Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor

Ishaq Hamza , K Sahapthan , Rohit Jorige

May 8, 2025

Outline

Motivation for Soft Actor-Critic (SAC)

- Limitations of existing Model-Free Deep RL: Sample complexity and brittleness of hyperparameters
- Demonstration of stability across different random seeds
- By combining off-policy updates with a stable stochastic actor-critic formulation, SAC achieves state-of-the-art performance on a range of continuous control benchmark tasks, outperforming prior on-policy and off-policy methods

Maximum Entropy RL Framework

Standard RL Objective: Traditional reinforcement learning maximizes the expected sum of rewards:

$$\sum_t \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r(s_t, a_t)]$$

Maximum Entropy Objective: To encourage stochastic policies, we augment the objective with an entropy term:

$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}\left[r(s_t, a_t) + \alpha H(\pi(\cdot|s_t))\right] \tag{1}$$

- $H(\pi(\cdot|s_t)) = -\mathbb{E}_{a\sim\pi(\cdot|s_t)}[\log \pi(a|s_t)]$
- The policy is incentivized to explore more widely while avoiding clearly unpromising actions and can capture multiple modes of near-optimal behavior
- Experiments show that it significantly improves learning speed over state-of-the-art RL methods optimizing the conventional objective

Definitions

- **Maximum Entropy Objective:**

$$J(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \left(r(s_t, a_t) + H(\pi(\cdot|s_t))\right)\right]$$

where:

  - $r(s_t, a_t)$: Reward.
  - $H(\pi) = -\sum_{a_t} \pi(a_t|s_t) \log \pi(a_t|s_t)$: Entropy, with $|A| < \infty$.
  - $\gamma \in [0, 1)$: Discount factor.

- **Soft State-Value Function:**

$$V^\pi(s_t) = \mathbb{E}_{a_t \sim \pi}\left[Q^\pi(s_t, a_t) - \log \pi(a_t|s_t)\right]$$

- **Soft Action-Value Function:**

$$Q^\pi(s_t, a_t) = r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim p(\cdot|s_t, a_t)}[V^\pi(s_{t+1})]$$

Soft Bellman Operator

- **Modified Soft Bellman Backup Operator:**

$$T^\pi Q(s_t, a_t) = r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim p}[V^\pi(s_{t+1})]$$

where:

$$V^\pi(s_t) = \mathbb{E}_{a_t \sim \pi}\left[Q^\pi(s_t, a_t) - \log \pi(a_t|s_t)\right]$$

- **Entropy-Augmented Reward:**

$$r^\pi(s_t, a_t) \triangleq r(s_t, a_t) + \mathbb{E}_{s_{t+1} \sim p}[H(\pi(\cdot|s_{t+1}))]$$

Soft Policy Evaluation

- **Lemma 1 (Soft Policy Evaluation):**
  - Consider T^π and $Q_0 : S \times A \to \mathbb{R}$, with $|A| < \infty$.
  - Define sequence: $Q_{k+1} = T^\pi Q_k$.
  - Then: $Q_k \to Q^\pi$, the soft Q-value of π, as $k \to \infty$.
- **Process:**
  - Iteratively apply T^π.
  - Convergence guaranteed for finite action spaces.
- *Proof:* Using the entropy-augmented reward, one can write the Bellman update as

$$Q(s_t, a_t) \leftarrow r^\pi(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim p, a_{t+1} \sim \pi}[Q(s_{t+1}, a_{t+1})]$$

  Since π is fixed for policy evaluation, the standard convergence results apply.

Soft Policy Improvement

- **Goal:** Update policy towards exponential of Q-function, projected onto Π.
- **Policy Update:**

$$\pi_{\text{new}} = \arg\min_{\pi' \in \Pi} D_{\text{KL}}\left(\pi'(\cdot|s_t) \,\middle\|\, \frac{\exp(Q_{\pi_{\text{old}}}(s_t, \cdot))}{Z_{\pi_{\text{old}}}(s_t)}\right)$$

where $Z_{\pi_{\text{old}}}(s_t) = \sum_{a_t} \exp(Q_{\pi_{\text{old}}}(s_t, a_t))$.

- **Lemma 2 (Soft Policy Improvement):**
  - For $\pi_{\text{old}} \in \Pi$, and π_{new} as above:

$$Q_{\pi_{\text{new}}}(s_t, a_t) \geq Q_{\pi_{\text{old}}}(s_t, a_t), \quad \forall (s_t, a_t) \in S \times A$$

Proof of Lemma 2: Part 2

- Let $\pi_{\text{old}} \in \Pi$, with $Q_{\pi_{\text{old}}}$, $V_{\pi_{\text{old}}}$. Define:

$$\pi_{\text{new}}(a_t|s_t) = \arg\min_{\pi' \in \Pi} D_{\text{KL}} \left(\pi'(a_t|s_t) \middle\| \frac{\exp(Q_{\pi_{\text{old}}}(s_t, a_t))}{Z_{\pi_{\text{old}}}(s_t)} \right).$$

- Since $\pi_{\text{new}} = \pi_{\text{old}}$ is feasible:

$$J_{\pi_{old}}(\pi_{new}(\cdot|s_t)) = \mathbb{E}_{a_t \sim \pi_{\text{new}}}[\log \pi_{\text{new}}(a_t|s_t) - Q_{\pi_{\text{old}}}(s_t, a_t) + \log Z_{\pi_{\text{old}}}(s_t)]$$

$$\leq \mathbb{E}_{a_t \sim \pi_{\text{old}}}[\log \pi_{\text{old}}(a_t|s_t) - Q_{\pi_{\text{old}}}(s_t, a_t) + \log Z_{\pi_{\text{old}}}(s_t)]$$

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

Proof of Lemma 2: Part 3

- Simplify, as $Z_{\pi_{\text{old}}}$ depends only on state:

$$\mathbb{E}_{a_t \sim \pi_{\text{new}}}[Q_{\pi_{\text{old}}}(s_t, a_t) - \log \pi_{\text{new}}(a_t|s_t)] \geq V_{\pi_{\text{old}}}(s_t).$$

- Soft Bellman equation:

$$Q_{\pi_{\text{old}}}(s_t, a_t) = r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim p}[V_{\pi_{\text{old}}}(s_{t+1})].$$

- Expand $Q_{\pi_{\text{old}}}$ repeatedly, applying the upper bound on $V_{\pi_{\text{old}}}$ and Bellman equation for $Q_{\pi_{\text{old}}}$:

$$Q_{\pi_{\text{old}}}(s_t, a_t) \leq Q_{\pi_{\text{new}}}(s_t, a_t).$$

- By Lemma 1, $Q_{\pi_{\text{new}}}$ converges to the soft Q-value of π_{new}.

Soft Policy Iteration Theorem

- **Theorem 1 (Soft Policy Iteration):**
  - Repeated soft policy evaluation and improvement from any $\pi \in \Pi$ converges to π^*:

$$Q_{\pi^*}(s_t, a_t) \geq Q_\pi(s_t, a_t), \quad \forall \pi \in \Pi,\ (s_t, a_t) \in S \times A.$$

- *Proof:* Let π_i be the policy at iteration i. By Lemma 2, the sequence Q_{π_i} is monotonically increasing. By virtue of upper-boundedness, the sequence converges to some π^*. At convergence, it must be the case that $J_{\pi^*}(\pi^*(\cdot|s_t)) < J_{\pi^*}(\pi(\cdot|s_t))$ for all $\pi \in \Pi$, $\pi \neq \pi^*$. Using the same iterative argument as in the proof of Lemma 2, we get $Q_{\pi^*}(s_t, a_t) > Q_\pi(s_t, a_t)$ for all $(s_t, a_t) \in S \times A$, that is, the soft value of any other policy in Π is lower than that of the converged policy. Hence π^* is optimal in Π.
- **Note.** Though the paper does not mention it, all fixed-behavior algorithms' results are maintained. (TD(0), TD-function approximation)

Soft Actor-Critic Algorithm Overview

- Large continuous domains require us to derive a practical approximation to soft policy iteration
- Function approximators are used for both the Q-function and the Policy, and instead of running evaluation and improvement to convergence, both networks are optimized alternatively with stochastic gradient descent
- We consider a parameterized soft state value function $V_\psi(s_t)$, soft Q-function $Q_\theta(s_t, a_t)$, and a tractable policy $\pi_\phi(a_t|s_t)$
- Value functions are modeled as expressive neural networks, and the policy as a Gaussian with mean and covariance given by neural networks
- Even though there is no need in principle to include a separate function approximator for the state value (since it is related to the Q-function and policy) it can stabilize training and is convenient to train simultaneously with the other networks

Training the Value Network

The soft value function is trained to minimize the squared residual error:

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

where D is the distribution of previously sampled states and actions (replay buffer). The gradient of this objective can be estimated with an unbiased estimator:

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

where the actions are sampled according to the current policy instead of the replay buffer.

Training the Q-Function Network

The soft Q-function parameters can be trained to minimize the soft Bellman residual:

$$J_Q(\theta) = \mathbb{E}_{(s_t,a_t)\sim D}\left[\frac{1}{2}\left(Q_\theta(s_t,a_t) - \hat{Q}(s_t,a_t)\right)^2\right]$$

where the target value is given by:

$$\hat{Q}(s_t,a_t) = r(s_t,a_t) + \gamma\mathbb{E}_{s_{t+1}\sim p}\left[V_{\bar{\psi}}(s_{t+1})\right]$$

This objective can be optimized with stochastic gradients:

$$\nabla_\theta J_Q(\theta) = \nabla_\theta Q_\theta(s_t,a_t)\left(Q_\theta(s_t,a_t) - r(s_t,a_t) - \gamma V_{\bar{\psi}}(s_{t+1})\right)$$

The update makes use of a target value network $V_{\bar{\psi}}$, where $\bar{\psi}$ can be an exponentially moving average of the value network weights, which has been shown to stabilize training.

Training the Policy Network

Finally, the policy parameters can be learned by directly minimizing the expected KL-divergence :

$$J_\pi(\phi) = \mathbb{E}_{s_t \sim D}\left[D_{\mathrm{KL}}\left(\pi_\phi(\cdot|s_t) \middle\| \frac{\exp(Q_\theta(s_t, \cdot))}{Z_\theta(s_t)} \right) \right]$$

Since the Q-function is represented by a neural network and is differentiable, we can reparameterize the policy using a neural network transformation for a lower-variance estimator, $a_t = f_\phi(\epsilon_t, s_t)$, where ϵ_t is an input noise vector, sampled from some fixed distribution, such as a spherical Gaussian. The objective can be rewritten as:

$$J_\pi(\phi) = \mathbb{E}_{s_t \sim D, \epsilon_t \sim \mathcal{N}}\left[\log \pi_\phi(f_\phi(\epsilon_t; s_t)|s_t) - Q_\theta(s_t, f_\phi(\epsilon_t; s_t))\right]$$

The gradient of this objective can be approximated as:

$$\nabla_\phi J_\pi(\phi) = \nabla_\phi \log \pi_\phi(a_t|s_t) + (\nabla_{a_t} \log \pi_\phi(a_t|s_t) - \nabla_{a_t} Q(s_t, a_t)) \nabla_\phi f_\phi(\epsilon_t; s_t)$$

where a_t is evaluated at $f_\phi(\epsilon_t; s_t)$

SAC Algorithm

- The algorithm also makes use of the minimum of two Q-functions for the value and policy gradients (trained to independently optimize $J_Q(\theta_i)$ to mitigate positive bias in the policy improvement step, which is known to degrade the performance of value-based methods
- In practice, we take a single environment step with current policy (for collecting experiences) followed by one or several gradient steps (using batched gradients from replay buffer)

Algorithm 1 Soft Actor-Critic

Initialize parameter vectors $\psi, \bar{\psi}, \theta, \phi$.
for each iteration **do**
  for each environment step **do**
    $\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t|\mathbf{s}_t)$
    $\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{a}_t, r(\mathbf{s}_t, \mathbf{a}_t), \mathbf{s}_{t+1})\}$
  end for
  for each gradient step **do**
    $\psi \leftarrow \psi - \lambda_V \hat{\nabla}_\psi J_V(\psi)$
    $\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i)$ for $i \in \{1, 2\}$
    $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$
    $\bar{\psi} \leftarrow \tau\psi + (1 - \tau)\bar{\psi}$
  end for
end for

Automatic Temperature Tuning - 1: The Problem of Manual α

The Challenge of Setting Temperature (α)

- Unlike standard Reinforcement Learning, where reward scaling is often done, in Maximum Entropy RL (the foundation of SAC), the reward magnitude is intrinsically linked to the optimal temperature α.
- If α is not carefully chosen to match the task's reward scale, the learning process in SAC can become unstable or yield suboptimal policies.
- Manually tuning this crucial hyperparameter α for each new and diverse environment is a laborious process, demanding significant time investment and expert intuition.

Automatic Temperature Tuning - 2: Entropy as a Constraint

Moving Beyond Fixed α: Automatic Entropy Adjustment

- SAC innovatively reframes the problem by treating the policy's entropy not as a direct consequence of a fixed α, but rather as an explicit **constraint** to be satisfied during optimization.

The Constrained Optimization Problem: The goal is to maximize the agent's cumulative reward:

$$\max_{\pi_{0:T}} \mathbb{E}_{\rho_\pi}\left[\sum_{t=0}^{T} r(s_t, a_t)\right]$$

While ensuring a minimum level of policy stochasticity (exploration) at each timestep:

$$\mathbb{E}_{(s_t,a_t)\sim\rho_\pi}\left[-\log \pi_t(a_t \mid s_t)\right] \geq \mathcal{H} \; \forall t$$

Here, $\mathcal{H}$ represents the desired minimum expected entropy, often related to the dimensionality of the action space.

Automatic Temperature Tuning - 3: Duality and Iterative Optimization

Lagrangian Duality

- To handle this constrained optimization, SAC employs the Lagrangian duality. This introduces a dual variable $\alpha_t \geq 0$ for each entropy constraint, effectively transforming the problem into an unconstrained one. This dual variable α_t will serve as our adaptive temperature.

The Dual Problem :

$$\min_{\alpha_T \geq 0} \max_{\pi_T} \mathbb{E}[r(s_T, a_T) - \alpha_T \log \pi(a_T|s_T)] - \alpha_T \mathcal{H}$$

Learning the Optimal Temperature (Step t): The optimal

temperature α_t^* is learned by minimizing the following expectation, aiming to keep the policy's entropy close to the target $\mathcal{H}$:

$$\alpha_t^* = \arg\min_{\alpha_t} \mathbb{E}_{a_t \sim \pi_t^*} [-\alpha_t \log \pi_t^*(a_t|s_t; \alpha_t) - \alpha_t \mathcal{H}]$$

Automatic Temperature Tuning - 4: Practical Implementation with Gradient Descent

Implementing Automatic Tuning with Neural Networks

- In practical deep reinforcement learning implementations of SAC, where policies and Q-functions are represented by neural networks, these optimization problems are solved using stochastic gradient descent. Similarly, the temperature α is adapted through gradient-based methods.

The Learnable Temperature Loss: A loss function $J(\alpha)$ is defined to guide the adjustment of α:

$$J(\alpha) = \mathbb{E}_{a_t \sim \pi_t}[-\alpha \log \pi_t(a_t|s_t) - \alpha\bar{\mathcal{H}}]$$

Here, $\bar{\mathcal{H}}$ is the user-defined target entropy. **The Gradient for Temperature Update:** The gradient of this loss with respect to α is used to update the temperature:

$$\nabla_\alpha J(\alpha) = \mathbb{E}_{a_t \sim \pi_t}[-\log \pi_t(a_t|s_t) - \bar{\mathcal{H}}]$$

Automatic Temperature Tuning Algorithm

Algorithm 1 Soft Actor-Critic

Input: θ_1, θ_2, ϕ ▷ Initial parameters

$\bar{\theta}_1 \leftarrow \theta_1, \bar{\theta}_2 \leftarrow \theta_2$ ▷ Initialize target network weights

$\mathcal{D} \leftarrow \emptyset$ ▷ Initialize an empty replay pool

for each iteration **do**

  for each environment step **do**

    $\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t|\mathbf{s}_t)$ ▷ Sample action from the policy

    $\mathbf{s}_{t+1} \sim p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ ▷ Sample transition from the environment

    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{s}_t, \mathbf{a}_t, r(\mathbf{s}_t, \mathbf{a}_t), \mathbf{s}_{t+1})\}$ ▷ Store the transition in the replay pool

  end for

  for each gradient step **do**

    $\theta_i \leftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i)$ for $i \in \{1, 2\}$ ▷ Update the Q-function parameters

    $\phi \leftarrow \phi - \lambda_\pi \hat{\nabla}_\phi J_\pi(\phi)$ ▷ Update policy weights

    $\alpha \leftarrow \alpha - \lambda \hat{\nabla}_\alpha J(\alpha)$ ▷ Adjust temperature

    $\bar{\theta}_i \leftarrow \tau \theta_i + (1 - \tau)\bar{\theta}_i$ for $i \in \{1, 2\}$ ▷ Update target network weights

  end for

end for

Output: θ_1, θ_2, ϕ ▷ Optimized parameters

Results

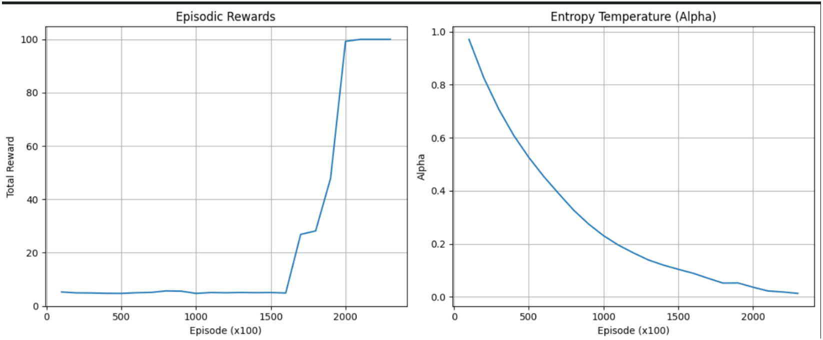


Figure: Varying alpha for inverted pendulum

Introduction to Entropy Maximization in Multi-Agent RL

- Multi-agent RL involves multiple agents learning policies π^i in a shared environment.
- Entropy maximization encourages exploration by adding an entropy term $H(\pi^i)$ to the reward.
- Objective: Balance exploitation (reward) and exploration (entropy).

Objective and Key Functions

- **Objective Function:** For agent i:

$$J(\pi^i) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \left(r^i(s_t, a_t) + \alpha H(\pi^i(\cdot|s_t))\right)\right]$$

where:

  - $r^i(s_t, a_t)$: Reward for agent i.
  - $H(\pi^i) = -\int_{\mathcal{A}^i} \pi^i(a^i|s_t) \log \pi^i(a^i|s_t)\, da^i$: Entropy.
  - $\alpha > 0$: Entropy coefficient.
  - $\gamma \in [0, 1)$: Discount factor.

- **State-Value Function:**

$$V^i(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \left(r^i(s_t, a_t) + \alpha H(\pi^i(\cdot|s_t))\right) \mid s_0 = s\right]$$

- **Action-Value Function:**

$$Q^i(s, a) = r^i(s, a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)}[V^i(s')]$$

Bellman Relations

- Incorporates entropy into standard RL Bellman equations.
- **For Q^i:**

$$Q^i(s,a) = r^i(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)} \left[\mathbb{E}_{a' \sim \pi(\cdot|s')}[Q^i(s',a')] + \alpha H(\pi^i(\cdot|s')) \right]$$

- **For V^i:**

$$V^i(s) = \mathbb{E}_{a \sim \pi(\cdot|s)} \left[r^i(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)}[V^i(s')] \right] + \alpha H(\pi^i(\cdot|s))$$

Policy Evaluation

- For a fixed policy $\pi = (\pi^i, \pi^{-i})$, compute value functions iteratively:
- Update for Q^i:

$$Q^{k+1}(s, a) = r^i(s, a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)}[V^k(s')]$$

- Update for V^i:

$$V^{k+1}(s) = \mathbb{E}_{a \sim \pi(\cdot|s)}[Q^{k+1}(s, a)] + \alpha H(\pi^i(\cdot|s))$$

- Convergence: As $k \to \infty$, $Q^k \to Q^i$ and $V^k \to V^i$.

Policy Improvement Discussion

- Goal: Improve policy π^i for agent i given other agents' policies π^{-i}.
- Marginalize over other agents' actions:

$$\tilde{Q}^i(s, a^i) = \mathbb{E}_{a^{-i} \sim \pi^{-i}(\cdot|s)}[Q^i(s, a^i, a^{-i})]$$

- Optimize:

$$J_s(\pi^i) = \mathbb{E}_{a^i \sim \pi^i}[\tilde{Q}^i(s, a^i)] + \alpha H(\pi^i(\cdot|s))$$

subject to $\int_{\mathcal{A}^i} \pi^i(a^i|s)\, da^i = 1$.

- Optimal policy:

$$\pi_*^i(a^i|s) = \frac{\exp\left(\frac{\tilde{Q}^i(s,a^i)}{\alpha}\right)}{\int_{\mathcal{A}^i} \exp\left(\frac{\tilde{Q}^i(s,b^i)}{\alpha}\right)\, db^i}$$

Multi-Agent Soft Policy Improvement (Non-CE)

Lemma: Let $\pi = (\pi^i_{\text{old}}, \pi^{-i})$, with $\pi^i_{\text{new}} = \arg\min_{\pi' \in \Pi} D_{\text{KL}} \left(\pi' \| \frac{\exp(\tilde{Q}^\pi_i)}{\tilde{Z}^\pi_i}\right)$, where $\tilde{Z}^\pi_i(s_t) = \sum_{a^i} \exp(\tilde{Q}^\pi_i(s_t, a^i_t))$. For $\pi' = (\pi^i_{\text{new}}, \pi^{-i})$:

$$Q^{\pi'}_i(s_t, a_t) \geq Q^\pi_i(s_t, a_t), \quad \forall (s_t, a_t) \in \mathcal{S} \times \mathcal{A}.$$

Proof:

- Since π^i_{new} minimizes the KL-divergence:

$$\mathbb{E}_{a^i_t \sim \pi^i_{\text{new}}}[\tilde{Q}^\pi_i(s_t, a^i_t) - \alpha \log \pi^i_{\text{new}}(a^i_t|s_t)] \geq V^\pi_i(s_t).$$

- By the soft Bellman equation and repeated expansion:

$$\begin{aligned} Q^\pi_i(s_t, a_t) \leq & r^i(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1}} \left[\mathbb{E}_{a_{t+1} \sim \pi'} \left[Q^\pi_i(s_{t+1}, a_{t+1}) \right.\right. \\ & \left.\left. - \alpha \log \pi^i_{\text{new}}(a_{t+1}|s_{t+1}) \right]\right] \leq Q^{\pi'}_i(s_t, a_t). \end{aligned}$$

Policy Iteration for Multi-Agent RL? **not yet.**
Note that we do have policy improvement. But assuming sequential updates for the policies of the agents, **an update in the policy of an agent may reduce the value of the policy of another agent**.
A stronger case is that of Nash equilibrium, where deviation of one agent from it's policy does not benefit it in any way. **Even if we converge to a Nash equilibrium, it is not guaranteed that the policies of the agents are optimal**.
To mitigate converging to sub-optimal equilibria, we propose a novel approach of **cross-entropy regularization** in the next section.

Introduction to Cross-Entropy Regularization

- Extends entropy maximization by adding cross-entropy (CE) terms.
- CE regularization: Encourages similarity or diversity between agents' policies.
- Controlled by coefficient β:
  - $\beta > 0$: Encourages similarity.
  - $\beta < 0$: Encourages diversity.
- Useful for coordination or competition in multi-agent systems.

Objective and Key Functions

- **Objective Function:**

$$J(\pi^i) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \left(r^i(s_t, a_t) + \alpha H(\pi^i(\cdot|s_t)) + \beta \sum_{j \neq i} CE(\pi^i(\cdot|s_t)||\pi^j(\cdot|s_t)) \right)\right]$$

where:

$$CE(\pi^i||\pi^j) = -\int_{\mathcal{A}^i} \pi^i(a^i|s) \log \pi^j(a^i|s)\, da^i$$

- **State-Value Function:**

$$V^i(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \left(r^i + \alpha H(\pi^i) + \beta \sum_{j \neq i} CE(\pi^i||\pi^j) \right) \mid s_0 = s \right]$$

- **Action-Value Function:**

$$Q^i(s, a) = r^i(s, a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)}[V^i(s')]$$

Bellman Relations

- Adjusted for CE regularization.
- **For Q^i:**

$$Q^i(s,a) = r^i(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)}[V^i(s')]$$

- **For V^i:**

$$\begin{aligned} V^i(s) = &\mathbb{E}_{a \sim \pi(\cdot|s)} \Bigg[r^i(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)}[V^i(s')] \\ &+ \alpha H(\pi^i(\cdot|s)) + \beta \sum_{j \neq i} CE(\pi^i(\cdot|s) || \pi^j(\cdot|s)) \Bigg] \end{aligned}$$

Policy Evaluation

- Iterative updates for fixed $\pi = (\pi^i, \pi^{-i})$:
- Update for Q^i:

$$Q^{k+1}(s,a) = r^i(s,a) + \gamma \mathbb{E}_{s' \sim p(\cdot|s,a)}[V^k(s')]$$

- Update for V^i:

$$\begin{aligned} V^{k+1}(s) = & \mathbb{E}_{a \sim \pi(\cdot|s)}[Q^{k+1}(s,a)] \\ & + \alpha H(\pi^i(\cdot|s)) + \beta \sum_{j \neq i} CE(\pi^i(\cdot|s) || \pi^j(\cdot|s)) \end{aligned}$$

- Converges as $k \to \infty$ with augmented reward $r^i \leftarrow r^i + \mathbb{E}_{a \sim \pi(\cdot|s)} \left[\alpha H(\pi^i(\cdot|s)) + \beta \sum_{j \neq i} CE(\pi^i(\cdot|s) || \pi^j(\cdot|s)) \right]$

Policy Improvement Discussion

- Optimize:

$$J_s(\pi^i) = \mathbb{E}_{a^i \sim \pi^i}[\tilde{Q}^i(s, a^i)] + \alpha H(\pi^i(\cdot|s)) + \beta \sum_{j \neq i} CE(\pi^i(\cdot|s)||\pi^j(\cdot|s))$$

- Where:

$$\tilde{Q}^i(s, a^i) = \mathbb{E}_{a^{-i} \sim \pi^{-i}(\cdot|s)}[Q^i(s, a^i, a^{-i})]$$

- Optimal policy now depends on other agents' policies via CE term.

$$\pi^i_*(a^i|s) = \frac{\exp\left(\frac{\tilde{Q}^i(s,a^i)}{\alpha}\right) \cdot \prod_{j \neq i} \left[\pi^j(a^i|s)\right]^{-\beta/\alpha}}{\int_{\mathcal{A}^i} \exp\left(\frac{\tilde{Q}^i(s,b^i)}{\alpha}\right) \cdot \prod_{j \neq i} [\pi^j(b^i|s)]^{-\beta/\alpha} \, db^i}$$

Multi-Agent Soft Policy Improvement (CE)

Lemma: Let
$\pi^i_{\text{new}} = \arg\max_{\pi' \in \Pi} \left\{ \mathbb{E}_{a^i_t \sim \pi'}[\tilde{Q}^\pi_i] + \alpha H(\pi') + \beta \sum_{j \neq i} CE(\pi' \| \pi^j) \right\}$. Then:

$$Q^{\pi'}_i(s_t, a_t) \geq Q^\pi_i(s_t, a_t).$$

Proof sketch:

- $J_s(\pi^i_{\text{new}}) \geq J_s(\pi^i_{\text{old}})$ implies:

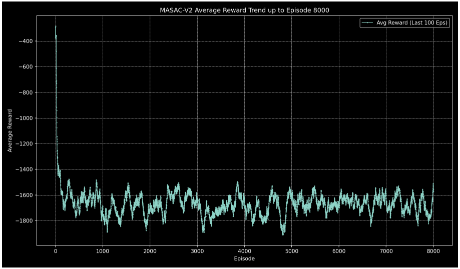
$$\mathbb{E}_{a^i_t \sim \pi^i_{\text{new}}}[\tilde{Q}^\pi_i - \alpha \log \pi^i_{\text{new}}] + \beta \sum_{j \neq i} CE(\pi^i_{\text{new}} \| \pi^j) \geq$$

  $\mathrm{E}_{a^i_t \sim \pi^i_{\text{old}}}[\tilde{Q}^\pi_i - \alpha \log \pi^i_{\text{old}}] + \beta \sum_{j \neq i} CE(\pi^i_{\text{old}} \| \pi^j)$.
- Using the Bellman expansion, the Q-function improves as in the non-CE case, with CE terms adjusting the baseline.

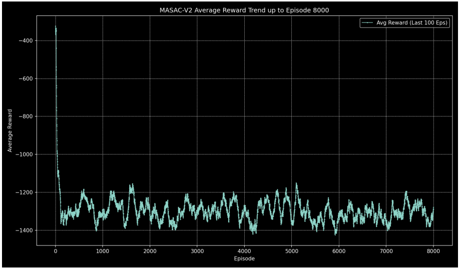
Simple Spread Environment (PettingZoo)

- **Description:** N agents must learn to cover N landmarks while avoiding collisions.
- **Reward:**
  - Global reward based on the distance of the closest agent to each landmark.
  - Local penalty for collisions with other agents.
- **Action Spaces:** Supports both discrete and continuous action spaces.
- **Default Configuration:** 3 agents and 3 landmarks.
- **Expectation:** The system may evolve to some sub-optimal equilibrium. But the cross entropy term could cause an upside variance, thereby increasing the total reward even around the sub-optimal equilibrium.

Results



b = 0, without cross entropy



b = 0.15, with cross entropy

References and Acknowledgments

References

- Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor
- Soft Actor-Critic Algorithms and Applications

Acknowledgments

SAC original paper summary: Sahaptan, Ishaq

Follow up paper summary: Rohit

Experiment design and code (single agent): Rohit, Sahaptan

Multi-agent theoretical extension: Ishaq

Multi-agent code: Rohit

Policy Improvement Lemma

- **Lemma (Multi-Agent Soft Policy Improvement, CE):** Let $\pi = (\pi^i_{\text{old}}, \pi^{-i})$, with:

$$\pi^i_{\text{new}}(\cdot|s_t) = \arg\max_{\pi' \in \Pi} \left\{ \mathbb{E}_{a^i_t \sim \pi'}[\tilde{Q}^\pi_i(s_t, a^i_t)] + \alpha H(\pi'(\cdot|s_t)) + \beta \sum_{j \neq i} CE(\pi'(\cdot|s_t) \| \pi^j(\cdot|s_t)) \right\}.$$

- Then for $\pi' = (\pi^i_{\text{new}}, \pi^{-i})$:

$$Q^{\pi'}_i(s_t, a_t) \geq Q^\pi_i(s_t, a_t), \quad \forall (s_t, a_t) \in \mathcal{S} \times \mathcal{A}.$$

Proof: Part 1

- Define:

$$J_s(\pi^i) = \mathbb{E}_{a_t^i \sim \pi^i}[\tilde{Q}_i^{\pi}(s_t, a_t^i)] - \alpha \mathbb{E}_{a_t^i \sim \pi^i}[\log \pi^i(a_t^i|s_t)] \\ - \beta \sum_{j \neq i} \mathbb{E}_{a_t^i \sim \pi^i}[\log \pi^j(a_t^i|s_t)].$$

- Since π^i_{new} maximizes J_s:

$$J_s(\pi^i_{\text{new}}) \geq J_s(\pi^i_{\text{old}}).$$

- Expand:

$$\mathbb{E}_{a_t^i \sim \pi^i_{\text{new}}}\left[\tilde{Q}_i^{\pi}(s_t, a_t^i) - \alpha \log \pi^i_{\text{new}}(a_t^i|s_t) \qquad - \beta \sum_{j \neq i} \log \pi^j(a_t^i|s_t)\right] \geq$$

$$\mathrm{E}_{a_t^i \sim \pi^i_{\text{old}}}\left[\tilde{Q}_i^{\pi}(s_t, a_t^i) - \alpha \log \pi^i_{\text{old}}(a_t^i|s_t) \qquad - \beta \textstyle\sum_{j \neq i} \log \pi^j(a_t^i|s_t)\right].$$

Proof: Part 2

- The right-hand side:

$$\mathbb{E}_{a_t^i \sim \pi_{\text{old}}^i}\left[\tilde{Q}_i^\pi(s_t, a_t^i) - \alpha \log \pi_{\text{old}}^i(a_t^i|s_t) \qquad - \beta \sum_{j \neq i} \log \pi^j(a_t^i|s_t)\right] = V_i^\pi(s_t),$$

- Since:

$$\mathbb{E}_{a_t^i \sim \pi_{\text{old}}^i}[\tilde{Q}_i^\pi(s_t, a_t^i) - \alpha \log \pi_{\text{old}}^i(a_t^i|s_t)] + \beta \sum_{j \neq i} CE(\pi_{\text{old}}^i \| \pi^j) =$$
$$\mathbb{E}_{a_t \sim \pi}[Q_i^\pi(s_t, a_t) - \alpha \log \pi_{\text{old}}^i(a_t^i|s_t)] + \beta \sum_{j \neq i} CE(\pi_{\text{old}}^i \| \pi^j).$$

- Thus:

$$\mathbb{E}_{a_t^i \sim \pi_{\text{new}}^i}\left[\tilde{Q}_i^\pi(s_t, a_t^i) - \alpha \log \pi_{\text{new}}^i(a_t^i|s_t) \qquad - \beta \sum_{j \neq i} \log \pi^j(a_t^i|s_t)\right] \geq V_i^\pi(s_t).$$

Proof: Part 3

- Soft Bellman equation:

$$Q_i^\pi(s_t, a_t) = r^i(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim p}[V_i^\pi(s_{t+1})].$$

- Substitute:

$$V_i^\pi(s_{t+1}) \leq \mathbb{E}_{a_{t+1}^i \sim \pi_{\text{new}}^i}\left[\tilde{Q}_i^\pi(s_{t+1}, a_{t+1}^i) - \alpha \log \pi_{\text{new}}^i(a_{t+1}^i | s_{t+1}) - \beta \sum_{j \neq i} \log \pi^j(a_{t+1}^i | s_{t+1})\right].$$

- So:

$$Q_i^\pi(s_t, a_t) \leq r^i(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim p}\left[\mathbb{E}_{a_{t+1}^i \sim \pi_{\text{new}}^i}\left[\tilde{Q}_i^\pi(s_{t+1}, a_{t+1}^i) - \alpha \log \pi_{\text{new}}^i(a_{t+1}^i | s_{t+1}) - \beta \sum_{j \neq i} \log \pi^j(a_{t+1}^i | s_{t+1})\right]\right].$$

- Expand:

$$\mathbb{E}_{a^i_{t+1}\sim\pi^i_{\text{new}}}[\tilde{Q}^{\pi}_i(s_{t+1}, a^i_{t+1})] = \mathbb{E}_{a_{t+1}\sim\pi'}[Q^{\pi}_i(s_{t+1}, a_{t+1})],$$

and the CE term adjusts based on π^j.

- Continue:

$$Q^{\pi}_i(s_{t+1}, a_{t+1}) \leq r^i(s_{t+1}, a_{t+1}) + \gamma\mathbb{E}_{s_{t+2}\sim p}\left[\mathbb{E}_{a^i_{t+2}\sim\pi^i_{\text{new}}}\left[\tilde{Q}^{\pi}_i(s_{t+2}, a^i_{t+2}) - \alpha\log\pi^i_{\text{new}} - \beta\sum_{j\neq i}\log\pi^j\right]\right].$$

- The operator $\mathcal{T}^{\pi'}Q(s_t, a_t) = r^i(s_t, a_t) + \gamma\mathbb{E}_{s_{t+1}\sim p}[\mathbb{E}_{a_{t+1}\sim\pi'}[Q(s_{t+1}, a_{t+1}) - \alpha\log\pi^i_{\text{new}} + \beta\sum_{j\neq i} CE]]$ contracts, so:

$$Q^{\pi}_i \leq Q^{\pi'}_i.$$