always used), to $100K$ and decreasing to 5000 samples.

For each of the algorithms and given the training data size, we train for 1M training updates and measure the *normalized score* metric as done in *D4RL* [4]. Experimental results comparing performance dependent on the total number of offline samples is presented in figure 1.

Our experimental results show that the performance drops for each Offline-RL algorithm as we reduce the number of training dataset. For all our offline RL algorithms, we compare the performance with *Discriminative Actor Critic* (DAC) [8] - adversarial imitation learning and *Off-policy Adversarial Inverse RL* (OAIRL) [2] method, which use the same number of expert samples but with the advantage of 1 million environment interactions. The advantage of environment interactions makes the comparison unfair. But the idea is to show, even with smaller expert samples adversarial imitation and IRL methods manages to get consistent performance. Comparetive experiments on these algorithms has proven to be significant later in the paper to support our claim that validation performance always correlates with policy's actual online evaluation improvement discussed in 3.2.1 (further experiments are in Appendix A.2).

While this is a result that one would typically expect, we find an interesting phenomenon in our results. Note that the performance varies for each algorithm depending on the training data size. For example, while the recent state-of-the-art algorithm TD3-BC performs significantly better for 1M training sample, this algorithm is in fact worse for 5000 samples. This phenomenon can be seen in almost all of our experiment in figure 1 (except in $1(f)$), where even though TD3-BC performs best for 1M standard sample size, it is the worst performing algorithm as we reduce the size of the dataset. The reason is due to the MSE regularization term in it's actor loss dictates the actor gradient update and thus overfits very easily with smaller training samples and we proof our hypothesis through validation performance in following section 3.2. We also see the similar trend in IQL's [7] performance but the reason is not so apparent, we need further experiments to hypothesize or come to a solid conclusion. This tells us that the performance of each of these algorithms can vary significantly, and comparisons are not always consistent, as to the best performing algorithm, depending on the training dataset size. This is exactly why we can not guarantee consistent performance with abundant training dataset.

In offline RL benchmark we compare algorithms on different categories of training samples i.e. expert, medium, medium-expert, random and the intuition is that, in real-world application we can not always guarantee to collect optimal-expert, thus we want to pick an algorithm that guarantees a better performance for any kind of dataset. Similarly, for any real world application there is no way to quantify the *"sufficient amount"* of data that we must collect so that training agent can provide expected performance. Thus we consider *sample complexity*, sensitivity of algorithms performance to training dataset size, as a metric to evaluate the offline-RL performance. An Offline-RL algorithm that give better performance with smaller training samples are more reliable in real-world application than the others. We find the sample complexity analysis to be a very useful metric to evaluate the reliability of Offline-RL algorithm.

3.2 Does existing offline RL algorithms have overfitting phenomenon?

We conjecture that the phenomenon observed in figure 1 hints to an overfitting phenomenon for offline RL algorithms. For offline RL, we consider an agent is overfitted over the training dataset when the training objective reduces the divergence between the policy action and expert action over the observed training states and yet fails to provide performance improvement in the oracle (online evaluation).

We emphasize that, to the best of our understanding, no previous works studied similar complexity analysis for different offline RL algorithms. Since most prior works only evaluate performance for

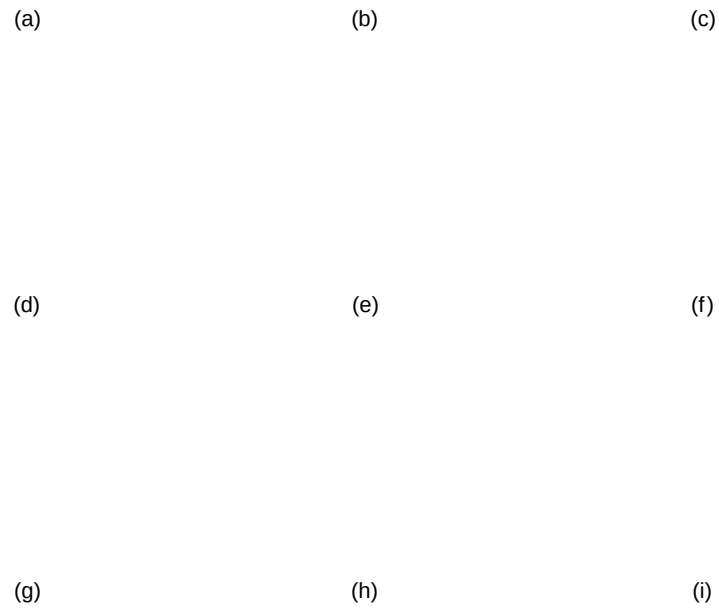

Figure 1: Performance Comparison (D4RL Normalized Score) of DAC with of ine-RL Varying Expert data.

1M sample sizes on D4RL benchmarks, we emphasize that this is not always a good measure, as we see in our analysis in this section. In the subsequent sections, we provide a measure to study the over tting phenomenon in of ine RL, and want to emphasize the readers, that since the goal of of ine RL is similar to supervised learning, such characterization of over tting and sample complexity is necessary for any of ine RL algorithm empirically.

3.2.1 Evaluating Over tting in Of ine RL

To prove our over tting hypothesis, similar to supervised learning, we propose to use separate validation dataset. We held-out 2000 expert trajectories (which is approximately 2000 ,000 $\{s_V, a_V, r_V, s'_V\}$ tuples) from the D4RL dataset [4] during training. We perform evaluation over the validation dataset, which provide an unbiased and the true progress of the learning agent.

Metric on Training and Validation Dataset : We provide a metric for measuring training and validation performance in of ine RL, akin to the standard loss typically studied in supervised learning. As an evaluation criterion, we use the Mean-Square-Error (MSE) loss between expert-action a_V and policy-action $\pi_\mu(s_V)$ as to measure actor's deviation from the expert. Note that we use MSE instead of the KL divergence metric here, since most of ine RL algorithms that we study are based on deterministic policies, as typically in BCQ [6] and other algorithms.

Figure 2 shows the over tting phenomenon for different of ine RL algorithms. We plot the MSE loss over the training and validation dataset, and vary the sample size. For each algorithm, we train up to 1M iterations (as typically done in standard experiments), but with different sample sizes. We nd that as the sample size decreases, the difference between training and validation

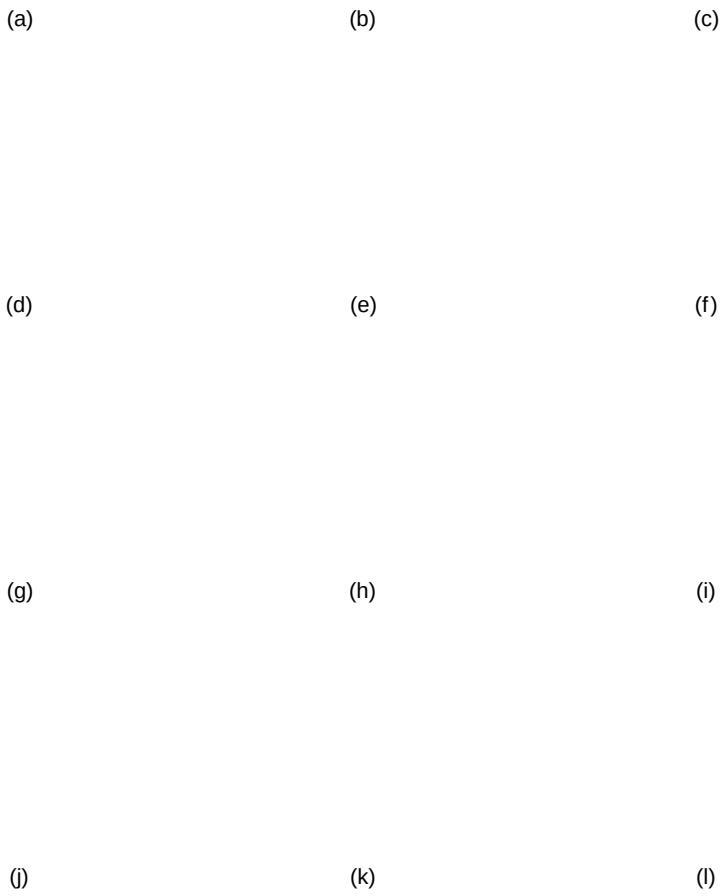

Figure 2: MSE loss between $¼_¹(s_E)$ and a_E for different Of ine-RL algorithms over the training (orange) and the validation (blue) dataset as we vary number of training expert samples

error increases signi cantly, which shows that the algorithms are more likely to over t (due to a more complex policy class compared to the dataset size).

We get a good generalization in estimation when we make improvement in estimating both the training and validation dataset. We know our training model is over tting over the training the dataset when the training loss gets reduced with each gradient update but the performs worse on the validation set. For 1 million expert samples, algorithms performs lowest validation error. For 5000 training dataset the Actor gets the lowest training (orange) error but gets the highest validation error. It suggests that the Actor over ts the expert samples and we see the consequence in the policy performance (gure 1).

The largest deviation in training-validation performance in found for TD3-BC. This con rms our hypothesis for TD3-BC's performance drop with smaller training sample discussed in section 3.1 and consolidates the fact that validation performance and Of ine policy evaluation are correlated. We further show how the actors training loss gives a false sense of improvement in appendix A.2.

3.2.2 Validation Performance of Of ine RL algorithms

This section further con rms our conjecture above - the validation dataset is a useful metric to truly measure the performance improvement for different algorithms. Figure 3 further con rms this. We plot the cumulative performance return over 1M training iterations, for each of the sample size of the dataset over the HalfCheetah environment for different algorithms. We nd that the validation performance is consistent with the cumulative return metric - for example, in gure 3 (b) and 3(f) for the TD3-BC algorithm, the performance improvement is highest when validation loss is the lowest; similarly for sample size of 5000, the validation error for TD3-BC is highest which leads to the lowest performance of this algorithm, as measured by the cumulative returns. Without evaluations in between training, we can further guarantee of an improvement using the validation performance. The evaluation on the validation dataset provide a clear indication whether training agent is improving or diverging from expected behavior.

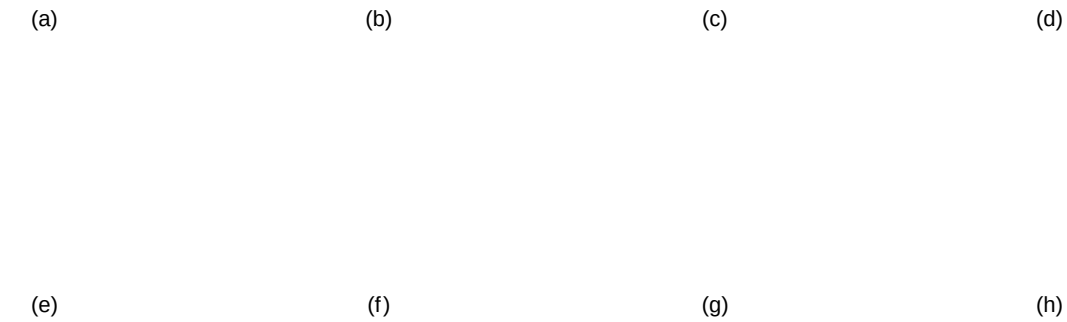

Figure 3: Performance curve evaluated over 1M gradient updates of (a) DAC, (b) TD3-BC, (c) BCQ, (d) BC and corresponding MSE loss (e-h) between ¼(s_V) and a_V over the validation dataset as we vary number of training dataset.

3.2.3 Further Discussion on the Validation Performance

In the gure 4 we see a clear deviation in actor performance on the validation set as we decrease the training samples size. But we do not nd any signi cant change in DAC's estimation with expert sample complexity. Despite providing bad estimation compared to of ine-RL algorithms, DAC performs better. The of ine-RL algorithms are provided with expert samples and are compelled to mimic the expert behavior. And since the expert samples are collected from the same expert, validation estimation are co-related with algorithms performance. We do not have access to optimal expert ¼¤, rather collected expert trajectories are sub-optimal, thus DAC still performs better without proving good validation performance. Thus under sub-optimal expert, validation is most useful when we compare algorithms that mimics expert.

Figure 4: Compare Validation loss of different learning algorithms

4 Conclusion

We investigated the sample complexity of different of ine RL algorithms, by varying the size of the training dataset for the same training procedure for each of the algorithms. Our experimental studies leads to a surprising nding : the cumulative return performance as typically shown in standard of ine RL algorithms over 1M dataset size, is not always a good indicative measure of whether the algorithm is robust under smaller dataset. Our experiment with smaller training dataset shows, the performance of the state of the art of ine RL algorithms fall dramatically since the objective function do not consider improving sample complexity.

The key contribution of our work is therefore to provide an important message for studying of ine RL algorithms empirically. We emphasize that studying sample complexity of of ine RL algorithms is important, to truly evaluate the performance comparison for each algorithm. We show that current of ine-RL algorithms over t with smaller dataset and the best performing algorithm can perform very poorly under such condition. Thus to make Of ine-RL algorithm more reliable in real-world application, where collecting data is non-trivial and no way to quantify the required amount of the data to achieve expert like performance, we need to consider model over tting into account. We show how training loss can be misleading. Unlike recent studies [1, 10] that use online performance to evaluate over tting, we propose a complete of ine evaluation of the policy leveraging a validation dataset to foresee if agent is improving. Improving performance in validation set shows a consistent online performance improvement in all our experiments. Thus in real-world applications (i.e. self-driving car, drone auto-pilot, medical trails, controlling power system etc.), where a badly trained agent can be extremely risky or costly to evaluate, a validation performance can provide performance improvement guarantee.

References

[1] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. “Striving for Simplicity in Off-policy Deep Reinforcement Learning”. In: CoRRabs/1907.04543 (2019). arXiv:1907.04543. URL: http://arxiv.org/abs/1907.04543 .

[2] Samin Yeasar Arnob. “Off-Policy Adversarial Inverse Reinforcement Learning”. In: CoRR abs/2005.01138 (2020). arXiv:2005.01138. URL: https://arxiv.org/abs/2005.01138 .

[3] Damien Ernst, Pierre Geurts, and Louis Wehenkel. “Tree-based batch mode reinforcement learning”. In: Journal of Machine Learning Research 6 (2005), pp. 503–556.

[4] Justin Fu et al. “D4RL: Datasets for Deep Data-Driven Reinforcement Learning”. In: CoRR abs/2004.07219 (2020). arXiv:2004.07219. URL: https://arxiv.org/abs/2004.07219 .

[5] Scott Fujimoto and Shixiang Shane Gu. “A Minimalist Approach to Of ine Reinforcement Learning”. In: CoRRabs/2106.06860 (2021). arXiv:2106.06860. URL: https://arxiv.org/abs/2106.06860 .

[6] Scott Fujimoto, David Meger, and Doina Precup. “Off-Policy Deep Reinforcement Learning without Exploration”. In: CoRRabs/1812.02900 (2018). arXiv:1812.02900. URL: http://arxiv.org/abs/1812.02900 .

[7] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Of ine Reinforcement Learning with Implicit Q-Learning . 2021. arXiv:2110.06169 [cs.LG] .

[8] Ilya Kostrikov et al. “Addressing Sample Inef ciency and Reward Bias in Inverse Reinforcement Learning”. In: CoRRabs/1809.02925 (2018). arXiv: 1809.02925. URL: http://arxiv.org/abs/1809.02925 .

[9] Ilya Kostrikov et al. “Of ine Reinforcement Learning with Fisher Divergence Critic Regularization”. In: CoRRabs/2103.08050 (2021). arXiv: 2103.08050. URL: https://arxiv.org/abs/2103.08050 .

[10] Aviral Kumar et al. A Work ow for Of ine Model-Free Robotic Reinforcement Learning . 2021. arXiv: 2109.10813 [cs.LG] .

[11] Aviral Kumar et al. “Conservative Q-Learning for Of ine Reinforcement Learning”. In: CoRR abs/2006.04779 (2020). arXiv: 2006.04779. URL: https://arxiv.org/abs/2006.04779 .

[12] Aviral Kumar et al. “Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction”. In: CoRRabs/1906.00949 (2019). arXiv: 1906.00949. URL: http://arxiv.org/abs/1906.00949.

[13] Sascha Lange, Thomas Gabel, and Martin Riedmiller. “Batch reinforcement learning”. In: Reinforcement learning . Springer, 2012, pp. 45–73.

A Appendix

A.1 Performance curve of different algorithm

In gure 5 we plot the mean performance of the algorithms for seeds 0-4 with 100% con dence interval over 1 million gradient updates. We compare the performance of each algorithm varying training sample size on MuJoCo control tasks.

A.2 Compare Training and Validation Actor Evaluation

We compare the actor's training loss and actor's validation loss (MSE($\pi_\mu(s_V), a_V$)) over 1 million gradient updates. It clearly shows how actors training loss (blue) gives a false sense of imporvement.

For example, from the experiments conducted on IQL [7] (gure 6 (m ¡ p)) shows, we nd the actor training loss (blue) to be declining as we update the actor network for all our experiment, even when we reduce the number expert training dataset (from columns right to left). In idea case, this indicates the actor's performance should be improving for all experiments. But the corresponding policy evaluation in online from gure 5(g) does not approve that.

Thus we use the validation set to perform the policy-action deviation from the experts. For larger expert dataset in the training assures a declining validation loss curve but the validation loss increases for smaller dataset, and proves that smaller expert over ts the policy. We see the similar pattern in all our of ine RL experiments.

For DAC and OAIRL, since the number of expert data has negligible impact (gure 5(a ¡ f)), the validation performance (gure 6 (a ¡ h)) is always decreases with the actor's network gradient update.

(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k) (l)

(m) (n) (o)

(p) (q) (r)

Figure 5: Performance Comparison (D4RL Normalized Score) of DAC with of ine-RL Varying Expert data.