Munchausen Reinforcement Learning with Continuous Action Space - Proposal

Marcel Brucker
Technical University of Munich
Munich, Germany
marcel.brucker@tum.de

Finn Süberkrüb
Technical University of Munich
Munich, Germany
finn.sueberkrueb@tum.de

I. Objective

Many Atari games and simple test scenarios for reinforcement learning (RL) algorithms use discrete action spaces. But for real-world applications, such as a simple inverse pendulum or more advanced object grasping, a continuous action space is needed.

One of the best known model-free, non-continuous RL algorithms are Deep Q-Networks (DQN). A slight adaptation of the value update has resulted in the Munchausen DQN (M-DQN) [1] which has improved performance compared to the standard DQN.

The method is deemed to be easily applicable to other RL algorithms which is why the goal of the this work is to augment Soft Actor-Critic (SAC) by the Munchhausen idea (M-SAC) for the use on continuous action spaces. Our algorithm will be tested on common problems like the quadruped agent of OpenAI Gym [4] to compare the performance of M-SAC with alternative algorithms.

II. Related Work

A. M-DQN

The core idea of Munchhaus RL [1] is very simple. The scaled log policy is added to the immediate reward. In other words, we reduce the reward given for actions where the policy is quite uncertain about the best action providing us an additional stimulus for training. This suggests that we pull ourselves out of the swamp by our own hair, like Baron Munchausen, where the name comes from.

B. SAC

SAC [3] is one of the most popular model-free reinforcement learning algorithms. Its success results from sample-efficient learning, due to its off-policy nature, and simultaneously demonstrating stability and convergence at the level of on-policy methods or even above. There is an actor that determines the next action given a state by its policy and a critic that evaluates the chosen action using a Q-function. To make the algorithm more robust, not only the direct reward is maximized but also the entropy of the policy. The influence of maximizing the entropy with the reward ensures a good trade-off between exploration and exploitation. This is where the added term "soft" comes from.

III. Technical Outline

SAC already utilizes maximum entropy RL which optimizes policies π_θ to maximize both the expected return and the expected entropy of the policy (blue). So to add the Munchausen idea we simply have to add the scaled log policy (red) to the reward r_t inside our Q-function.

$$Q(s_t, a_t) = r_t + \tau[\alpha \ln \pi_\theta(a_t|s_t)]_{l_0}^{0} + \gamma \mathop{\mathbb{E}}_{s_{t+1}\sim p}\left[V(s_{t+1})\right], \tag{1}$$

where

$$V(s_t) = \mathop{\mathbb{E}}_{a_t\sim\pi_\theta}\left[Q(s_t, a_t) - \alpha \ln \pi_\theta(a_t, s_t)\right] \tag{2}$$

$\gamma \in [0, 1)$ is the typical discounting factor for infinite time horizons and s_t, a_t represent the state and action, respectively, at time t (hyperparameters are addressed below).
For the continous action space setup neural networks are used as function approximators for both the Q-function and the policy and instead of running soft policy iteration (alternating policy evaluation and policy improvement) to convergence, we alternate between optimizing both networks via methods like stochastic gradient descent.

A. Automatic Hyperparameter Tuning

M-DQN introduces 3 new hyperparamters.

- $\tau \in [0, 1]$
- $l_0 < 0$ limits the log-policy term, otherwise numerical problems may occur if the policy becomes too deterministic [1].
- α weighs the entropy term relative to the reward and hence defines the randomness of the optimal strategy. (Also called temperature parameter, as the idea comes from energy-based methods.)

Since especially the choice of the temperature parameter is difficult, it was originally fixed manually. An alternative baseline could be random search. We intend to try out an automated strategy to adapt the hyperparameter. Equivalent to the automatic temperature selection derived in [2], section 5. By a gradient-based adjustment of the parameter, the expected entropy of the policy over the visited states is matched to the specified target entropy.

References

[1] Nino Vieillard, Olivier Pietquin and Matthieu Geist (2020). Munchausen Reinforcement Learning. arXiv:2007.14430.

[2] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel and Sergey Levine (2019). Soft Actor-Critic Algorithms and Applications. arXiv:1812.05905.

[3] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, Sergey Levine (2018). Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. arXiv:1801.01290.

[4] Make a four-legged creature walk forward as fast as possible. https://gym.openai.com/envs/Ant-v2/.