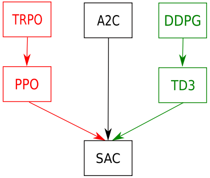
# From Policy Gradient to Actor-Critic methods

## Soft Actor Critic

Olivier Sigaud
with the help of Thomas Pierrot

Sorbonne Université
http://people.isir.upmc.fr/sigaud

## Soft Actor Critic: The best of two worlds




- TRPO and PPO: $\pi_\theta$ stochastic, on-policy, low sample efficiency, stable
- DDPG and TD3: $\pi_\theta$ deterministic, replay buffer, better sample efficiency, unstable
- SAC: "Soft" means "entropy regularized", $\pi_\theta$ stochastic, replay buffer
- Adds entropy regularization to favor exploration (follow-up of several papers)
- Attempt to be stable and sample efficient
- Three successive versions

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A. Abbeel, P. et al. (2018) Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018) Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*

Haarnoja, T. Tang, H., Abbeel, P. and Levine, S. (2017) Reinforcement learning with deep energy-based policies. *arXiv preprint arXiv:1702.08165*

## Soft Actor-Critic

SAC learns a **stochastic** policy $\pi^*$ maximizing both rewards and entropy:

$$\pi^* = \arg\max_{\pi_\theta} \sum_t \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t) \sim \rho_{\pi_\theta}} \left[ r(\mathbf{s}_t, \mathbf{a}_t) + \alpha \mathcal{H}(\pi_\theta(.|\mathbf{s}_t)) \right]$$

- The entropy is defined as: $\mathcal{H}(\pi_\theta(.|\mathbf{s}_t)) = \mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(.|\mathbf{s}_t)} \left[ -\log \left( \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) \right) \right]$
- SAC changes the traditional MDP objective
- Thus, it converges toward different solutions
- Consequently, it introduces a new value function, the soft value function
- As usual, we consider a policy $\pi_\theta$ and a soft Q-value function $\hat{Q}_\phi^{\pi_\theta}$

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. (2016) Asynchronous methods for deep reinforcement learning. *arXiv preprint arXiv:1602.01783*

## Soft policy evaluation

- Usually, we define $\hat{V}_{\phi}^{\pi_\theta}(\mathbf{s}_t) = \mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(.|\mathbf{s}_t)}\left[\hat{Q}_{\phi}^{\pi_\theta}(\mathbf{s}_t, \mathbf{a}_t)\right]$
- In soft updates, we rather use:

$$\begin{aligned}
\hat{V}_{\phi}^{\pi_\theta}(\mathbf{s}_t) &= \alpha \mathcal{H}(\pi_\theta(.|\mathbf{s}_t)) + \mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(.|\mathbf{s}_t)}\left[\hat{Q}_{\phi}^{\pi_\theta}(\mathbf{s}_t, \mathbf{a}_t)\right] \\
&= \mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(.|\mathbf{s}_t)}\left[-\log\left(\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)\right)\right] + \mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(.|\mathbf{s}_t)}\left[\hat{Q}_{\phi}^{\pi_\theta}(\mathbf{s}_t, \mathbf{a}_t)\right] \\
&= \mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(.|\mathbf{s}_t)}\left[\hat{Q}_{\phi}^{\pi_\theta}(\mathbf{s}_t, \mathbf{a}_t) - \alpha \log\left(\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)\right)\right]
\end{aligned}$$

## Critic updates

- We define a standard Bellman operator:

$$\mathcal{T}^{\pi} \hat{Q}_{\phi}^{\pi_\theta}(\mathbf{s}_t, \mathbf{a}_t) = r(\mathbf{s}_t, \mathbf{a}_t) + \gamma V(\mathbf{s}_{t+1})$$
$$= r(\mathbf{s}_t, \mathbf{a}_t) + \gamma \mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(.|\mathbf{s}_{t+1})} \left[ \hat{Q}_{\phi}^{\pi_\theta}(\mathbf{s}_{t+1}, \mathbf{a}_t) - \alpha \log \left( \pi_\theta(\mathbf{a}_t | \mathbf{s}_{t+1}) \right) \right]$$

Critic parameters can be learned by minimizing:

$$J_Q(\theta) = \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) \sim \mathcal{D}} \left[ \left( \hat{Q}_{\phi}^{\pi_\theta}(\mathbf{s}_t, \mathbf{a}_t) - \left( r(\mathbf{s}_t, \mathbf{a}_t) + \gamma \hat{V}_{\phi}^{\pi_\theta}(\mathbf{s}_{t+1}) \right) \right)^2 \right]$$

where $V_\theta(\mathbf{s}_{t+1}) = \mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(.|\mathbf{s}_{t+1})} \left[ \hat{Q}_{\phi}^{\pi_\theta}(\mathbf{s}_{t+1}, \mathbf{a}_t) - \alpha \log \left( \pi_\theta(\mathbf{a}_t | \mathbf{s}_{t+1}) \right) \right]$

- Similar to DDPG update, but with entropy

## Actor updates

- Update policy such as to become greedy w.r.t to the soft Q-value
- Choice: update the policy towards the exponential of the soft Q-value

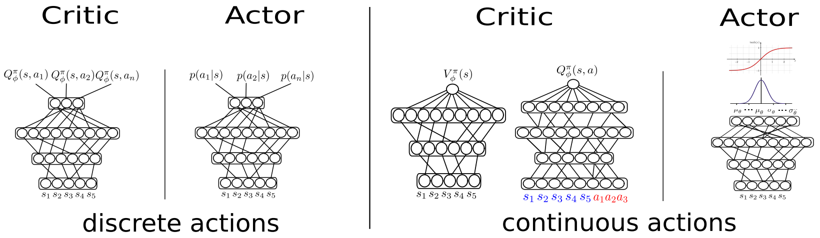
$$J_\pi(\theta) = \mathbb{E}_{\mathbf{s}_t \sim \mathcal{D}}[KL(\pi_\theta(.|\mathbf{s}_t))|| \frac{\exp(\frac{1}{\alpha}\hat{Q}_\phi^{\pi_\theta}(\mathbf{s}_t,.))}{Z_\theta(\mathbf{s}_t)}].$$

- $Z_\theta(\mathbf{s}_t)$ is just a normalizing term to have a distribution
- We do not minimize directly this expression but a surrogate one that has the same gradient w.r.t $\theta$

The policy parameters can be learned by minimizing:

$$J_\pi(\theta) = \mathbb{E}_{\mathbf{s}_t \sim \mathcal{D}}\left[\mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(.|\mathbf{s}_t)}\left[\alpha \log \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) - \hat{Q}_\phi^{\pi_\theta}(\mathbf{s}_t,\mathbf{a}_t)\right]\right]$$

## Continuous vs discrete actions setting




- SAC works in both the discrete action and the continuous action setting
- Discrete action settings:
  - The actor takes a state and returns probabilities over actions
  - The critic takes a state and returns one Q-value per action
- Continuous action settings:
  - The critic takes a state and an action vector and returns a scalar Q-value
  - Need to choose a distribution parameterisation for the actor
  - SAC uses a squashed Gaussian: $\mathbf{a} = \tanh(n)$ where $n \sim \mathcal{N}(\mu_\phi, \sigma_\phi)$

## Continuous vs discrete actions setting

SAC updates require to estimate an expectation over actions sampled from the actor, i.e. the computation of $\mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(.|s)}\left[F(\mathbf{s}_t, \mathbf{a}_t)\right]$ where $F$ is a scalar function.

- In the discrete action setting:
  - $\mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(.|\mathbf{s}_t)}\left[F(\mathbf{s}_t, \mathbf{a}_t)\right] = \pi_\theta(.|\mathbf{s}_t)^T F(.|\mathbf{s}_t)$
- In the continuous action setting:
  - The actor returns $\mu_\phi$ and $\sigma_\phi$
  - Re-parameterization trick: $\mathbf{a}_t = \tanh(\mu_\theta + \epsilon.\sigma_\theta)$ where $\epsilon \sim \mathcal{N}(0, 1)$
  - Thus, $\mathbb{E}_{\mathbf{a}_t \sim \pi_\phi(.|\mathbf{s}_t)}\left[F(\mathbf{s}_t, \mathbf{a}_t)\right] = \mathbb{E}_{\epsilon \sim \mathcal{N}(0,1)}\left[F(\tanh(\mu_\theta + \epsilon\sigma_\theta), \mathbf{s}_t)\right]$
  - This trick reduces the expectation estimate variance
  - And allows to backprop through the expectation w.r.t parameters $\theta$

## Critic update improvements (from TD3)

- As in TD3, use two critics $\hat{Q}^{\pi_\theta}_{\phi_1}$ and $\hat{Q}^{\pi_\theta}_{\phi_2}$
- The TD-target becomes:

$$y_t = r + \gamma \mathbb{E}_{\mathbf{a}_t \sim \pi(.|s_{t+1})} \left[ \min_{i=1,2} \hat{Q}^{\pi_\theta}_{\bar{\phi}_i}(s_{t+1}, \mathbf{a}_t) - \alpha \log\left(\pi(\mathbf{a}_t|s_{t+1})\right) \right]$$

And the losses:

$$\begin{cases} J(\theta) = \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t, s_{t+1}) \sim \mathcal{D}} \left[ \left( \hat{Q}^{\pi_\theta}_{\phi_1}(\mathbf{s}_t, \mathbf{a}_t) - y_t \right)^2 + \left( \hat{Q}^{\pi_\theta}_{\phi_2}(\mathbf{s}_t, \mathbf{a}_t) - y_t \right)^2 \right] \\ J(\theta) = \mathbb{E}_{s \sim \mathcal{D}} \left[ \mathbb{E}_{\mathbf{a}_t \sim \pi_\theta(.|\mathbf{s}_t)} \left[ \alpha \log \pi_\theta(\mathbf{a}_t|s_t) - \min_{i=1,2} \hat{Q}^{\pi_\theta}_{\bar{\phi}_i}(s_{t+1}, \mathbf{a}_t) \right] \right] \end{cases}$$

Fujimoto, S., van Hoof, H., & Meger, D. (2018) Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*

## Automatic Entropy Adjustment

- Choosing the optimal temperature $\alpha$ is non trivial
- $\alpha$ needs to be tuned for each task
- We can tune a lower bound $\bar{\mathcal{H}}$ for the policy entropy instead
- Use heuristic to compute $\bar{\mathcal{H}}$ from the action space size
- We change the optimization problem into a constrained one

$$\begin{cases} \pi^* = \underset{\pi}{\operatorname{argmax}} \sum_t \mathbb{E}_{(\mathbf{s}_t,\mathbf{a}_t)\sim\rho_{\pi_\theta}} \left[ r(\mathbf{s}_t, \mathbf{a}_t) \right] \\ \text{s.t. } \mathbb{E}_{(\mathbf{s}_t,\mathbf{a}_t)\sim\rho_{\pi_\theta}} \left[ -\log \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) \right] \geq \mathcal{H}_0, \quad \forall t \end{cases}$$

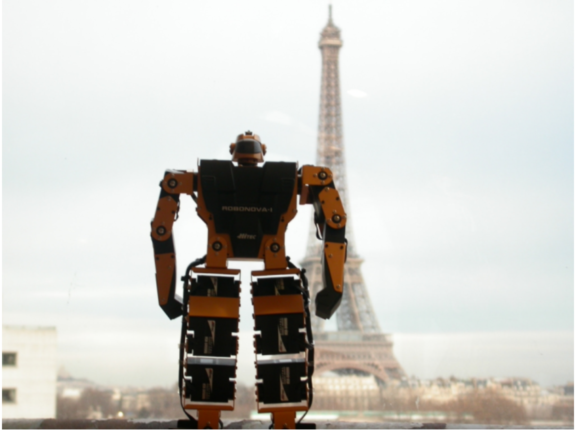
$\alpha$ can be learned to satisfy this constraint by minimizing:

$$J(\alpha) = \mathbb{E}_{\mathbf{s}_t\sim\mathcal{D}} \left[ \mathbb{E}_{\mathbf{a}_t\sim\pi_\theta(.|\mathbf{s}_t)} \left[ -\alpha \log \pi_\theta(\mathbf{a}_t|\mathbf{s}_t) - \alpha\mathcal{H}_0 \right] \right]$$

## Practical algorithm

- Initialize neural networks $\pi_\theta$ and $\hat{Q}_\phi^{\pi_\theta}$ weights
- Play $k$ steps in the environment by sampling actions with $\pi_\theta$
- Store the collected transitions in a replay buffer
- Sample $k$ batches of transitions in the replay buffer
- Update the temperature $\alpha$, the actor and the critic using SGD
- Repeat this cycle until convergence

## Any question?




Send mail to: Olivier.Sigaud@upmc.fr

Scott Fujimoto, Herke van Hoof, and Dave Meger.
Addressing function approximation error in actor-critic methods.
*arXiv preprint arXiv:1802.09477*, 2018.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine.
Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor.
*arXiv preprint arXiv:1801.01290*, 2018a.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al.
Soft actor-critic algorithms and applications.
*arXiv preprint arXiv:1812.05905*, 2018b.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu.
Asynchronous methods for deep reinforcement learning.
*arXiv preprint arXiv:1602.01783*, 2016.