Curious Learning with Multiple Soft Actor-Critics

Final Submission

Maximilian Keiff and Alina Munir

27.01.2022

In reinforcement learning a popular model is the actor-critic, which is based on policy gradient and is used to learn which action to take and evaluates it for a given state. Tackling the issue of hyperparameter sensitivity and sample inefficiency at the same time, Haarnoja et al. (2018) introduced the state-of-the-art algorithm soft actor-critic (SAC) which extends the actor-critic by an off-policy model and entropy maximization. The latter motivates the agent for more exploration but leaves space for improvement. In our research paper, we are extending the approach of SAC introduced by Röder et al. (2020) where it is evaluated that for continuous domain environment the incorporation of curiosity increases more than twice the learning rate and success rate of the problem. In this paper, we approach the curiosity factor in a different way that has been proven to outperform on simple actor-critic models (Martinez-Piazuelo et al., 2020; Nguyen et al., 2021). We investigate this factor of curiosity by initializing different critics on a simple off-policy continuous domain entropy maximizing Soft Actor-Critic (SAC) model (Jesus et al., 2021).

1 Introduction

Reinforcement learning (RL) is a subarea of machine learning that deals with an agent learning an objective in an environment by trial and error and rewards alone. In reality, this concept can be applied both to pure software agents that teach themselves for example to play a video game, and to robots that have to solve a physical task, such as reaching a given goal position as quickly as possible.

If the agent does not explicitly learn or use a transition model from the environment during the learning process, we speak of model-free learning. Q-learning belongs to such model-free RL algorithms, which learn to evaluate the value of an action in a given state. "Q" refers to the function that computes the expected rewards for an action state pair. These rewards are used to update the agent's policy. If the policy is trained with samples from itself, the algorithm is categorized as on-policy. Alternatively we speak of off-policy algorithms where samples can

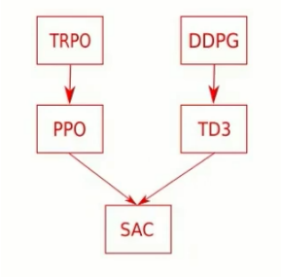

Figure 1: Taxonomy of actor critic algorithms.

be produced by another policy, this can also simply mean that old samples from a previous version of the target policy can be stored and reused from a buffer.

A special class of Q-learning algorithms use two neural networks to model the agent, or in this context we speak of an actor, and its critic, which approximates the value function by adjusting the Q-values. These so called actor-critic algorithms are divided into two broad approaches, according to whether the policy is stochastic or deterministic in nature. The first group includes algorithms such as TRPO and PPO, which are very stable but learn very slowly because they cannot reuse samples from previous iterations. The second group includes algorithms such as DDPG and TD3, which have better sample efficiency because they also store results from previous iterations in a replay buffer, but are dependent on intensive hyperparameter tuning and are therefore very unstable. SAC is marrying the advantages from both worlds of deep reinforcement learning approaches, namely the stochastic policies that learn on-policy and the deterministic policies that learn off-policy. SAC also belongs to the stochastic policies, but also uses a replay buffer and thus learns off-policy with samples from its previous iterations. In addition, SAC is particularly stable, since it also includes entropy regularization. Due to this entropy regularization, it is also called a soft actor critic model, because in comparison to TD3 determinism and thus convergence to a local minimum is avoided.

Research by Fujimoto et al. (2018) has shown that SAC with only one critic tends to overestimate some action values in noisy environments. To counter this bias, it has been proposed to use multiple critics with different initialization and then take the minimum of their value approximations to train the Q-values and policy of the actor.

This has also opened up new but so far unused opportunities, to develop further metrics on top of multiple critics. With our research, we want to introduce one approach for integrating such a metric, specifically the variance of the critics Q-values, into the SAC algorithm and investigate to what extent it promotes additional exploration.

2 Background

We will first explain the SAC algorithm in more detail, and in the following Section 3 we will present our own modifications. As just mentioned, SAC adds entropy

regularization to its stochastic policy to provide an incentive for curious learning. The entropy of a state is given as a metric for the balance of probabilities for the possible actions, thus indicating the randomness of the state and motivating more curious exploration. The entropy $\mathcal{H}(\pi(\cdot|s_t)) = \mathbb{E}_{a_t \sim \pi_\theta(\cdot|s_t)} [-\log \pi_\theta(a_t|s_t)]$ is added together with a so called temperature hyper-parameter α which regulates the impact on the learning objective of the actor.

$$\pi^* = \arg\max_{\pi_\theta} \sum_t \mathbb{E}_{(s_t,a_t)\sim\pi_\theta} [r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot|s_t))] \tag{1}$$

The equation basically expresses that SAC is maximizing the policy π^* over all timesteps t for the expected reward $r(s_t, a_t)$ and entropy of the state s_t.

In the same way, SAC introduces a new soft value function $V_\phi^{\pi_\theta}(s_t)$ that adds the entropy to the usual function of expectations provided by the Q-values of the actions in a state s_t.

$$V_\phi^{\pi_\theta}(s_t) = \mathbb{E}_{a_t \sim \pi_\theta(\cdot|s_t)} \left[Q_\phi^{\pi_\theta}(s_t, a_t) \right] + \alpha \mathcal{H}(\pi(\cdot|s_t)) \tag{2}$$

$$= \mathbb{E}_{a_t \sim \pi_\theta(\cdot|s_t)} \left[Q_\phi^{\pi_\theta}(s_t, a_t) - \alpha \log \pi_\theta(a_t|s_t) \right] \tag{3}$$

With this value function, the standard Bellman operator $\mathcal{T}^\pi Q_\phi^{\pi_\theta}(s_t, a_t)$ for the temporal difference target can be defined to update the Q-values. Here, the reward $r(s_t, a_t)$ of the selected action a_t is combined with the value of the successor state s_{t+1} weighted by the discount factor γ.

$$\mathcal{T}^\pi Q_\phi^{\pi_\theta}(s_t, a_t) = r(s_t, a_t) + \gamma V_\phi^{\pi_\theta}(s_{t+1}) \tag{4}$$

To train the parameters of the critic the function $J_Q(\theta)$ which we want to minimize by using minibatches of samples of state transactions of the form (s_t, a_t, s_{t+1}) from the replay buffer $\mathcal{D}$. This training procedure is exactly the same as for DDPG except for the entropy extension in the value function, but still backpropagation can be used to adjust the output Q-values.

$$J_Q(\theta) = \mathbb{E}_{(s_t,a_t,s_{t+1})\sim\mathcal{D}} \left[\left(\mathcal{T}^\pi Q_\phi^{\pi_\theta}(s_t, a_t) - Q_\phi^{\pi_\theta}(s_t, a_t) \right)^2 \right] \tag{5}$$

To train the policy of the actor, DDPG cannot be used as a guide because SAC has a stochastic policy. Instead, the actor is trained to become greedy towards the Q-values by minimizing the following function which estimates an expectation over actions sampled from the actor:

$$J_\pi(\theta) = \mathbb{E}_{s_t \sim \mathcal{D}} \left[\mathbb{E}_{a_t \sim \pi_\theta(\cdot|s_t)} \left[\alpha \log \pi_\theta(a_t|s_t) - Q_\phi^{\pi_\theta}(s_t, a_t) \right] \right] \tag{6}$$

Both the actor and the critic can be modeled by a neural network. Figure 2 shows how for continuous action spaces the critic takes a state and an action vector and outputs a Q-value scalar. The actor takes a state vector and outputs a normal distribution defined by the parameters μ_θ and σ_θ. To backpropagate with these two parameters for the actor's neural network, the re-parameterization trick

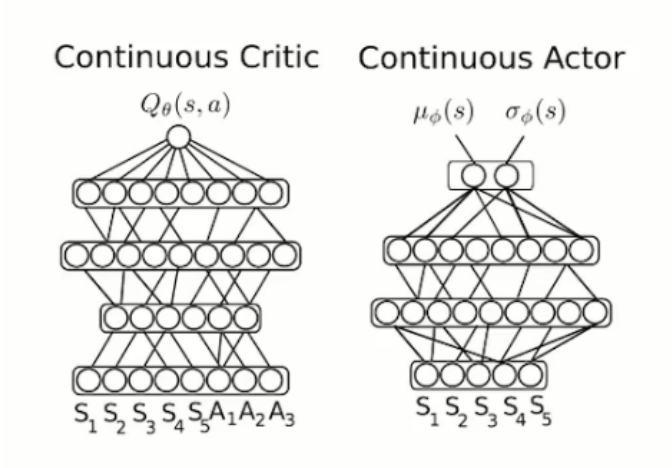

Figure 2: Networks of SAC for continuous action spaces.

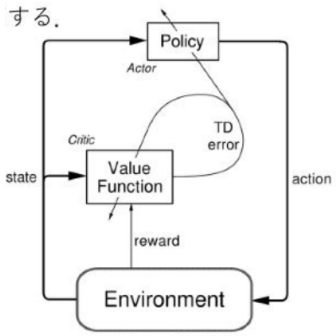

Figure 3: SAC model.

needs to be applied, where the chosen action $a_t = \tanh(\mu_\theta + \varepsilon\sigma_\theta)$ is bounded by a hyperbolic tangent, where $\varepsilon \sim \mathcal{N}(0,1)$ (**?**).

Already in TD3 it has been shown that a single critic tends to overestimate specific actions in given states (Fujimoto et al., 2018). To counteract this bias, additional critics Q_{ϕ_i} with $i = 1, 2, \ldots$ each with their own network are added, and the respective minimum $\min_i Q_{\phi_i}(s_{t+1}, a)$ of their evaluations is adopted to compute the temporal difference target Q-values.

Another aspect the authors draw attention to is that the choice of the temperature parameter α is not trivial and can be instead be learned, provided at least a lower bound $\mathcal{H}_0$ is given, which depends on the size of the action space. The complete algorithm for SAC is visualized in Figure 3 and can be simplified into following steps:

1. Initialize weights of actor π_θ and critics $Q_{\phi_i}^{\pi_\theta}$ neural networks
2. Run k steps in the environment by sampling actions with π_θ
3. Store the collected transitions and rewards $(s_t, a_t, r(s_t, a_t), s_{t+1})$ in a replay buffer $\mathcal{D}$

4. Sample k batches of transitions from the replay buffer $\mathcal{D}$
5. Update the temperature α, the actor π_θ and the critics $Q_{\phi_i}^{\pi_\theta}$ by SGD
6. Repeat this cycle until convergence

3 Methodology

Building on the idea that multiple critics evaluating the selected actions per state in parallel leads to a more stable behavior of the SAC, we will try to use them additionally to encourage a more curious learning of the actor. The previous incentive to keep the stochastic policy from converging too early but rather explore further consists of the additional weighting on the entropy of the probabilities of the actions in a specific state. In this context, it makes sense to try out additional metrics that result from the so far unused possibilities opened up by introducing multiple critics. With our research, we specifically want to investigate the variance of the state ratings of the critics as a possible incentive for curious learning. First we reconsider the temporal difference target $y_t := \mathcal{T}^\pi Q_{\phi_i}^{\pi_\theta}(s_t, a_t)$ for multiple critics:

$$y_t = r(s_t, a_t) + \gamma V_{\phi_i}^{\pi_\theta}(s_{t+1}) \tag{7}$$

$$= r(s_t, a_t) + \gamma \mathbb{E}_{a_{t+1} \sim \pi_\theta(\cdot|s_{t+1})} \left[\min_i Q_{\phi_i}^{\pi_\theta}(s_{t+1}, a_{t+1}) - \alpha \log \pi_\theta(a_{t+1}|s_{t+1}) \right] \tag{8}$$

The normalized variance S from the Q-values of the critics for the transitions can be calculated as follows:

$$S = \frac{Var_i \left(Q_{\phi_i}^{\pi_\theta}(s_{t+1}, a_{t+1}) \right) - Var_{min}}{Var_{max} - Var_{min}} \tag{9}$$

Here V_{min} and V_{max} are estimates for the minimum and maximum variance. This variance S weighted with an additional coefficient $\lambda \in [0, 1]$ is added to the standard Bellman operator. The hyper-parameter coefficient λ is used here only to control how much we want to balance the variance and reward.

$$y_t = (1 - \lambda) \cdot r(s_t, a_t) + \gamma V_{\phi_i}^{\pi_\theta}(s_{t+1}) + \lambda S \tag{10}$$

The specific choice of this presented formula is due to the limited time of this seminar and we will discuss alternative approaches in Section 5.

We implement this modified SAC algorithm with variance driven curiosity in the IDEAS / LeCAREbot deep reinforcement learning framework (Röder et al., 2020) provided by the Knowledge Technology Research Group of the University of Hamburg. Our algorithm is based on their already implemented SAC algorithm and only overwrites the training method and adds the variance coefficient λ to the injected hyper-parameters. The test environments we use for our performance analysis are the FetchReach, BlockStack and AntReach (see Figure 4) with the gripper above benchmarks from OpenAI gym (Brockman et al., 2016), which we simulate in the physics engine Mujuco (Todorov et al., 2012).

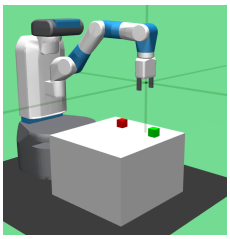
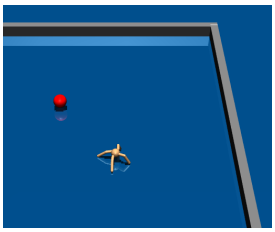

Figure 4: The FetchReach or BlockStack and the AntReach environment.

For four critics we will compare the established SAC algorithm with our variance driven SAC and adjust the influence of variance and rewards by the coefficent λ to different degrees. The results of the experiments were collected and visualized using the tracking tool MLflow [1].

4 Results

5 Discussion

We might want to investigate the following equation:

$$y_t = r(s_t, a_t) + \gamma \mathbb{E}_{a_{t+1} \sim \pi_\theta(\cdot|s_{t+1})} \left[\min_i Q_{\phi_i}^{\pi_\theta}(s_{t+1}, a_{t+1}) \right] + (1-\lambda)\gamma\alpha\mathcal{H}(\pi(\cdot|s_{t+1})) + \lambda S \tag{11}$$

As random initialization of neural network and action selection of the agent adds the stochasticity, in future work we plan to run the code several times and get the mean graphs as our final result.

6 Conclusion

References

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

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.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018.

[1] `https://mlflow.org/` (Accessed on 1/23/2022)

De Jesus, J., Kich, V.A., Kolling, and A.H. Soft actor-critic for navigation of mobile robots, 2021. URL https://doi.org/10.1007/s10846-021-01367-5. J Intell Robot Syst 102.

Juan Martinez-Piazuelo, Daniel E. Ochoa, Nicanor Quijano, and Luis Felipe Giraldo. A multi-critic reinforcement learning method: An application to multi-tank water systems. *IEEE Access*, 8:173227–173238, 2020. doi: 10.1109/ACCESS.2020.3025194.

Ngoc Duy Nguyen, Thanh Thi Nguyen, Peter Vamplew, Richard Dazeley, and Saeid Nahavandi. A prioritized objective actor-critic method for deep reinforcement learning. *Neural Computing and Applications*, 33(16):10335–10349, Aug 2021. ISSN 1433-3058. doi: 10.1007/s00521-021-05795-0. URL https://doi.org/10.1007/s00521-021-05795-0.

Frank Röder, Manfred Eppe, Phuong D. H. Nguyen, and Stefan Wermter. Curious hierarchical actor-critic reinforcement learning, 2020.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.