Intelligent Control:

*Implementing Soft Actor-Critic (SAC) from Scratch*

ΜΑΡΚΑΝΤΩΝΑΤΟΥ ΕΙΡΗΝΗ | up1095489

ΣΟΛΩΜΟΝΙΔΗ ΕΛΕΥΘΕΡΙΑ | up1095489

**ΕΤΟΣ:** 4ο

ΕΥΦΥΗΣ ΕΛΕΓΧΟΣ| 23/6/2025

Policy: Actor

Value: Critic

Actor- Critic Issues:

* *Exploration:*

Doesn’t escape from local optima

* *Generalization and safety:*

Performs poorly in unseen situations. The actor learns to be good only on one strategy.

S- AC:

Instead of only asking the Actor to maximize his reward when training, the critic will ask the actor to do so while acting as randomly as possible. That is maximizing the entropy of its current policy. “Soft” means “entropy regularization”. The entropy of the policy being maximized, the actor is incentivized to explore alternative strategies and to recover from unexpected situations.

Consequently, it introduces a new value function, the soft action – value function.

SAC parameterizes a policy and a soft Q- value function .

How does it work?

I want to find the optimal policy.

In most RL algorithms:

In A2C- like algorithms, the entropy is used externally as an additional exploration mechanism.

SAC changes the traditional MDP objective. Thus, it converges towards different solutions.

The maximum entropy objective generalizes the standard objective by augmenting it with an entropy term, such that the optimal policy additionally aims to maximize its entropy at each visited state:

*,*

Where a is the temperature term that determines the relative importance of the entropy term versus the reward, and thus controls the stochasticity of the optimal policy.

In standard RL :

*Bellman Optimality Equation*:

,

where

*Soft Bellman Equation:*

where

Instead of picking just the **best** action, it considers **all possible actions** based on their likelihood under the current policy.

It also adds the entropy term

Can entropy be whatever you want?

No. It is designed to measure uncertainty or randomness in a probability distribution. The term is the probability that the policy chooses action in .

* When is close to 1 is near 0 (low surprise)
* When is close to 0 is a large negative number (high surprise)
* **So is small for confident choices, large for unlikely ones**

In SAC we want policies that balance high reward with high entropy. So penalizes overconfident actions and rewards diverse exploratory behavior.

Σχήμα ισως?

I want to find the Bellman operator that will allow me to optimize my Q- values.

Basic Idea behind Actor Critic:

How can we improve a policy?

*Policy Iteration*

We start from an initial policy

1. Policy Evaluation

Given a policy π (a mapping from states to actions) compute , which estimates how good it is to follow π from each state (By solving Bellman expectation)

1. (Greedy) Policy Improvement

Use current value function to find . For each state, choose action that maximises expected return assuming one-step ??? and then follow current Value Function. Stop when policy stops changing.

In SAC we use a policy iteration formulation, where we instead evaluate the Q-function of the current policy and update the policy through an off policy gradient rule.

For a fixed policy, the soft Q- value can be computed iteratively, starting from any function and repeatedly applying Equation.

In the policy improvement step, we update the policy towards the exponential of the new soft Q function. We make actions that have higher soft Q-values more probable (softmax like update).

In reinforcement learning environments, actions can typically be either discrete or continuous. For discrete action spaces, action probabilities can be modeled directly using techniques like the softmax function. In contrast, continuous action spaces require modeling a probability density function, such as a Gaussian distribution.

A Gaussian Distribution is a common way to model continuous stochastic policies:

,

where s is the current state.

Why are Gaussians tractable?

If you define a Gaussian as shown in Equation 1 then sampling an action is:

*(reparameterisation trick)*

Since in practice we prefer policies that are tractable we will restrict the policy to Gaussians. In the Soft Actor-Critic (SAC) algorithm, Gaussian distributions are used to represent stochastic policies in continuous action spaces.

Policy Improvement Step:

The partition function Z normalizes the distribution, and while it is intractable in general, it does not contribute to the gradient with respect to the new policy and can thus be ignored.

Implementation:

3 Neural Networks. We will consider a parameterized soft Q-function, and a tractable policy ). The Q neural network is trained to approximate the **soft Bellman operator** while the policy network can be modeled as a Gaussian with mean and covariance given by neural networks:

: The most likely action

: randomness around that action

Why output instead of ?

* σ must be positive (it is a standard deviation)
* If you output it directly, your network might output negative or zero values, which would break the Gaussian:

3 objective functions: