Modified Actor-Critics

Extended Abstract

Erinc Merdivan
AIT Austrian Institute of Technology
& CentraleSupélec

Sten Hanke
FH Joanneum Gesellschaft mbH

Matthieu Geist
Google Research, Brain Team

ABSTRACT

Recent successful deep reinforcement learning algorithms, such as Trust Region Policy Optimization or Proximal Policy Optimization, are fundamentally variations of conservative policy iteration. These algorithms iterate policy evaluation followed by a softened policy improvement step. As so, they are naturally on-policy. Here, we propose to combine (any kind of) soft greediness with Modified Policy Iteration. The proposed abstract framework applies repeatedly: (i) a partial policy evaluation step that allows off-policy learning and (ii) any softened greedy step. Our contribution can be seen as a new generic tool for the deep reinforcement learning toolbox.

ACM Reference Format:
Erinc Merdivan, Sten Hanke, and Matthieu Geist. 2020. Modified Actor-Critics. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020,* IFAAMAS, 3 pages.

1 INTRODUCTION

Many deep reinforcement learning (RL) algorithms are based on approximate dynamic programming. For example, the celebrated DQN [10] is based on approximate value iteration. As a pure critic approach, it can only deal with finite action spaces. A more versatile approach, that allows handling both discrete and continuous action spaces, consists in considering actor-critic architectures, where both the value function and the policy are represented. Most of such recent approaches are either variations of policy gradient [7, 9, 18], inspired by conservative policy iteration [15, 16, 19], or make use of entropy regularization [3–5].

If approximate policy iteration has already been the building block of actor-critics in the past [2], it has not been considered with deep learning approximators, as far as we know. We assume that this is due to the fact that the greedy operator is unstable (much like gradient descent with too big step sizes). A clever way to address this issue has been introduced by Kakade and Langford [6] with Conservative Policy Iteration (CPI). Instead of taking the greedy policy, the new policy is a stochastic mixture of the current one and of the greedy one. This softens greediness and stabilizes learning.

With this classical approach, the current policy is a stochastic mixture of all past policies, which is not very practical. The core idea of CPI has been adapted in the deep RL literature by modifying how the greediness is softened. For example, Trust Region Policy Optimization (TRPO) [15] or Actor-Critic using Kronecker-factored Trust Region (ACKTR) [19] add a constraint on the greedy step, imposing that the average Kullback-Leibler (KL) divergence between consecutive policies is below a given threshold, and Proximal Policy Optimization (PPO) [16] modifies the greedy step with a clipping loss that forces the ratio of action probabilities of consecutive policies to remain close to 1. To some extent, even policy gradient approaches can be seen as such, as following the policy gradient should provide a softened improvement (see also [13] for a connection between CPI and policy gradient). Other approaches consider an entropy penalty [3–5], which effect is also to soften greediness (but can also modify the evaluation step).

We will call generally "Softened Policy Iteration" (SPI) any approach that combines policy evaluation with a softened greedy step. As they require policy evaluation, these approaches are naturally on-policy. In classical dynamic programming, Modified Policy Iteration (MPI) [12] replaces the full evaluation of the policy by a partial evaluation. This idea has been extended to the approximate setting (Approximate MPI [14]), but never with deep learning approximators, as far as we know. This is probably due to the instability of the greedy step. Yet, a partial evaluation presents some interest, compared to a full policy evaluation. It allows for an easier extension to off-policy learning by making use of Temporal Difference (TD) learning instead of using rollouts. It also draws a bridge between value and policy iterations (because MPI has these two algorithms as special cases). In this work, we propose an abstract actor-critic framework that brings together MPI and SPI, by mixing the partial evaluation of MPI with the softened greediness of SPI. We name the resulting approach Modified Soft Policy Iteration (MoSoPI).

2 BACKGROUND

A Markov Decision Process (MDP) is a tuple $\{S, A, P, r, \gamma\}$, with S the state space, A the action space, P the transition kernel ($P(s'|s,a)$ denotes the probability to go from s to s' under action a), $r \in \mathbb{R}^{S\times A}$ the reward function and $\gamma \in (0,1)$ the discount factor. A (stochastic) policy π is a mapping from states to distribution of actions ($\pi(a|s)$ denotes the probability of choosing a in s). The quality of a policy is quantified by the value function, $v_\pi(s) = \mathbb{E}_\pi[\sum_{t\geq 0}\gamma^t r(s_t,a_t)|s_0 = s]$,where $\mathbb{E}_\pi$ denotes the expectation respectively to the trajectories sampled by the policy π and the dynamics P. Write T_π the Bellman operator, defined for any $v \in \mathbb{R}^S$ as $\forall s \in S$, $[T_\pi v](s) = \mathbb{E}_{a\sim\pi(.|s)}[r(s,a)+\gamma v(s')]$. The value function v_π is the unique fixed point of the operator T_π. The aim of RL is to maximize either the value function for each state or an average value function. To do so, the notion of Bellman optimality operator is useful:$\forall v \in \mathbb{R}^S$, $Tv = \max_\pi T_\pi v$. The optimal value function v_* is the unique fixed point of T. The notion of greedy operator can be derived from T. We say that π is greedy respectively to $v \in \mathbb{R}^S$ (that is not necessarily a value function) if $\pi \in \mathcal{G}(v) \Leftrightarrow Tv = T_\pi v$. The value function might not be convenient from a practical viewpoint, as applying the operators T and $\mathcal{G}$ requires knowing the dynamics. To

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand.

alleviate this issue, a classical approach is to consider a Q-function, that adds a degree of freedom on the first action to be chosen, $Q_\pi(s,a) = \mathbb{E}_\pi[\sum_{t\geq 0}\gamma^t r(s_t,a_t)|s_0 = s, a_0 = a]$. Similarly to the value function, we can define the associated T_π, T and $\mathcal{G}$ operators. Value and Q-functions are linked by $v_\pi(s) = \mathbb{E}_{a\sim\pi(.|s)}[Q_\pi(s,a)]$, and the advantage function is defined as the state-wise centered Q-function, $A_\pi(s,a) = Q_\pi(s,a) - v_\pi(s)$.

3 MODIFIED SOFTENED POLICY ITERATION

Policy iteration (PI) alternates improvement and evaluation:

$$\begin{cases}\pi_{k+1} = \mathcal{G}(v_k)\\ v_{k+1} = v_{\pi_{k+1}}\end{cases}. \tag{1}$$

In the exact case, everything can be computed analytically (given finite and small enough state and action spaces), and this PI scheme will converge in finite time. In an approximate setting, one has to approximate both the value function and the policy (possibly implicitly), and to learn them from samples.

We start by discussing the approximation of policy evaluation. First, as explained before, it is more convenient to work with Q-functions. Let Q_θ be a parameterized Q-function, Q_π can be estimated using rollouts. Write generally $\hat{\mathbb{E}}$ for an empirical estimation, assume that a set of state-action couples $(s_i,a_i)_{1\leq i\leq n}$ is available, and that we can simulate the return R_i (the cumulative discounted reward from a rollout starting in (s_i,a_i) and following the policy afterwards), then the Q-function can be estimated by minimizing $J(\theta) = \hat{\mathbb{E}}[(R_i - Q_\theta(s_i,a_i))^2]$. There exist approaches for estimating Q-functions directly from transitions, such as LSTD [1], but they usually assume a linear parameterization.

If the action space is finite, the greedy policy can be deduced from the estimated Q-function $\hat{Q}_k$: $\pi_{k+1}(a|s) = 1_{\{a=\text{argmax}_b\,\hat{Q}_k(s,b)\}}$. Generally, one can also adopt a parameterized policy π_w and solve the greedy step as maximizing the following optimization problem: $J(w) = \hat{\mathbb{E}}\left[\mathbb{E}_{a\sim\pi_w(.|s_i)}[\hat{Q}_k(s_i,a)]\right]$. Notice that this would correspond to solving $\mathbb{E}_{s\sim\mu}[[T_{\pi_w}v](s)]$ for some distribution μ instead of the greedy step in (1). Adding a state-dependant baseline to $\hat{Q}_k$ does not change the minimizer, and one consider usually an estimated advantage function $\hat{A}_k$ to reduce the variance of the gradient. With discrete actions, this corresponds to a cost-sensitive multi-class classification problem [2].

Softened Policy Iteration. The greedy step can be unstable in an approximate setting. To alleviate this problem, Kakade and Langford [6] proposed to soften it by mixing the greedy policy with the current one. Let $\alpha_k \in (0,1)$, the greedy step $\pi_{k+1}\in\mathcal{G}(v_k)$ is replaced by $\pi_{k+1} = (1-\alpha_k)\pi_k + \alpha_k\mathcal{G}(v_k)$. This comes with a monotonic improvement guarantee, given a small enough α_k. However, it is not very practical, as the new policy is a mixture of all previous policies. To alleviate this problem, Schulman et al. [15] proposed to soften the greediness with a KL penalty between consecutive policies, that leads to minimize: $\hat{\mathbb{E}}[\mathbb{E}_{a\sim\pi_w(.|s_i)}[\hat{Q}_k(s_i,a)]]$ s.t. $\hat{\mathbb{E}}[\text{KL}(\pi_w(.|s_i)||\pi_k(.|s_i))] \leq \epsilon$. Other approaches are possible. For example, PPO combines the previous approximate greedy step with importance sampling and a clipping of the ratio of probabilities: $J(w) = \hat{\mathbb{E}}[\mathbb{E}_{a\sim\pi_k(.|s_i)}[\text{clip}_\epsilon(\frac{\pi_w(a|s_i)}{\pi_k(a|s_i)}\hat{A}_k(s_i,a))]]$. The clip_ϵ operator saturates the ratio of probabilities when it deviates too from 1 (at $1+\epsilon$ if the advantage is positive, $1-\epsilon$ else), without it it would be equivalent to the initial greedy step. In this work, we call SPI any policy iteration combined with a softened greedy step, that we frame as satisfying $T_{\pi_{k+1}}v_k \geq T_{\pi_k}v_k$ (so, we ask the policy to provide some improvement, without being necessarily the greedy one). In that sense, even a policy gradient step can be seen as softened greediness.

Modified Policy Iteration. If SPI modifies the greedy step, MPI [12] modifies the evaluation step. The operator T_{π_k} being a contraction, we can write $v_{\pi_k} = (T_{\pi_k})^\infty v$, for any $v\in\mathbb{R}^S$, so notably for $v = v_{k-1}$. MPI does partial evaluation by iterating the operator a finite number of times. Let $m\geq 1$, MPI iterates

$$\begin{cases}\pi_{k+1} = \mathcal{G}(v_k)\\ v_{k+1} = (T_{\pi_{k+1}})^m v_k\end{cases}. \tag{2}$$

For $m=\infty$, we retrieve PI, and for $m=1$ we retrieve value iteration (VI): as $T_{\mathcal{G}(v)}v = Tv$, with $m=1$ it reduces to $v_{k+1} = Tv_k$.

We have that $(T_\pi)^m v = \mathbb{E}_\pi[\sum_{t=0}^{m-1}\gamma^t r(s_t,a_t) + \gamma^m v(s_m)|s_0 = s]$. This suggests two ways of estimating a value function (or next, directly a Q-function). First, consider the case $m=1$ and a parameterized Q-function. The classical approach consists in solving the following regression problem: $J(\theta) = \hat{\mathbb{E}}\left[(y_i - Q_\theta(s_i,a_i))^2\right]$ with $y_i = r_i + \gamma\mathbb{E}_{a'\sim\pi_{k+1}(.|s')}[\hat{Q}_k(s',a')]$. With $m>1$, a solution is to perform an m-step rollout (using π_{k+1}) and to replace y_i by $y_i^m = \sum_{t=0}^{m-1}\gamma^t r_{i+t} + \gamma^m\mathbb{E}_{a'\sim\pi_{k+1}(.|s_{t+m})}[\hat{Q}_k(s_{t+m},a')]$. This can be corrected for off-policy learning, using for example importance sampling or Retrace [11].

Another approach is to solve m times the previous regression problem, replacing $\hat{Q}_k$ by the newly computed Q_θ after each regression but keeping the policy π_{k+1} fixed over the m regressions. In other words, solving for $J(\theta)$ is one application of an approximate Bellman evaluation operator, and this amounts to applying it m times. Although using m-step returns is pretty standard in deep RL (even if its relation to the classical MPI is rarely acknowledged), the second approach is less usual and has never been experimented in a deep RL context, to the best of our knowledge.

Modified Soft Policy Iteration. MoSoPI simply consists in bringing together a softened policy step of SPI (so, any kind of softened greediness) and the partial evaluation step of MPI:

$$\begin{cases}\text{find } \pi_{k+1} \text{ s.t. } T_{\pi_{k+1}}v_k \geq T_{\pi_k}v_k\\ v_{k+1} = (T_{\pi_{k+1}})^m v_k\end{cases}. \tag{3}$$

To get a practical algorithm, one just has to choose a soft greedy step and to estimate the partial evaluation of the Q-function.

4 TO GO FURTHER

The full version of this article [8] provides additional things. To justify the proposed approach (mixing softened greedy step with partial evaluation), we also propose and discuss a convergence analysis of MoSoPI in an ideal case (no approximation error). As a proof of concept of this general simple idea, we also instantiate in details this framework with the PPO greediness. Empirical comparisons to the original PPO on various Mujoco tasks [17] shows that the resulting Modified PPO (MoPPO) is much more sample efficient, despite a loss of stability. We also show that it is competitive with the state-of-art off-policy algorithm Soft Actor Critic (SAC) [4].

REFERENCES

[1] Steven J Bradtke and Andrew G Barto. 1996. Linear least-squares algorithms for temporal difference learning. *Machine learning* (1996).

[2] Victor Gabillon, Alessandro Lazaric, Mohammad Ghavamzadeh, and Bruno Scherrer. 2011. Classification-based Policy Iteration with a Critic. In *International Conference on Machine Learning (ICML)*.

[3] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. 2017. Reinforcement Learning with Deep Energy-Based Policies. In *International Conference on Machine Learning (ICML)*.

[4] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *International Conference on Machine Learning (ICML)* (2018).

[5] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. 2018. Soft Actor-Critic Algorithms and Applications. *arXiv preprint arXiv:1812.05905* (2018).

[6] Sham Kakade and John Langford. 2002. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning (ICML)*.

[7] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. 2016. Continuous control with deep reinforcement learning. *International Conference on Learning Representations (ICLR)* (2016).

[8] Erinc Merdivan, Sten Hanke, and Matthieu Geist. 2019. Modified Actor-Critics. *arXiv preprint arXiv:1907.01298* (2019).

[9] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning (ICML)*.

[10] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* (2015).

[11] Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. 2016. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[12] Martin L Puterman and Moon Chirl Shin. 1978. Modified policy iteration algorithms for discounted Markov decision problems. *Management Science* (1978).

[13] Bruno Scherrer and Matthieu Geist. 2014. Local policy search in a convex space and conservative policy iteration as boosted policy search. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases(ECML-KDD)*. Springer.

[14] Bruno Scherrer, Mohammad Ghavamzadeh, Victor Gabillon, Boris Lesner, and Matthieu Geist. 2015. Approximate modified policy iteration and its application to the game of Tetris. *Journal of Machine Learning Research (JMLR)* (2015).

[15] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International Conference on Machine Learning (ICML)*.

[16] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[17] Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. Mujoco: A physics engine for model-based control. In *International Conference on Intelligent Robots and Systems (IROS)*. IEEE.

[18] Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, and Nando de Freitas. 2017. Sample efficient actor-critic with experience replay. *International Conference on Learning Representations (ICLR)* (2017).

[19] Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba. 2017. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. In *Advances in neural information processing systems (NeurIPS)*.