

CS-866 Deep Reinforcement Learning

Improved Policy-Based Learning III: Soft Actor-Critic (SAC)



Nazar Khan
Department of Computer Science
University of the Punjab

Exploration Challenges in Deep RL

Brittleness in Deep RL:

- ▶ Only a fraction of the state space is sampled
- ▶ Agents often get stuck in **local optima**
- ▶ Performance can vary significantly due to:
 - ▶ Hyperparameter choices
 - ▶ Random number seed

Key Insight: For large problems, **exploration is crucial**, for both value-based and policy-based RL methods.

Goal: Allow the agent to try actions that currently seem suboptimal, to avoid brittle policies.

Exploration Strategies Overview

Too little exploration:

- ▶ Brittle policies
- ▶ Highly sensitive to initialization and hyperparameters

Exploration mechanisms:

- ▶ Add noise to **deterministic** policies
- ▶ Maintain entropy in **stochastic** policies

Bottom line: We need mechanisms to prevent premature policy collapse.

Deterministic Policy & Noise

Deterministic policy: $\pi_{\theta}(s) \rightarrow a$

Add exploration noise:

- ▶ Continuous actions: Gaussian noise
- ▶ Discrete actions: Dirichlet noise

Example (1D continuous action):

$$\pi_{\theta, \text{behavior}}(a|s) = \pi_{\theta}(s) + \mathcal{N}(0, \sigma)$$

$\mathcal{N}(\mu, \sigma)$ = Normal distribution, σ = exploration hyperparameter

Stochastic Policies & Entropy Regularization (SAC)

Stochastic policy: $\pi(a|s)$

$$\pi_{\theta, \text{behavior}}(a|s) = \pi_{\theta}(a|s)$$

Exploration comes from sampling, but:

- Policy distribution may collapse (too narrow)
- Risk: loss of exploration

Solution: Entropy regularization

Add **entropy bonus** to objective:

$$\theta_{t+1} = \theta_t + R \nabla_{\theta} \log \pi_{\theta}(a_t|s_t) + \eta \nabla_{\theta} H[\pi_{\theta}(\cdot|s_t)]$$

$$H = - \sum_i p_i \log p_i \quad (\text{in SAC: } -\log \pi_{\theta}(a|s))$$

$\eta > 0$: entropy weighting parameter

Why Entropy Helps

High-entropy policies:

- ▶ Encourage broad exploration
- ▶ Avoid collapsing distribution
- ▶ Faster learning due to better exploration

Most policy gradient algorithms (A3C, TRPO, PPO):

- ▶ Optimize expected return only
- ▶ Do not explicitly optimize entropy

SAC advantage:

- ▶ Optimizes return *and* entropy
 - ▶ More stable across random seeds
 - ▶ Less sensitive to hyperparameters
-

SAC and Off-Policy Learning

SAC also uses a **replay buffer**:

- ▶ Many policy methods are **on-policy**
 - ▶ (A3C, TRPO, PPO)
 - ▶ Require fresh data each update
 - ▶ High sample cost
- ▶ SAC is **off-policy**
 - ▶ Learns from past experience
 - ▶ Improves sample efficiency
 - ▶ Avoids local maxima via replay diversity

Challenge: Off-policy RL can destabilize learning

SAC result: Stable off-policy performance + efficient exploration
