Contents

1 Soft-actor critic (SAC) **1**
1.1 Summary 1
1.2 Detail of the implementations 2
1.2.1 Soft action-value function Q 2
1.2.2 policy π 2
1.3 Side note - principle of maximum entropy 3
1.4 Reference 3

1 Soft-actor critic (SAC)

1.1 Summary

Desirable properties

1. Sample efficiency –> Should be able to learn with little observations.

2. No sensitive hyperparameters.

3. Off-policy learning. We can use data collected during a previous task.

On-policy algorithms such as Trust Region Policy Optimization (TRPO) or Proximal Policy Optimization (PP0) suffers from 1) while off-policy algorithms such as Deep Q learning suffer from 2). Soft-actor critic try to take the best of both world by adding an entropy term to the objective.

The SAC objective is:

$$J(\pi) = \mathop{\mathbb{E}}_{\pi}\left(\sum_{t}\left(R(s_t, a_t) - \alpha\log(\pi(a_t|s_t)\right)\right) \tag{1}$$

where s_t, a_t are the state and action resp. The entropy term

1. encourage exploration

2. allow the learning process to capture multiples modes of near optimal behavior by assigning them equal probability weights.

3. Also, the authors argues that this entropy allow the agent to learn considerably faster.

1.2 Detail of the implementations

SAC makes use of 2 networks:

1. A soft Q-function Q parameterized by θ
2. A policy π parameterized by ϕ

1.2.1 Soft action-value function Q

The soft action-value function is trained to minimize the soft version of the Bellman estimates

$$J_q(\theta) = \mathop{\mathbb{E}}_{s_t, a_t \sim D} \big[\frac{1}{2}\big(Q_\theta(s_t, a_t) - \hat{Q}(s_t, a_t)\big)\big] \tag{2}$$

where our estimate $\hat{Q}$ is

$$\hat{Q}(s_t, a_t) = \mathop{\mathbb{E}}_{s_{t+1}} \big[r(s_t, a_t, s_{t+1}) + \gamma V_{\bar{\theta}}(s_{t+1})\big] \tag{3}$$

and the soft value function V is implicitly parameterized by θ through its definition

$$V_(s_t) = \mathop{\mathbb{E}}_{a_t \sim \pi} (Q(s_t, a_t) - \log(\pi(a_t|s_t)) \tag{4}$$

This objective can be optimised via gradient descent following

$$\nabla_\theta J_Q(\theta) = \nabla_\theta Q_\theta(a_t, s_t)(Q_\theta(a_t, s_t) - \big(r(s_t, a_t) + \gamma\big(Q_{\bar{\theta}}(s_{t+1} a_{t+1}) - \alpha \log(\pi(a_{t+1}|s_{t+1}))\big)\big)) \tag{5}$$

As in deep-Q nework, they use a target network $Q_{\bar{\psi}}$ updated only every N iterations or using an exponential moving average on θ.

1.2.2 policy π

The policy is trained by minimising the following KL divergence

$$J_\pi(\phi) = \mathop{\mathbb{E}}_{s_t \sim D} \big[D_{KL}\big(\pi_\phi(\dot{|}s_t)|| \frac{\exp(\frac{1}{\alpha} Q_\theta(s_t, \dot{}))}{Z_\theta(s_t)})\big] \tag{6}$$

where Z is a partition function and does not contributes to the gradient. Ignoring the partition function, multiplying by and plugging in the definition of the KL divergence, we get

$$J_\pi(\phi) = \mathop{\mathbb{E}}_{s_t \sim D} \big[\mathop{\mathbb{E}}_{a_t \sim \pi_\phi} \big[\alpha \log(\pi(a_t|s_t)\big) - Q_\theta(s_t, a_t)\big]\big] \tag{7}$$

1.3 Side note - principle of maximum entropy

This principle prescribes the use of the least committed distribution fitting the observation when working with a ill-posed problem. In other words, using a Dirac distribution which agree with your single data point to model the source is not a good idea.

$$H(\pi) = \mathbb{E}\left(-\log(\pi(a_t, s_t)\right) \tag{8}$$

1.4 Reference

Soft-Actor critic and applications Soft Actor-Critic Soft Actor-Critic Demystified ReparameterizationTrick open-ai-SAC