A Walk in the Park: Learning to Walk in 20 Minutes With Model-Free Reinforcement Learning

He Zhi, CSE of SYSU

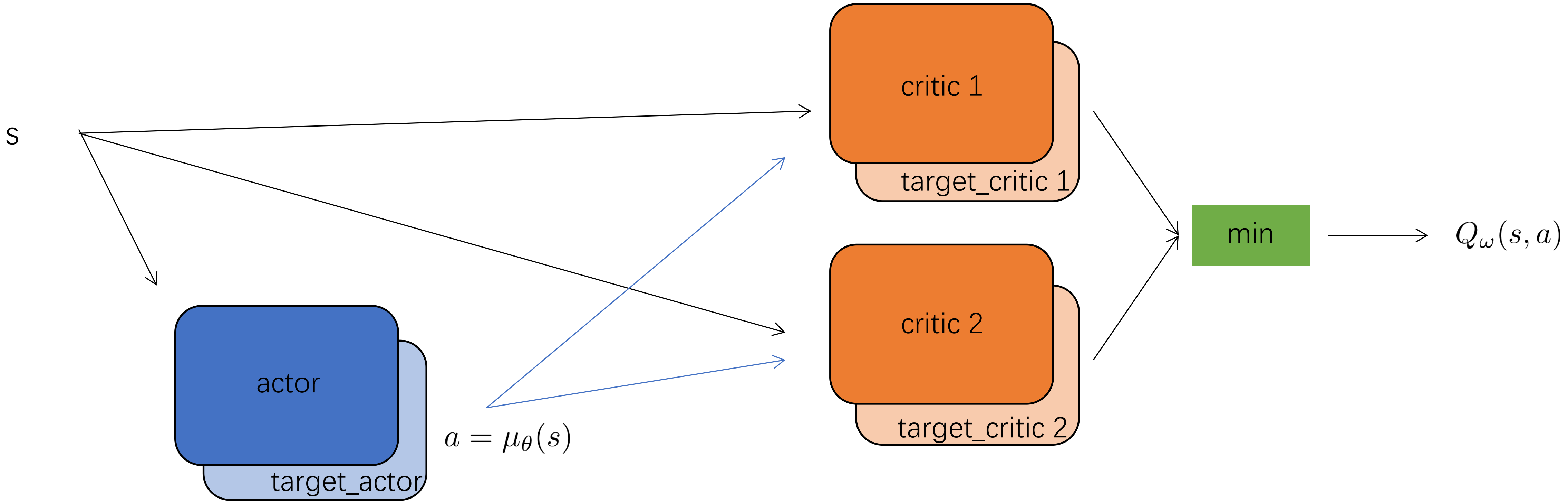
Email: hezh58@mail2.sysu.edu.cn

Learning inefficiency

Regularization/Normalization

Preliminaries:SAC (soft actor critic)

- A max-entropy DRL: $\pi = \arg\max_{\pi} \mathbb{E}[\sum_t r(s_t, a_t) + \alpha H(\pi(\Delta|s_t))]$



REDQ

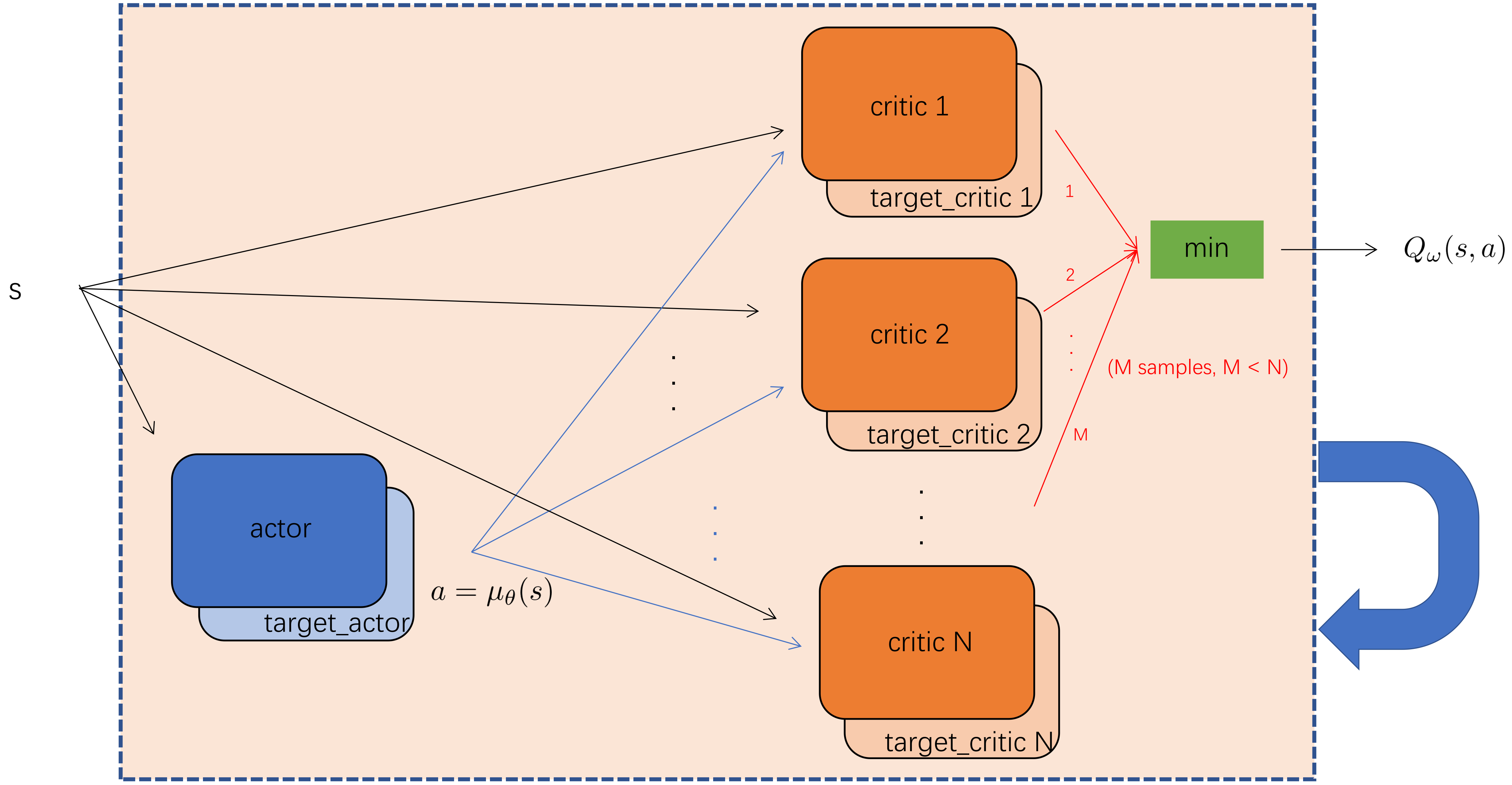
UTD: Update-To-Data ratio

$$= \frac{\#gradient \quad steps}{\#environment-interacting \quad steps}$$

- N: number of ensemble of Q networks
- M: number of radomized sampling Q networks to minimise
- G: number of updating epochs

《Randomized Ensembled Double Q-Learning: Learning Fast Without a Model》

REDQ


s
critic 1
target_critic 1
critic 2
target_critic 2
critic N
target_critic N
actor
target_actor
$a = \mu_\theta(s)$
1
2
M
(M samples, M < N)
min
$Q_\omega(s, a)$

REDQ

Algorithm 1 Randomized Ensembled Double Q-learning (REDQ)

Initialize policy parameters θ, N Q-function parameters ϕ_i, $i = 1, \ldots, N$, empty replay buffer $\mathcal{D}$. Set target parameters $\phi_{\text{targ},i} \leftarrow \phi_i$, for $i = 1, 2, \ldots, N$

repeat

  Take one action $a_t \sim \pi_\theta(\cdot|s_t)$. Observe reward r_t, new state s_{t+1}.

  Add data to buffer: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, a_t, r_t, s_{t+1})\}$

  for G updates **do**

    Sample a mini-batch $B = \{(s, a, r, s')\}$ from $\mathcal{D}$

    Sample a set $\mathcal{M}$ of M distinct indices from $\{1, 2, \ldots, N\}$

    Compute the Q target y (same for all of the N Q-functions):

$$y = r + \gamma \left(\min_{i \in \mathcal{M}} Q_{\phi_{\text{targ},i}}(s', \tilde{a}') - \alpha \log \pi_\theta(\tilde{a}' \mid s') \right), \quad \tilde{a}' \sim \pi_\theta(\cdot \mid s')$$

    for $i = 1, \ldots, N$ **do**

      Update ϕ_i with gradient descent using

$$\nabla_\phi \frac{1}{|B|} \sum_{(s,a,r,s') \in B} \left(Q_{\phi_i}(s, a) - y\right)^2$$

      Update target networks with $\phi_{\text{targ},i} \leftarrow \rho \phi_{\text{targ},i} + (1 - \rho)\phi_i$

  Update policy parameters θ with gradient ascent using

$$\nabla_\theta \frac{1}{|B|} \sum_{s \in B} \left(\frac{1}{N} \sum_{i=1}^{N} Q_{\phi_i}(s, \tilde{a}_\theta(s)) - \alpha \log \pi_\theta(\tilde{a}_\theta(s)|s) \right), \quad \tilde{a}_\theta(s) \sim \pi_\theta(\cdot \mid s)$$

- SAC： G = 1， N = M = 2

  UTD = 1

- REDQ: G > 1, N > M >> 2

  UTD = 20

《Randomized Ensembled Double Q-Learning: Learning Fast Without a Model》

DroQ

- small emsemble
- dropout Q-network

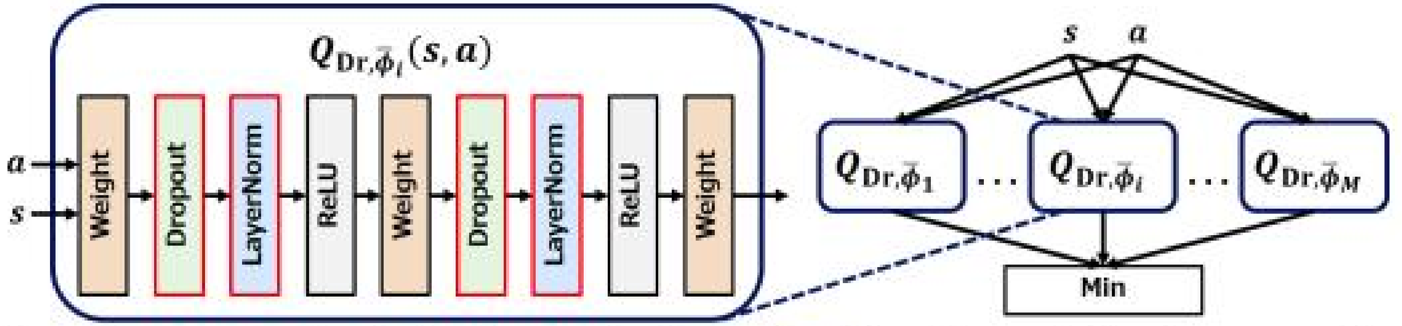


Figure 1: Dropout Q-function implementation (left part) and how dropout Q-functions are used in target (right part). **Dropout Q-function implementation:** Our dropout function is implemented by modifying that used by Chen et al. (2021b). Our modification (highlighted in red) is adding dropout (Dropout) and layer normalization (LayerNorm). "Weight" is a weight layer and "ReLU" is the activation layer of rectified linear units. Parameters $\bar{\phi}_i$ represent the weights and biases in weight layers. **How dropout Q-functions are used in target:** M dropout Q-functions are used to calculate the target value as $\min_{i=1,\ldots,M} Q_{\mathrm{Dr},\bar{\phi}_i}(s,a)$.

《DROPOUT Q-FUNCTIONS FOR DOUBLY EFFICIENT REINFORCEMENT LEARNING》

DroQ

Algorithm 2 DroQ

1: Initialize policy parameters θ, M Q-function parameters ϕ_i, $i = 1, \ldots, M$, and empty replay buffer $\mathcal{D}$. Set target parameters $\bar{\phi}_i \leftarrow \phi_i$, for $i = 1, \ldots, M$.

2: **repeat**

3: Take action $a_t \sim \pi_\theta(\cdot|s_t)$. Observe reward r_t, next state s_{t+1}; $\mathcal{D} \leftarrow \mathcal{D} \bigcup (s_t, a_t, r_t, s_{t+1})$.

4: **for** G updates **do**

5: Sample a mini-batch $\mathcal{B} = \{(s, a, r, s')\}$ from $\mathcal{D}$.

6: Compute the Q target y for the dropout Q-functions:

$$y = r + \gamma \left(\min_{i=1,\ldots,M} Q_{\mathrm{Dr},\bar{\phi}_i}(s', a') - \alpha \log \pi_\theta(a'|s') \right), \quad a' \sim \pi_\theta(\cdot|s')$$

7: **for** $i = 1, \ldots, M$ **do**

8: Update ϕ_i with gradient descent using

$$\nabla_\phi \frac{1}{|\mathcal{B}|} \sum_{(s,a,r,s') \in \mathcal{B}} \left(Q_{\mathrm{Dr},\phi_i}(s, a) - y \right)^2$$

9: Update target networks with $\bar{\phi}_i \leftarrow \rho \bar{\phi}_i + (1 - \rho) \phi_i$.

10: Update θ with gradient ascent using

$$\nabla_\theta \frac{1}{|\mathcal{B}|} \sum_{s \in \mathcal{B}} \left(\frac{1}{M} \sum_{i=1}^{M} Q_{\mathrm{Dr},\phi_i}(s, a) - \alpha \log \pi_\theta(a|s) \right), \quad a \sim \pi_\theta(\cdot|s)$$

《DROPOUT Q-FUNCTIONS FOR DOUBLY EFFICIENT REINFORCEMENT LEARNING》

Experiment

Fig. 4: Examples of learned gaits acquired on a variety of real-world terrains. Left to right: flat, solid ground covered in dense foam mats; a 5cm memory foam mattress; loose ground comprised of eucalyptus bark; a grassy lawn; a gently sloped hiking trail.

- SACs
- REDQ
- DroQ

Experiment

- **state**: root orientation,
  root angular velocity,
  root linear velocity,
  joint angles,
  joint velocities,
  binary foot contacts,
  previous action,
  …

- **action**: motor angles of joints for every leg

- **reward function**: $r(s,a) = r_v(s,a) - 0.1 v_{yaw}^2$

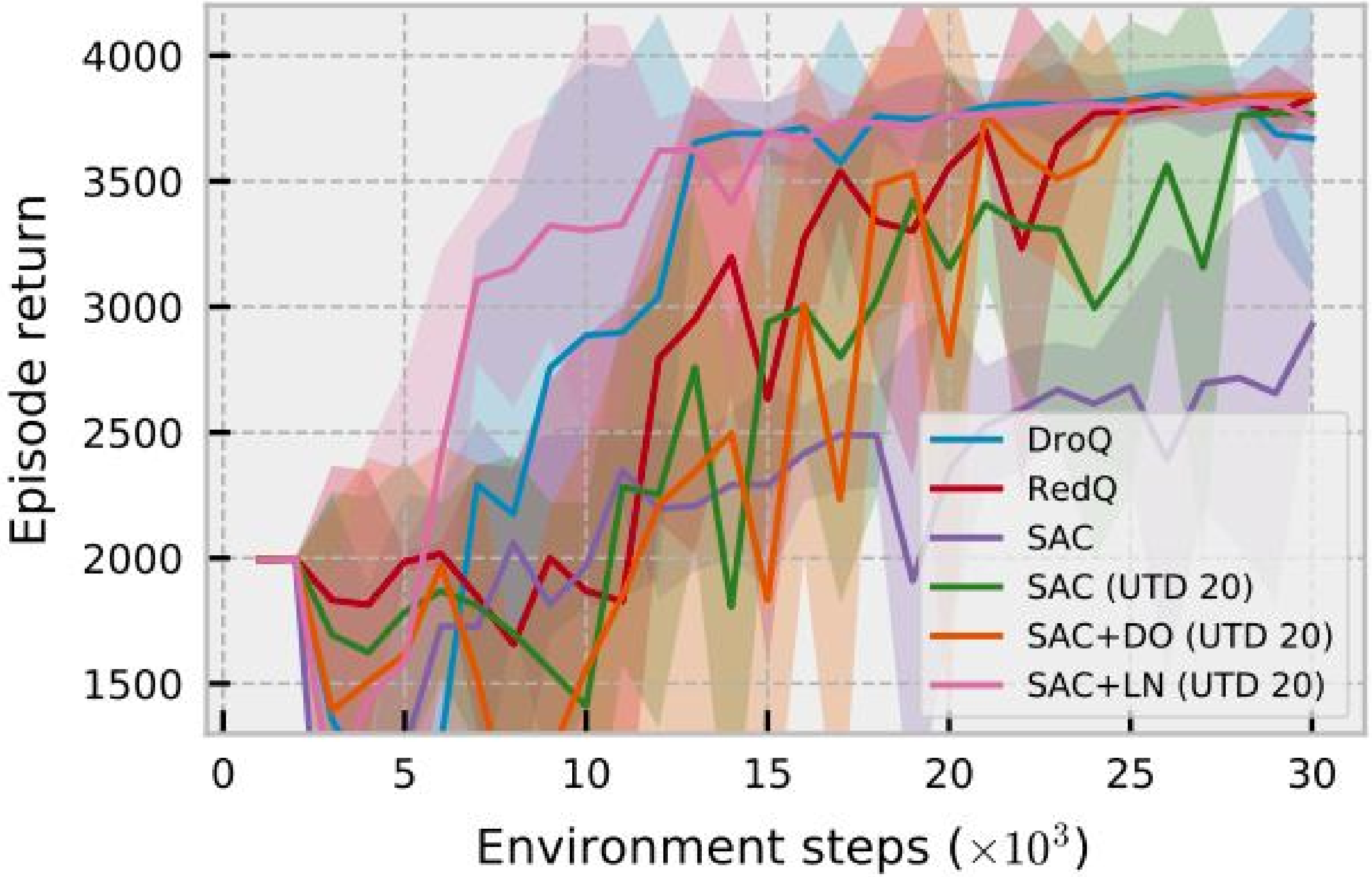
$$r_v(s,a) \begin{cases} 1 & \text{if } v_x \in [v_t, 2v_t] \\ 0 & \text{if } v_x \in (-\infty, -v_t] \cup [4v_t, \infty] \\ 1 - \frac{|v_x - v_t|}{2v_t} & \text{otherwise.} \end{cases}$$

v_{yaw} : angular yaw velocity

v_t : target velocity

Result

(c) SAC variants



Inspiration

- UTD > 1
- Dealing with bias of Q function learned

"REDQ has two critical components that allow it to maintain stable and near-uniform bias under high UTD ratios: an ensemble and in-target minimization."

from 《RANDOMIZED ENSEMBLED DOUBLE Q-LEARNING: LEARNING FAST WITHOUT A MODEL》