ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

Adaptive Behavior Cloning Regularization for Stable Offline-to-Online Reinforcement Learning

Yi Zhao[1*], Rinu Boney[2 *], Alexander Ilin[2] , Juho Kannala[2], Joni Pajarinen[1,3]

1- Aalto University - Department of Electrical Engineering and Automation

2- Aalto Universiity - Department of Computer Science - Finland

3- Technical University Darmstadt - Department of Computer Science - Germany

Abstract. Offline reinforcement learning, by learning from a fixed dataset, makes it possible to learn agent behaviors without interacting with the environment. However, depending on the quality of the offline dataset, such pre-trained agents may have limited performance and would further need to be fine-tuned online by interacting with the environment. During online fine-tuning, the performance of the pre-trained agent may collapse quickly due to the sudden distribution shift from offline to online data. We propose to adaptively weigh the behavior cloning loss during online fine-tuning based on the agent's performance and training stability. Moreover, we use a randomized ensemble of Q functions to further increase the sample efficiency of online fine-tuning by performing a large number of learning updates. Experiments show that the proposed method yields state-of-the-art offline-to-online reinforcement learning performance on the popular D4RL benchmark.

1 Introduction

Offline or batch reinforcement learning (RL) [1, 2] deals with the training of RL agents from fixed datasets generated by possibly unknown behavior policies, without any interactions with the environment. However, the performance of trained policies will be limited by the quality of the offline dataset and it is often necessary or desirable to fine-tune them by interacting with the environment, which is called offline-to-online reinforcement learning. In practice, offline RL methods often fail during online fine-tuning by interacting with the environment. This offline-to-online RL setting is challenging due to: (i) the sudden distribution shift from offline data to online data. This could lead to severe bootstrapping errors which completely distorts the pre-trained policy leading to a sudden performance drop from the very beginning of online fine-tuning, and (ii) constraints enforced by offline RL methods on the policy to stay close to the behavior policy. While these constraints help in dealing with the sudden distribution shift they significantly slow down online fine-tuning from newly collected samples.

We propose to adaptively weigh the offline RL constraints such as behavior cloning loss during online fine-tuning. This could prevent sudden performance collapses due to the distribution shift while also enabling sample-efficient learning from the newly collected samples. We propose to perform this adaptive weighing according to the agent's performance and the training stability. We

*Equal contribution

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

start with TD3+BC, a simple offline RL algorithm recently proposed by [3] which combines TD3 [4], a widely used off-policy RL algorithm, with a simple behavior cloning loss, weighted by an α hyperparameter. We adaptively weigh the α hyperparameter using a control mechanism similar to the proportional derivative (PD) controller. The α value is decided based on two components: the difference between the moving average return and the target return (proportional term) as well as the difference between the current episodic return and the moving average return (derivative term).

We demonstrate that these simple modifications lead to stable online fine-tuning after offline pre-training on datasets of different quality. We also use a randomized ensemble of Q functions [5] to further improve the sample-efficiency. We attain state-of-the-art online fine-tuning performance on locomotion tasks.

2 Stable Offline-to-online Reinforcement Learning

2.1 Offline Pre-training

Offline RL aims to learn a policy from pre-collected fixed datasets without interacting with the environment [1, 2, 6, 7, 8, 9]. [3] proposes TD3+BC, a simple offline RL algorithm that regularizes policy learning in TD3 with a behavior cloning loss that constraints the policy actions to stay close to the actions in the offline dataset $\mathcal{D}$. This is achieved by adding a behavior cloning term to the policy loss:

$$\pi_\theta = \arg\max_\theta \mathbb{E}_{(\mathbf{s},\mathbf{a})\sim\mathcal{D}} \left[\bar{Q}(\mathbf{s}, \pi_\theta(\mathbf{s})) - \alpha(\pi_\theta(\mathbf{s}) - \mathbf{a})^2 \right] \tag{1}$$

where α is a weighing hyperparameter and

$$\bar{Q}(\mathbf{s}, \pi_\theta(\mathbf{s})) = \frac{Q(\mathbf{s}, \pi_\theta(\mathbf{s}))}{\frac{1}{N}\sum_{\mathbf{s}_i,\mathbf{a}_i} Q(\mathbf{s}_i, \mathbf{a}_i)}$$

normalizes the Q values which help in balancing both losses. The sum in the denominator is taken over a mini-batch and the gradients do not flow through the critic term in the denominator.

Furthermore, we propose to use an ensemble of Q functions to better deal with the distribution shift from offline pre-training and to improve the sample-efficiency of online fine-tuning. We use the Randomized Ensembled Double Q-learning (REDQ) method proposed by [5] to learn an ensemble of critic networks. Specifically, we maintain an ensemble of N critic networks and randomly samples M networks for each critic update. Given a mini-batch $\mathcal{B}$ of B transitions $(\mathbf{s}, \mathbf{a}, r, \mathbf{s}')$, all critic networks in the ensemble are updated as:

$$\nabla_{\phi_i} \frac{1}{|B|} \sum_{(\mathbf{s},\mathbf{a},r,\mathbf{s}')\in\mathcal{B}} \left(Q_{\phi_i}(\mathbf{s}, \mathbf{a}) - r - \gamma \min_{i\in\mathcal{M}} Q_{\phi_i}(\mathbf{s}', \mathbf{a}') \right)^2 \tag{2}$$

where $\mathcal{M}$ is a random subset of M critic networks, Q_{ϕ_i} is the i-th Q function parameterized by ϕ_i. We observed that taking the minimum over randomly

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

sampled M networks to calculate the target is better than taking the average or minimum over all N networks.

2.2 Online Fine-tuning with Adaptive Regularization

RL agents trained from offline data tend to have limited performance and would further need to be fine-tuned online by interacting with the environment. During online fine-tuning, the performance of the pre-trained agent may collapse quickly due to the sudden distribution shift from offline data to online data, as shown in Fig. 1 (with $\alpha = 0$). Keeping the constrain used in offline pre-training, such as in Equation 1, could mitigate the collapse. However, this will force the policy to stay close to the behavior policy (used to collect the dataset), thus leads to slow improvement.

In the TD3+BC algorithm we consider in this paper, a hyperparameter α is used to balance the RL objective and the behaviour cloning term which constrains the policy to stay close to the behavior policy (see Equation 1). We use α_{off} and α_{on} to distinguish the α hyperparameter value used during offline and online training respectively. By default, we use $\alpha_{\text{off}} = 0.4$ aligned with the TD3+BC paper. In Fig. 1, we present the influence of α_{on} on the TD3+BC during fine-tuning by trying different values of α_{on} from $[0.0, 0.1, 0.3]$. We can clearly see that using the behavior cloning loss with proper α_{on} enables stable fine-tuning. However, the value of α_{on} depends on the quality of the offline dataset and has significant influence of the fine-tuning performance. For example, $\alpha_{\text{on}} = 0$ fits well on the Hopper-Random task while causes immediate collapse on Hopper-Medium and Hopper-Medium-Expert tasks.

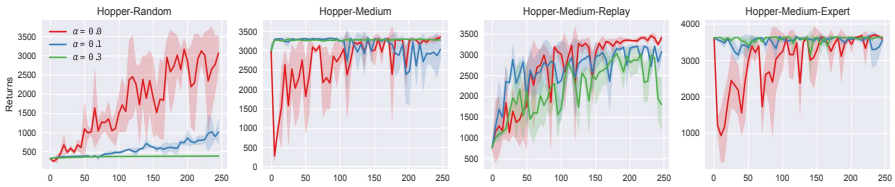

Fig. 1: Results of online fine-tuning on the D4RL benchmark using TD3+BC with different α_{on} hyperparameters. We plot the mean and standard deviation across 3 runs. Using the behavior cloning loss with proper α_{on} enables the stable fine-tuning. But the optimal value of α_{on} differs between datasets.

In our experiments, we found that if the offline dataset has narrow distribution or if the policy has already converged to a desired performance (comparable to the expert), it is usually beneficial to maintain a higher α_{on}. When the data distribution is broader or when we still need to improve the agent by a large margin, a smaller α_{on} works better. However, during experiments, we can not find a single α_{on} that is suitable for all tasks and its value needs to be tuned carefully per task, which makes this method hard to be used in practice.

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

To solve this problem, we propose to automatically adapt the weight of the behavior cloning loss according to two factors: (i) the difference between the moving average return R_{avg} and the target return R_{tar}, and (ii) the difference between the current episodic return R_{cur} and the moving average return R_{avg}. We adaptively change the α_{on} as:

$$\alpha_{\text{on}} \leftarrow \alpha_{\text{on}} + \Delta(\alpha_{\text{on}}) = \alpha_{\text{on}} + (K_P \cdot (R_{\text{avg}} - R_{\text{tar}}) + K_D \cdot \max(0,\ R_{\text{avg}} - R_{\text{cur}})) \quad (3)$$

where we constrain α_{on} between 0 and 0.4 (the value used during offline pre-training). R_{cur} and R_{avg} are normalized following the return normalization procedure used in D4RL [10]. R_{tar} is the target episodic return, which we set as 1.0 (corresponding to the expert policy) for all tasks. K_P controls how fast we decrease the α_{on} according to current performance and K_D determines how fast we increase the α_{on} when the performance drops. Intuitively, when the agent's performance reaches the target episodic return, we try to maintain it during fine-tuning while decrease the α_{on} to allow the agent improving further when the agent's performance is low. The second term increases the α_{on} when performance drops during training to mitigate performance collapse. Equation 3 allows for adaptive weighing of the behavior cloning loss throughout online fine-tuning. It automatically adjusts the constraint enforced by the behavior cloning loss. Both K_P and K_D are tuned on the Hopper-Random and Hopper-Medium tasks (see section 3) via grid search and keep fixed in rest tasks ($K_P = 3e-5$, $K_D = 1e-4$).

3 Experiments

The goal of our experiments is to evaluate the stability and sample-efficiency of the proposed algorithm on online fine-tuning after offline pre-training on datasets of different quality. We evaluate our algorithm on the D4RL benchmark [10], which includes three locomotion environments (HalfCheetah, Hopper, and Walker) and each environment consists of four offline datasets: Random, Medium, Medium-Replay, Medium-Expert.

In Figure 2, we compare our algorithm, called REDQ+AdaptiveBC (REDQ [5] with adaptive behavior cloning), with two state-of-the-art offline-to-online RL algorithms, Advantage Weighted Actor-Critic (AWAC) [11] and Balanced Replay [12], and two baseline methods, TD3 with finetuning (TD3-ft) and REDQ. AWAC implicitly constraints the policy network to stay close to the behavior policy. Balanced Replay [12] prioritizes near-on-policy samples from the replay buffer. For a fair comparison, we reimplement this algorithm based on TD3+BC while ensuring that we are able to reproduce the original results. TD3-ft is the standard TD3 algorithm [4] that was pre-trained offline using TD3+BC [3]. REDQ [5] is an RL method trained from scratch, without any access to the offline data. This baseline emphasizes the importance of offline pre-training and online fine-tuning.

During offline pre-training, all algorithms are pre-trained on the offline dataset for one million gradient steps. After pre-training, we fine-tune the agents for

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1.
Available from http://www.i6doc.com/en/.

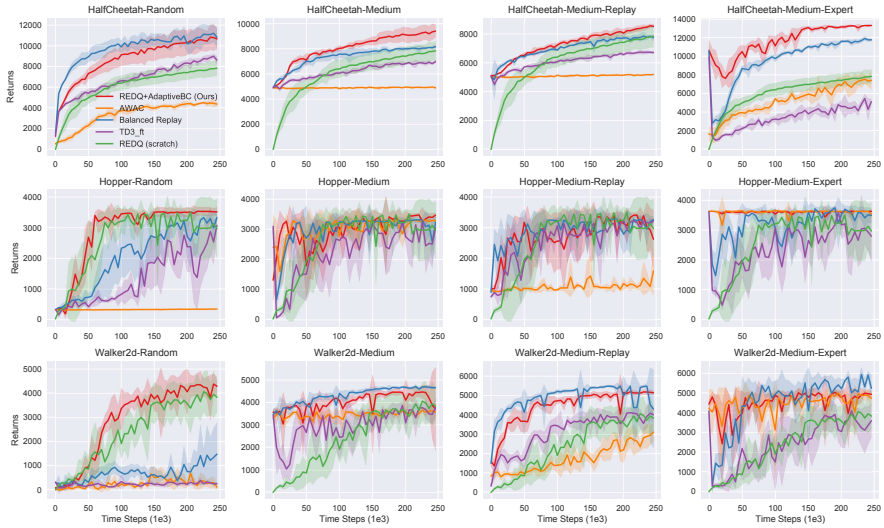

Fig. 2: Results of online fine-tuning on the D4RL benchmark. We plot the mean and standard deviation across 5 runs. Our REDQ+AdaptiveBC method attains performance competitive to the state-of-the-art. Our method is able to consistently improve the pre-trained agent during fine-tuning without suffering from dramatic performance collapse at the beginning of training.

250,000 time steps by interacting with the environment. We evaluate the agent every 5000 time steps and each evaluation consists of 10 episodes. We attain performance competitive to the state-of-the-art in this benchmark with our method stably improving the performance during online fine-tuning.

We outperform REDQ and TD3-ft on all tasks by a large margin. Compared to AWAC, our method consistently improves the pre-trained policy and outperforms or matches other methods on all tasks, among different environments and different datasets. Compared to Balanced Replay, our method does not collapse dramatically on all three Medium-Expert tasks. Furthermore, we want to stress the simplicity of our methods. Balanced Replay needs to learn an additional network to estimate the "closeness" of the sampled data with the current policy, and AWAC is incompatible with other offline pre-training algorithms. While our algorithm can be modified within lines of code based on the TD3+BC and it is straightforward to apply our methods to other existing offline pre-training algorithms, such as Conservative Q-Learning (CQL) [13].

4 Conclusion

We consider the problem of offline-to-online RL where an agent is first pre-trained on offline data (collected by a possibly unknown behavior policy) and the agent is then fine-tuned online by interacting with the environment. This is

ESANN 2022 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 5-7 October 2022, i6doc.com publ., ISBN 978287587084-1. Available from http://www.i6doc.com/en/.

desirable as pre-trained agents may have limited performance depending on the quality of the offline dataset. Offline-to-online RL is challenging due to the sudden distribution shift from offline data to online data, and also the constraints enforced by offline RL algorithms (such as a behavior cloning loss) during pre-training. In this paper, we propose a simple mechanism to adaptively weigh a behavior cloning loss during online fine-tuning, based on agent's performance and training stability. We further combine a randomized ensemble of Q networks to enable sample-efficient online fine-tuning performance. We achieve performance competitive to the state-of-the-art online fine-tuning methods on twelve locomotion tasks in the popular D4RL benchmark.

Acknowledgements The authors acknowledge the computational resources provided by the Aalto Science-IT project.

References

[1] Sascha Lange, Thomas Gabel, and Martin Riedmiller. Batch reinforcement learning. In *Reinforcement learning*, pages 45–73. Springer, 2012.

[2] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[3] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *arXiv preprint arXiv:2106.06860*, 2021.

[4] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pages 1587–1596. PMLR, 2018.

[5] Xinyue Chen, Che Wang, Zijian Zhou, and Keith W Ross. Randomized ensembled double Q-learning: Learning fast without a model. In *International Conference on Learning Representations*, 2021.

[6] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, pages 104–114. PMLR, 2020.

[7] Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. *arXiv preprint arXiv:1906.04733*, 2019.

[8] Noah Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, Nicolas Heess, and Martin Riedmiller. Keep doing what worked: Behavior modelling priors for offline reinforcement learning. In *International Conference on Learning Representations*, 2020.

[9] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062. PMLR, 2019.

[10] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[11] Ashvin Nair, Murtaza Dalal, Abhishek Gupta, and Sergey Levine. Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

[12] Seunghyun Lee, Younggyo Seo, Kimin Lee, Pieter Abbeel, and Jinwoo Shin. Offline-to-online reinforcement learning via balanced replay and pessimistic q-ensemble. *arXiv preprint arXiv:2107.00591*, 2021.

[13] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020.