# A Note on Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor, Haarnoja et al. [2018a]

**Keng Fu, Hsu**
Department of Computer Science
National Chiao Tung University
ierosodin.cs07g@nctu.edu.tw

## 1 Introduction

Soft actor-critic(SAC) is an algorithm that optimizes a stochastic policy in an off-policy way. The main contribution of this paper (Haarnoja et al. [2018a]) is to introduce maximum entropy to the off-policy reinforcement learning. Instead of using the deterministic policy, SAC tends to use the stochastic policy to maximize entropy objective function. The purpose of using maximum entropy is that the policy is incentivized to explore more widely. The maximum entropy would try to equal the probability mass of each policy, and the policy objective function would be trained to maximize a trade-off between expected return and entropy. Also, increasing entropy results in more exploration and the experiment seems to improve the learning speed compared with the conventional RL objective function.

## 2 Problem Formulation

In this paper (Haarnoja et al. [2018a]), the policy learning is under the continuous state spaces and action spaces. We use $S$ and $A$ to represent the state spaces and the action spaces. The transition probability from $s_t$ and $a_t$ to $s_{t+1}$ can be represented by $p : S \times S \times A$. $r_t$ denotes the environment reward corresponding to state $s_t$ and action $a_t$. $\rho_\pi(s_t)$ and $\rho_\pi(s_t, a_t)$ represents the state and state action marginals of the trajectory distribution induced by a policy $\pi(s_t, a_t)$. Finally, the Markov decision process(MDP) is defined by $(S, A, p, r)$.

To apply maximum entropy to the off policy reinforcement learning, the objective function can be written as:

$$J(\pi) = \sum_{t=0}^{T} E_{(s_t,a_t)\sim\rho_\pi}[r(s_t, a_t) + \alpha H(\pi(\cdot|s_t))],$$

where $\alpha$ is the relative importance of the entropy term against the reward. When $\alpha \to 0$, it would become the conventional objective function, which maximums the expected sum of rewards.

## 3 Theoretical Analysis

### 3.1 Soft policy iteration

We introduce policy evaluation and policy improvement in soft policy iteration and prove that it would converge to the optimal solution.

Preprint. Under review.

### 3.1.1 Policy evaluation

In the policy evaluation step of soft policy iteration, we maximum entropy object as below:

$$J(\pi) = \sum_{t=0}^{T} E_{(s_t,a_t)\sim\rho_\pi}[r(s_t,a_t) + \alpha H(\pi(\cdot|s_t))]$$

We can simply count entropy term into a part of reward:

$$r_{soft}(s_t,a_t) = r(s_t,a_t) + \gamma\alpha E_{s_{t+1}\sim p} H(\pi(\cdot|s_{t+1}))$$

And then the original Q function would become

$$\begin{aligned}
Q_{soft}(s_t,a_t) &= r(s_t,a_t) + \gamma\alpha E_{s_{t+1}\sim p} H(\pi(\cdot|s_{t+1})) + \gamma E_{s_{t+1},a_{t+1}}[Q_{soft}(s_{t+1},a_{t+1})] \\
&= r(s_t,a_t) + \gamma E_{s_{t+1}\sim p, a_{t+1}\sim\pi}[Q_{soft}(s_{t+1},a_{t+1})] + \gamma\alpha E_{s_{t+1}\sim p} H(\pi(\cdot|s_{t+1})) \\
&= r(s_t,a_t) + \gamma E_{s_{t+1}\sim p, a_{t+1}\sim\pi}[Q_{soft}(s_{t+1},a_{t+1})] + \gamma E_{s_{t+1}\sim p} E_{a_{t+1}\sim\pi}[-\alpha\log\pi(a_{t+1}|s_{t+1})] \\
&= r(s_t,a_t) + \gamma E_{s_{t+1}\sim p}[E_{a_{t+1}\sim\pi}[Q_{soft}(s_{t+1},a_{t+1}) - \alpha\log(\pi(a_{t+1}|s_{t+1}))]] \\
&= r(s_t,a_t) + \gamma E_{s_{t+1},a_{t+1}}[Q_{soft}(s_{t+1},a_{t+1}) - \alpha\log(\pi(a_{t+1}|s_{t+1}))]
\end{aligned}$$

We rewrite the Bellman backup operator $T^\pi$ as below:

$T^\pi Q(s_t,a_t) = r(s_t,a_t) + \gamma E_{s_{t+1}\sim p}[V(s_t)]$, where $V(s_t) = E_{a_t\sim\pi}[Q(s_t,a_t) - \alpha\log\pi(a_t|s_t)]$

We can apply the standard convergence results for policy evaluation and prove that it would converge because we view the maximum entropy term as a part of the reward.

### 3.1.2 Policy improvement

In the policy improvement, We use the $Q$ function and $V$ function corresponding to the old policy to get the new policy. Here, we apply KL-divergence to choose the best policy for each state.

$$\begin{aligned}
\pi_{new}(\cdot|s_t) &= \arg\min_{\pi'\in\prod} D_{KL}(\pi'(\cdot|s_t)||\exp(Q^{\pi_{old}}(s_t,\cdot) - \log Z^{\pi_{old}}(s_t))) \\
&= \arg\min_{\pi'\in\prod} J_{\pi_{old}}(\pi'(\cdot|s_t))
\end{aligned},$$

where $Z$ is the normalize term.

Because we always choose the best policy according to the Q function,

$J_{\pi_{old}}(\pi_{new}(\cdot|s_t)) \leq J_{\pi_{old}}(\pi_{old}(\cdot|s_t))$

$$\begin{aligned}
&\Rightarrow E_{a_t\sim\pi_{new}}[\log\pi_{new}(a_t|s_t) - Q^{\pi_{old}}(s_t,a_t) + \log Z^{\pi_{old}}(s_t)] \leq E_{a_t\sim\pi_{old}}[\log\pi_{new}(a_t|s_t) \\
&\quad - Q^{\pi_{old}}(s_t,a_t) + \log Z^{\pi_{old}}(s_t)] \\
&\Rightarrow E_{a_t\sim\pi_{new}}[\log\pi_{new}(a_t|s_t) - Q^{\pi_{old}}(s_t,a_t)] \leq E_{a_t\sim\pi_{old}}[\log\pi_{new}(a_t|s_t) - Q^{\pi_{old}}(s_t,a_t)] \\
&\Rightarrow E_{a_t\sim\pi_{new}}[Q^{\pi_{old}}(s_t,a_t) - \log\pi_{new}(a_t|s_t)] \geq V^{\pi_{old}}(s_t)
\end{aligned}$$

Then from the Bellman equation,

$$\begin{aligned}
Q^{\pi_{old}}(s_t,a_t) &= r(s_t,a_t) + \gamma E_{s_{t+1}\sim p}[V^{\pi_{old}}(s_{t+1})] \\
&\leq r(s_t,a_t) + \gamma E_{s_{t+1}\sim p}[E_{a_t\sim\pi_{new}}[Q^{\pi_{old}}(s_{t+1},a_{t+1}) - \log\pi_{new}(a_{t+1}|s_{t+1})]] \\
&\vdots \\
&\leq Q^{\pi_{new}}(s_t,a_t)
\end{aligned}$$

We can repeatedly apply soft policy and soft policy improvement to get the final policy $\pi^*$. Assume that the optimal policy is $\bar{\pi}$. If $\pi^* \neq \bar{\pi}$, $J_{\bar{\pi}}(\bar{\pi}(\cdot|s_t)) < J_{\bar{\pi}}(\pi^*(\cdot|s_t)) \Rightarrow Q^{\bar{\pi}}(s_t,a_t) < Q^{\pi^*}(s_t,a_t)$

However, $Q^{\pi^*}(s_t,a_t)$ must be larger than $Q^{\pi}(s_t,a_t)$ for any other $(s_t,a_t) \in S\times A$, $\pi \neq \pi^*$, a contradiction.

Hence, soft policy iteration would converge to the best Q function and the corresponding policy.

### 3.2 Soft actor-critic

Here, we use the neural network to parameterize state value function, soft Q function, and a tractable policy. Though we can get the state value from Q function and policy function, we still separate it for stable training.

#### 3.2.1 Value function

The goal of the soft value function is to minimize the squared residual error as below.

$J_V(\psi) = E_{s_t \sim D}[\frac{1}{2}(V_\psi(s_t) - E_{a_t \sim \pi_\phi}[Q_\theta(s_t, a_t) - \log \pi_\phi(a_t|s_t)])^2]$,

where $D$ is the reply buffer or the history data of state and action.

And the corresponding gradient is

$\nabla_\psi J_V(\psi) = \nabla_\psi V_\psi(s_t)(V_\psi(s_t) - Q_\theta(s_t, a_t) + \log \pi_\phi(a_t|s_t))$

Though we have the action for this state in the reply buffer, we choose the action according to the current policy for the stability of training.

Also, a newer version of SAC (Haarnoja et al. [2018b]) learns the Q functions instead of a value function for stable training.

#### 3.2.2 Q function

We train the policy network to minimize the expected KL-divergence.

$$\begin{aligned} J_\pi(\phi) &= E_{s_t \sim D}[D_{KL}(\pi_\phi(\cdot|s_t)||\frac{\exp(Q_\theta(s_t, \cdot))}{Z_\theta(s_t)})] \\ &= E_{s_t \sim D, a_t \sim \pi_\phi}[\log \pi_\phi(a_t|s_t) - Q_\theta(s_t, a_t) + \log Z(s_t)] \end{aligned}$$

Here, because the Q function is represented by a neural network, which can be differentiated, we apply the reparameterization trick.

$a_t = f_\phi(\epsilon_t; s_t)$, where $\epsilon$ is an input noise vector from a Gaussian distribution.

And then we can rewrite the objective function:

$J_\pi(\phi) = E_{s_t \sim D, \epsilon_t \sim N}[\log \pi_\phi(f_\phi(\epsilon_t; s_t)|s_t) - Q_\theta(s_t, f_\phi(\epsilon_t; s_t))]$

And the corresponding gradient is

$$\begin{aligned} \nabla_\phi J_\pi(\phi) &= \nabla_\phi(\log \pi_\phi(a_t|s_t) - Q_\theta(s_t, a_t)) + (\nabla_{a_t} \log \pi_\phi(a_t|s_t) - Q_\theta(s_t, a_t))\nabla_\phi f_\phi(\epsilon_t; s_t) \\ &= \nabla_\phi \log \pi_\phi(a_t|s_t) + (\nabla_{a_t} \log \pi_\phi(a_t|s_t) - Q_\theta(s_t, a_t))\nabla_\phi f_\phi(\epsilon_t; s_t) \end{aligned}$$

## 4 Conclusion

The paper presents the soft actor-critic network, which trains a stochastic policy with entropy regularization, and explores in an off-policy way. A newer version of SAC (Haarnoja et al. [2018b]) adds a coefficient $\alpha$ to the entropy regularization term to explicitly control the explore-exploit trade-off, with higher corresponding to more exploration, and lower corresponding to more exploitation. From the result, SAC is a stable and robust off-policy reinforcement learning algorithm compared with other deterministic networks, and the curve of average return seems to be smoother. There are still lots of further exploration of maximum entropy methods like incorporating second-order information (e.g., trust regions), and it would be a different field of reinforcement learning research. Here are some researches related to the soft actor-critic:

- *Reinforcement Learning with Deep Energy-Based Policies* (Haarnoja et al. [2017])
- *Soft Actor-Critic Algorithms and Applications* (Haarnoja et al. [2018b])

## References

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

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications, 2018b.

Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies, 2017.