ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

Automatic Trade-off Adaptation in Offline RL

Phillip Swazinna[1,2], Steffen Udluft[1], and Thomas Runkler[1,2]

1) Siemens Technology - Data Analytics & Artificial Intelligence
Otto-Hahn-Ring 6, Munich, Germany

2) TU Munich - School of Computation, Information, and Technology
Boltzmannstrasse 3, Garching, Germany

Abstract. Recently, offline RL algorithms have been proposed that remain adaptive at runtime. For example, the LION algorithm [1] provides the user with an interface to set the trade-off between behavior cloning and optimality w.r.t. the estimated return at runtime. Experts can then use this interface to adapt the policy behavior according to their preferences and find a good trade-off between conservatism and performance optimization. Since expert time is precious, we extend the methodology with an autopilot that automatically finds the best parameterization of the trade-off, yielding a new algorithm which we term AutoLION.

1 Introduction & Related Work

Over the past few years, offline RL has become a popular field of reinforcement learning research, since it promises to alleviate one of the most pressing issues when trying to apply RL methods to real-world (i.e. potentially physical) systems: Online environment interaction. Direct interaction is often prohibited in real systems, since as opposed to e.g. simulated Atari video games, they incur significant (opportunity) costs and due to potential safety violations. Offline RL methods such as [2, 3] thus constitute a large step towards broader applicability of reinforcement learning methodologies in practice, since they demonstrate the ability to learn purely from static, pre-collected datasets.

A remaining issue is, that each of the proposed algorithms in one way or the other takes a trade-off between optimizing for return only (i.e. pure RL) and regularizing the policy towards the dataset distribution, since otherwise trained policies would exploit the return estimating models and transfer badly to the real system. The problem is, that nobody can with certainty determine the amount of regularization that is best, i.e. how strictly does the policy need to be regularized given the dataset remains an open question (depending on the concrete form of this regularization, different terms such as conservatism, pessimism, risk-avoidance, uncertainty-avoidance, reconstruction penalty, behavior constraint, proximity, etc. have been proposed).

Recently, [4] proposed that offline RL policies should be trained to be adaptive at runtime, i.e. remain adaptive after the training phase is over. The authors propose to maintain a distribution of possible MDPs, which can then at runtime be narrowed down to perform actions that are estimated to be optimal in the currently believed MDP. [1, 5] take this a step further and train policies

ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

that can adapt their level of regularization after training. While [5] is developed for discrete action spaces, such as Atari video games, LION [1] is designed for continuous control environments such as the industrial benchmark (IB) [6] or MuJoCo robotics locomotion tasks. LION enables human expert users to make the concrete trade-off conditioning hyperparameter choice in order to provide them with a utility instead of completely automating the regularization trade-off choice as in [5]. While this is a valuable feature in practice, some users would likely welcome the option to delegate this task in some situations, such as when they are supervising many systems simultaneously, allowing them to better focus on the problematic cases. Our goal is therefore to extend LION with an autopilot mode that automatically chooses the trade-off at runtime.

2 Preliminaries: LION

The LION (Learning in Interactive Offline eNvironments) algorithm was proposed in [1], introducing a model-based offline RL method to train policies that remain trade-off adaptive after training has conceded. Instead of training policies for a fixed trade-off between performance and proximity to the dataset-generating policy (a.k.a. behavioral policy), it samples the trade-off parameter randomly for each starting state of a trajectory during training and conditions the policy on it, enabling policies to learn the entire range from pure behavior cloning over regularized RL up to “pure” (unregularized) RL.

The algorithm trains an ensemble of K (potentially recurrent) transition models $\hat{T}^k(\cdot|s,a),\ k=0,...,K-1$ that predict the future states and rewards in an environment based on past states and actions. It then performs “imagined” trajectory rollouts with the target policy through the trained transition models as environment surrogates and optimizes the resulting return estimate by back-propagation through time. At the same time, the target policy actions in these trajectories are compared with the actions of a learned model of the behavioral policy $\beta(\cdot|s)$, in order to regularize the policy towards the known state-action space. The trade-off between the two components is in this case determined by the hyperparameter λ, which the policy $\pi_\theta(\cdot|s,\lambda)$ also conditions on:

$$L(\theta) = \mathbb{E}_{s_0\sim\mathcal{D},\ \lambda\sim B(c,d)} \sum_{t=0}^{H} \gamma^t \left[\lambda e(s_t, \pi_\theta(s_t,\lambda)) - (1-\lambda) p(\beta(s_t), \pi_\theta(s_t,\lambda))\right] \quad (1)$$

where $\mathcal{D}$ denotes the initial dataset, $B(b^\alpha, b^\beta)$ is the Beta distribution from which λ is sampled, $e(s,a) = \min_k r(\hat{T}^k(s,a))$ denotes the minimum reward prediction by any of the ensemble members, and $p(a_\beta, a_\pi) = \frac{1}{D}\sum_d (a^d_\beta - a^d_\pi)^2$ is the penalty term to regularize the policy (D is the action dimensionality).

3 Automatic trade-off search with LION (AutoLION)

The LION methodology has been proposed in order to provide expert practitioners with a utility: Instead of automating them away, users can benefit from

ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

a still highly autonomous system, yet also have the possibility to interact with the policy and alter its behavior on a higher level of abstraction, by handpicking λ at runtime, depending on their observations and knowledge about the system. However, we find that practitioners would still benefit from a solution that integrates a more autopilot-like behavior - since expert time is always costly and limited, and since users likely look after more than just a single system, it could be beneficial if λ could be tuned automatically (i.e. choose a specific value $\lambda^t \in [0, 1]$) most of the time, and only in critical situations or whenever the user desires they would take over. We thus propose strategies how to find well performing λ values at runtime.

Metrics - before we introduce the strategies to find λ at runtime, we define metrics by which to evaluate them. Depending on the concrete system, different criteria may be of concern: In safety critical systems for example the metric has to reflect that the most important factor is to not enter potentially dangerous system states, while in other systems it is most important to find a good value as quickly as possible, since evaluations incur large opportunity cost as the system cannot be productively used. In many other systems it is likely most important to underperform the behavioral policy as little and as few times as possible in order to facilitate trust in the method and to enable continuous productive use. Finally it could be most important to actually find the single best λ value, since the policy will run for a very long time and any underperformance in the search phase is easily offset by performing better in the long run.
We consider the metrics Final Return (R), as well as Return under Budget (RUB) indicating the best return possible with a limited budget, Mean Behavioral Regret (MBR) representing the average underperformance during the search phase with respect to the behavioral, and Mean Optimal Regret (MOR) which is the average underperformance in the search phase compared to the best λ value.

Search - we compare different search strategies to build AutoLION, that aim to optimize different performance metrics. As a baseline, we include the example strategy proposed in [1]: λ is initialized with zero and increased in small steps - as soon as a new return is worse than the previous one, the search stops and the previous λ is chosen. We refer to this strategy as Increase-Conservative (Inc-Con). Since this strategy can be considered quite restrictive, we consider a relaxation Increase-Behavioral (Inc-Beh), which stops only once the performance drops below that of the behavioral policy, and then uses the best so far observed λ value. These two strategies obviously aim to reduce regret compared to the behavioral performance, which is often a key metric, however we also evaluate a Greedy strategy, which starts at the opposite end and moves towards lower λ values in the hope of finding the optimum quicker if the policy can generalize well. Then we compare different gradient-free optimizing strategies that have previously been proposed to see if they can outperform the simple strategies in speed, performance, as well as regret measures: PSO (Particle Swarm Optimization [7]) is a swarm based optimization method which uses neighbourhood

ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

information for search. Scr (ScrHammersleySearchPlusMiddlePoint [8]) is a one-shot optimizer which hopefully reduces the number of evaluations necessary. DE (Differential Evolution [9]) iteratively improves the candidate solution(s), similar to PSO, and NGOpt [8] is a meta optimizer that, like Scr, should produce decent solutions even with low evaluation budgets.

4 Experiments

We use the industrial benchmark (IB) datasets and MuJoCo Swimmer, Hopper, and Walker datasets as proposed in [3]. The IB is a benchmark motivated by commonly encountered control problems in industrial real-world scenarios, exhibiting transitions with heteroscedastic noise, delayed reward components, partially observable states, and high dimensional state and action spaces. The respective datasets have been gathered by combining three deterministic baseline policies (bad, mediocre, optimized) with different levels of ε-greedy exploration (0%, 20%, 40%, 60%, 80%, 100%). The considered MuJoCo tasks are also well known RL benchmarks with highly complex, yet deterministic and fully observable state transitions. The respective datasets have been gathered by adding different noise sources to an expert policy. Combined, we evaluate AutoLION on 19 datasets from different tasks and domains and a wide variety of settings, which should give a general overview on how the method performs.

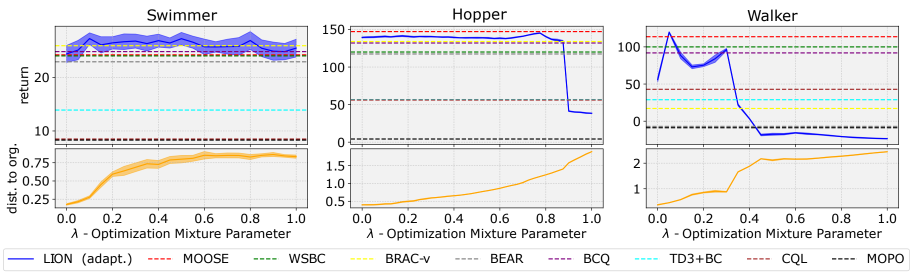

Fig. 1: Visualization of the performance landscape of the LION algorithm over the λ spectrum (blue), together with the corresponding proximity to the behavioral policy (orange). We show the same set of offline RL baselines as in [1] - they are presented as horizontal dashed lines since they cannot adapt the trade-off and since we cannot assign their regularization parameter to a corresponding λ value.

We train LION policies as proposed in [1], and then search the space of trade-off parameters λ with the proposed search methods from the previous section in order to provide an automated adaptive offline RL algorithm that works in continuous action spaces. Return and distance to behavior policy curves over the λ spectrum for the MuJoCo datasets are provided in Fig. 1. The evaluations in

ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

terms of R, MOR, MBR, as well as RUB are provided in Fig. 2: We can see that, for obvious reasons, the Inc-* strategies perform best in terms of regret compared with the behavioral policy, since they use underperformance with respect to the behavioral policy as their stopping criterion (the conservative strategy is even stricter, i.e. it uses an upper bound on the behavioral performance since the return is never allowed to decrease). The conservative strategy has to pay for this with relatively low final returns and is outperformed by any other strategy in our comparison. Our proposed Inc-Beh strategy can be seen as an almost dominant solution compared with the conservative one, since it has close to zero MBR like Inc-Con, but mostly better MOR, and very good R, almost reaching the top performances of Greedy, PSO, Scr, and DE. The Greedy strategy appears competitive with the other gradient-free optimization algorithms in terms of return (especially on the bad datasets), however it is dominated by Scr and DE in every metric, especially in the two regret measurements. match or outperform it while also exhibiting lower MBR. PSO & NGOpt appear less favourable - they underperform Scr & DE in terms of regret while achieving similar returns.

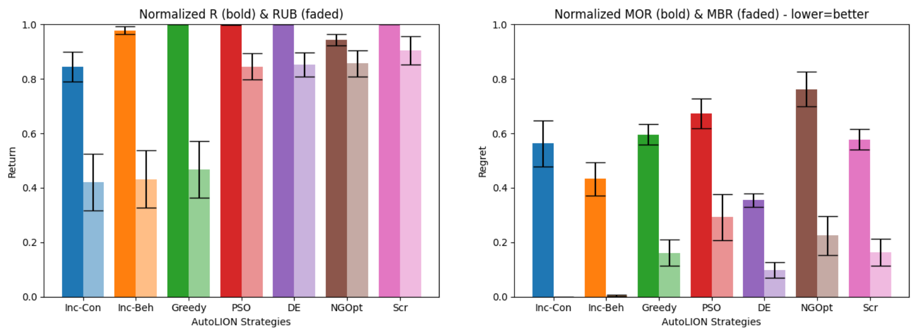

Fig. 2: (a) Mean and standard error of normalized unconstrained returns versus returns under budget for each AutoLION strategy, aggregated across all 19 datasets. While Inc-Beh and Greedy are competitive in the unconstrained budget setting, their performance drops significantly when the evaluation budget is limited. The gradient-free search algorithms all appear much more robust in this regard, especially Scr does not loose much performance under the limited budget. (b) Mean and standard error of Normalized Mean Regret w.r.t. the optimal λ choice (MOR) and Mean Regret w.r.t. the behavioral choice (i.e. $\lambda = 0$). Inc-Con and Inc-Beh are unmatched in their MBR, while Differential Evolution achieves the best MOR (and the best MBR among the gradient-free search strategies). We normalize returns and regrets to the (0,1) interval by subtracting dataset-wise minimum and dividing by (maximum - minimum).

ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 4-6 October 2023, i6doc.com publ., ISBN 978-2-87587-088-9. Available from http://www.i6doc.com/en/.

5 Discussion & Conclusion

In this paper, we propose to augment conditional, adaptive offline RL policies by a trade-off search phase, which can automatically adjust the corresponding hyperparameter in order to optimize returns once the policy is deployed without any retraining. We introduce relevant performance metrics by which strategies can be evaluated and test the approach with seven different strategies, by augmenting the LION algorithm with our approach, yielding an extension to automatically adapt the trade-off for offline RL at runtime, providing the user an autopilot like option. We thus term our developed approach AutoLION. By testing the algorithm on nineteen datasets from the industrial benchmark and MuJoCo domains, we find that depending on the concrete practical setting, two options dominate: If MBR is critical, i.e. the policy should rarely underperform the behavioral, our proposed relaxation Inc-Beh is likely the best option, while if quick trade-off search under an evaluation budget is needed, Scr performs best. At the intersection between the two, Differential Evolution can represent a meaningful compromise, unifying relatively high returns in both constrained and unconstrained budget settings with relatively low regrets compared to the other gradient-free optimization algorithms. Future work might come up with combinations of the strategies, by e.g. employing MBR-penalized optimizations with few-shot optimizers like Scr or NGOpt.

References

[1] Swazinna, Phillip, Steffen Udluft, and Thomas Runkler. "User-Interactive Offline Reinforcement Learning." International Conference on Learning Representations (ICLR), 2023.

[2] Rigter, Marc, Bruno Lacerda, and Nick Hawes. "Rambo-rl: Robust adversarial model-based offline reinforcement learning." Neural Information Processing Systems (NeurIPS), 2022.

[3] Swazinna, Phillip, Steffen Udluft, and Thomas Runkler. "Overcoming Model Bias for Robust Offline Deep Reinforcement Learning." Engineering Applications of Artificial Intelligence (EAAI), 2021.

[4] Ghosh, Dibya, Anurag Ajay, Pulkit Agrawal, and Sergey Levine. "Offline RL policies should be trained to be adaptive." International Conference on Machine Learning (ICML), 2022.

[5] Hong, Joey, Aviral Kumar, and Sergey Levine. "Confidence-Conditioned Value Functions for Offline Reinforcement Learning." International Conference on Learning Representations (ICLR), 2023.

[6] Hein, Daniel, Stefan Depeweg, Michel Tokic, Steffen Udluft, Alexander Hentschel, Thomas A. Runkler, and Volkmar Sterzing. "A benchmark environment motivated by industrial control problems." 2017 IEEE Symposium Series on Computational Intelligence (SSCI), 2017.

[7] Eberhart, Russell, and James Kennedy. "Particle swarm optimization." Proceedings of the IEEE international conference on neural networks. Vol. 4. Citeseer, 1995.

[8] J. Rapin and O. Teytaud. "Nevergrad - A gradient-free optimization platform." `https://GitHub.com/FacebookResearch/Nevergrad`, 2018

[9] Storn, Rainer. "On the usage of differential evolution for function optimization." Proceedings of North American fuzzy information processing. IEEE, 1996.