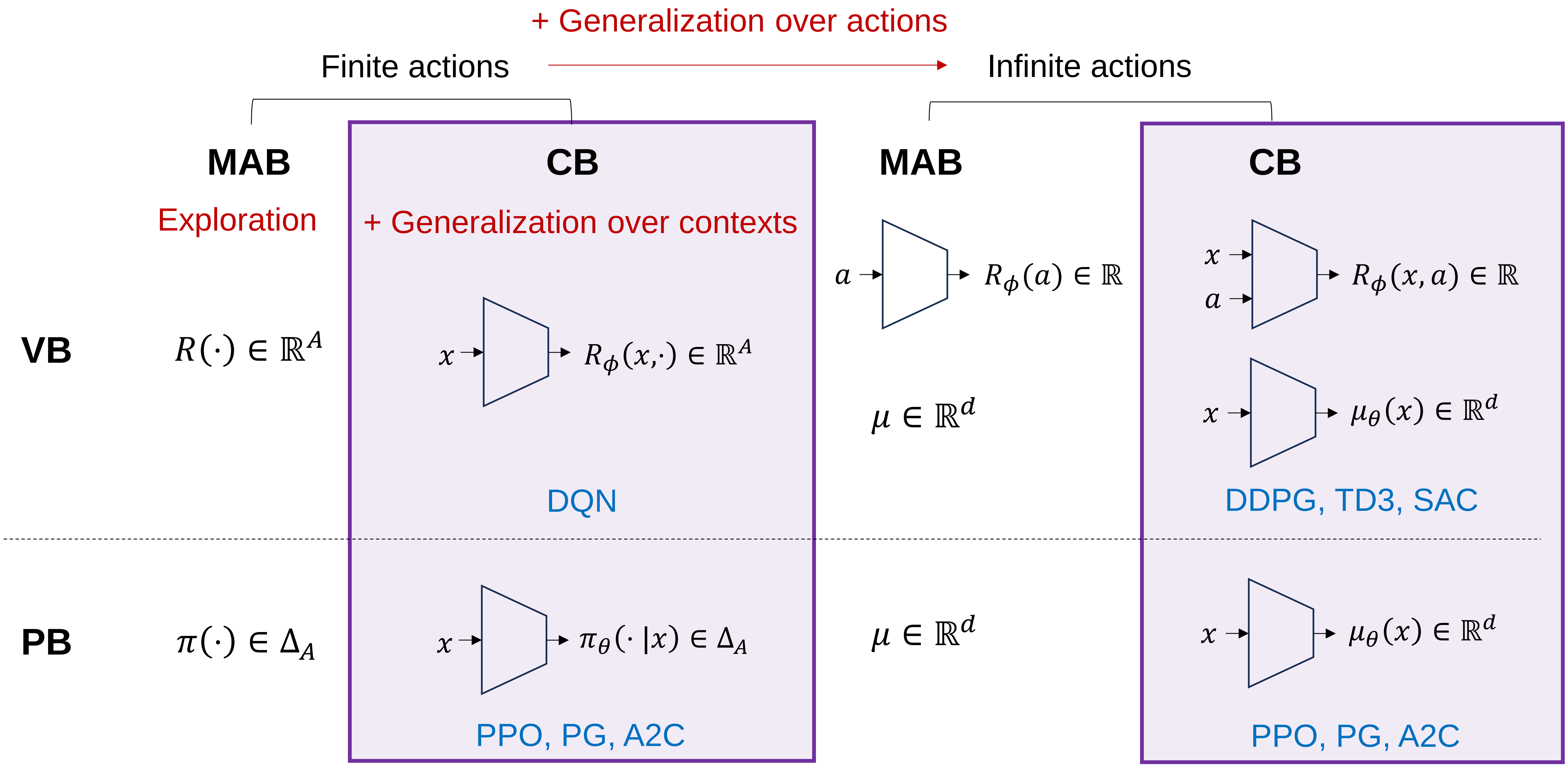
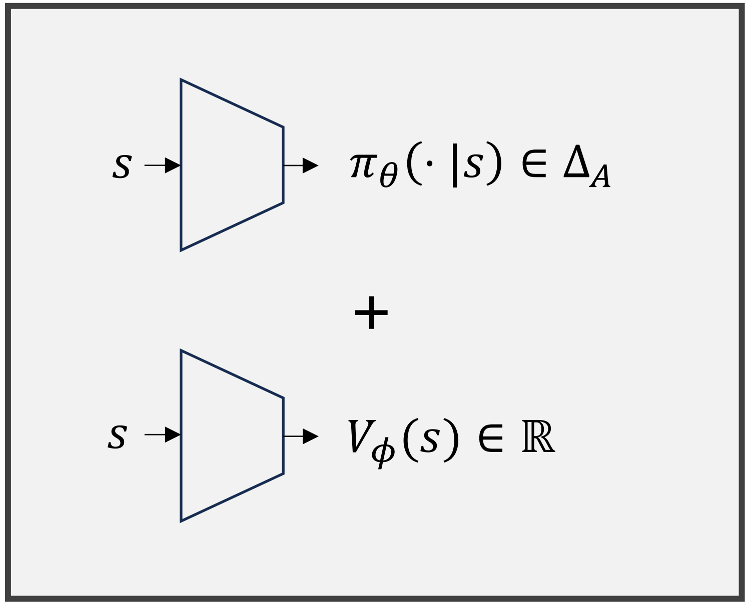
RL with Continuous Action Sets

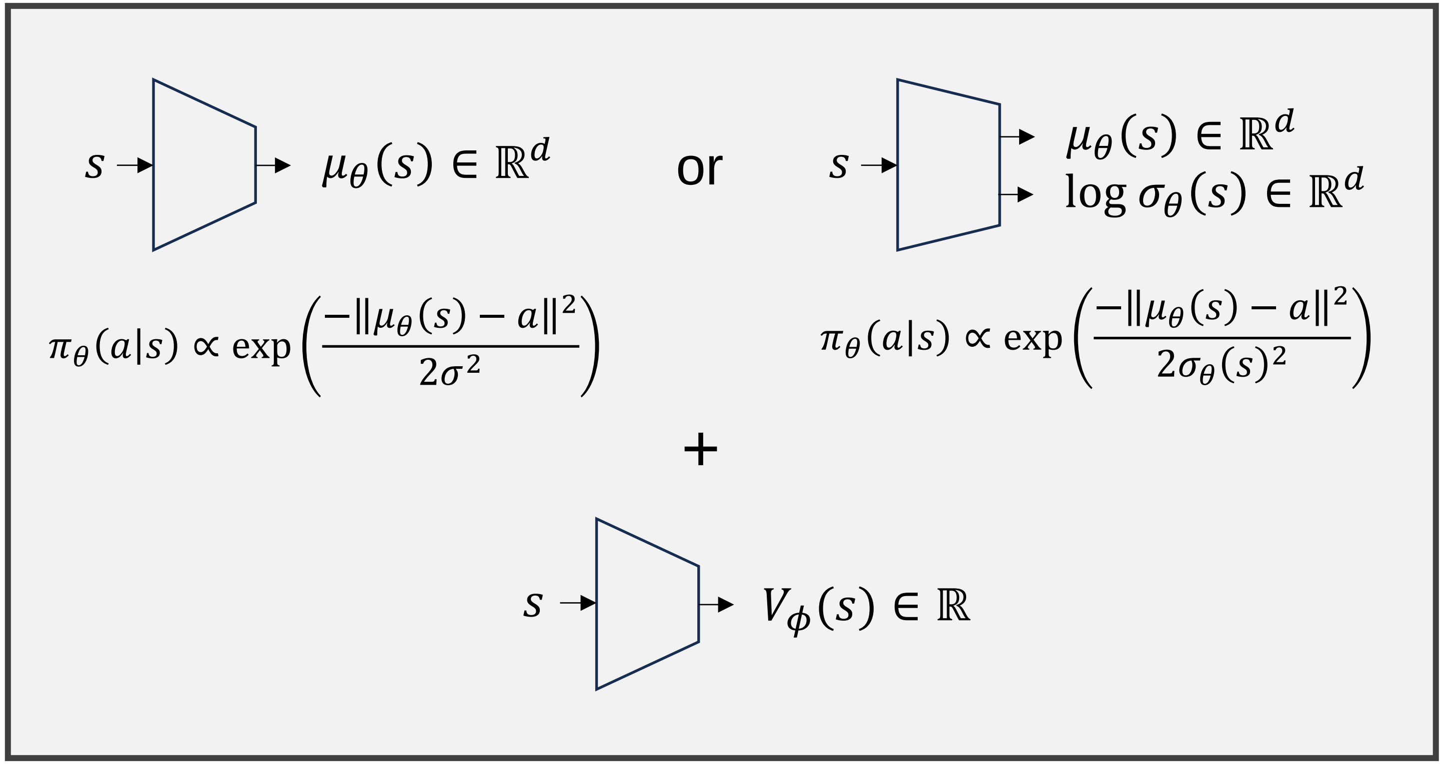
3 main challenges in online RL: Exploration, Generalization, (Temporal) Credit Assignment



PPO / PG / A2C in Discrete / Continuous Action Sets



Discrete actions



Continuous actions

Algorithms involving a policy and value network where the value is used in the policy update are called **actor-critic** algorithms.

PPO / PG / A2C in Continuous Action Sets

$$\theta_{k+1} \leftarrow \underset{\theta}{\operatorname{argmax}} \left\{ \sum_{i=1}^{N} \left(\frac{\pi_\theta(a_i|s_i)}{\pi_{\theta_k}(a_i|s_i)} A_i - \frac{1}{\eta} \mathrm{KL}\left(\pi_\theta(\cdot\,|s_i), \pi_{\theta_k}(\cdot\,|s_i)\right) \right) \right\}$$

$$\theta_{k+1} \leftarrow \theta_k + \eta\ \frac{1}{N} \sum_{i=1}^{N} \nabla_\theta \log \pi_\theta(a_i|s_i) \Big|_{\theta=\theta_k} A_i$$

where A_i is a weighted average of the following (GAE):

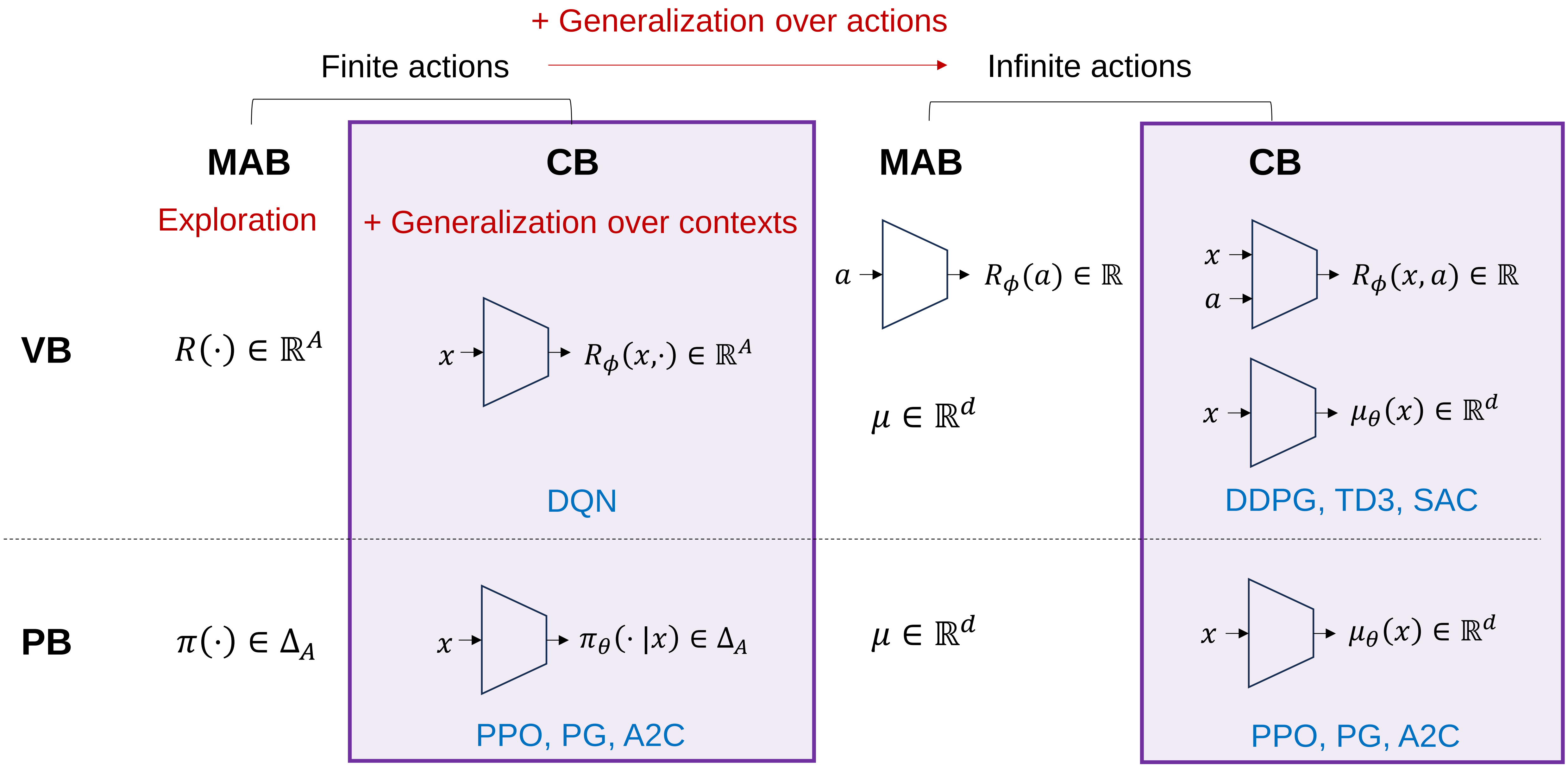
$r_i + \gamma V_\phi(s_{i+1}) - V_\phi(s_i)$

$r_i + \gamma r_{i+1} + \gamma^2 V_\phi(s_{i+2}) - V_\phi(s_i)$

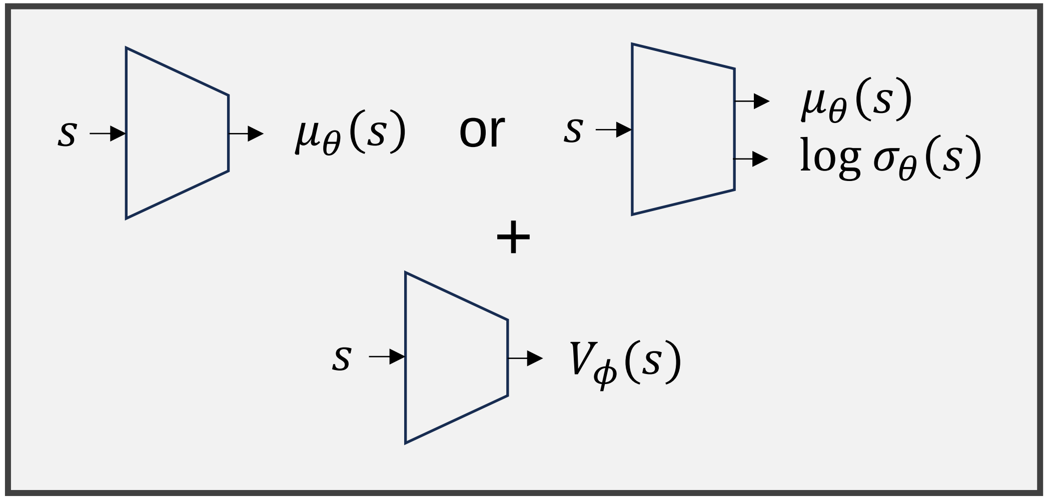
$r_i + \gamma r_{i+1} + \gamma^2 r_{i+2} + \gamma^3 V_\phi(s_{i+3}) - V_\phi(s_i)$

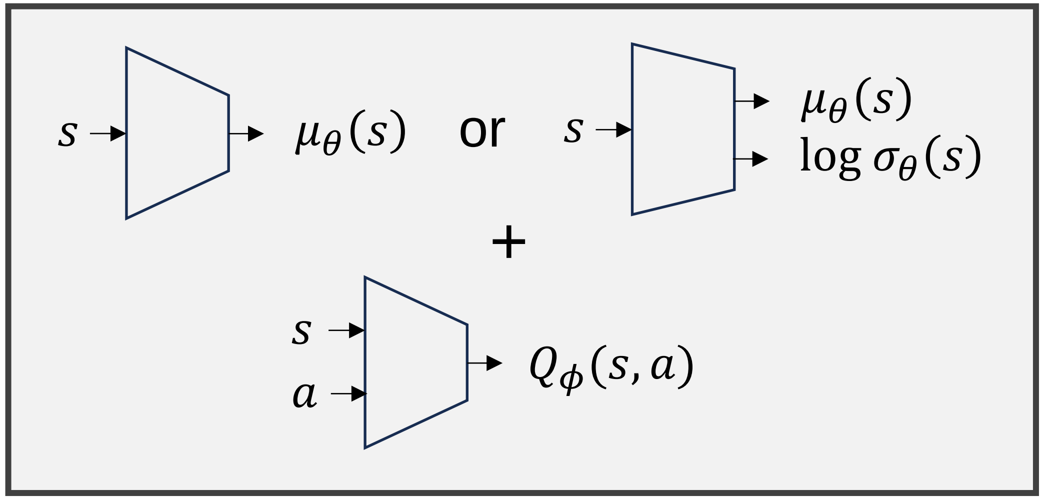
...

3 main challenges in online RL: Exploration, Generalization, (Temporal) Credit Assignment



Two Types of Actor-Critic Algorithms





	PPO / PG / A2C	DDPG / TD3 / SAC
Update θ with	$\frac{\pi_\theta(a_i \mid s_i)}{\pi_{\theta_k}(a_i \mid s_i)}\left(r_i + \gamma V_\phi(s_{i+1}) - V_\phi(s_i)\right)$	$Q_\phi(s_i, a_i)$
Idea more aligned with	Policy-based bandits (forming unbiased reward estimator)	Value-based bandits (forming reward estimator from regression)
	Policy Iteration (policy improvement based on $Q^\pi(s,a)$)	Policy Iteration or Value Iteration (policy improvement based on $Q^\star(s,a)$) – e.g. DQN
Training type	On-policy	On-policy or off-policy (using data collected from previous policies)

Deep Deterministic Policy Gradient (DDPG)

For $k = 1,\ 2, \ldots$

Use $\mu_\theta(s) + \mathcal{N}(0, \sigma^2)$ to collect samples and place them in **replay buffer**

Sample a batch $\{(s_i, a_i, r_i, s_i')\}_{i=1}^{n}$ from the replay buffer

$$\phi \leftarrow \phi - \lambda \nabla_\phi \sum_{i=1}^{n} \left(Q_\phi(s_i, a_i) - r_i - \gamma Q_{\bar{\phi}}\left(s_i', \mu_{\bar{\theta}}(s_i{}')\right) \right)^2$$

$$\theta \leftarrow \theta + \eta \sum_{i=1}^{n} \nabla_\theta Q_\phi\left(s_i, \mu_\theta(s_i)\right)$$

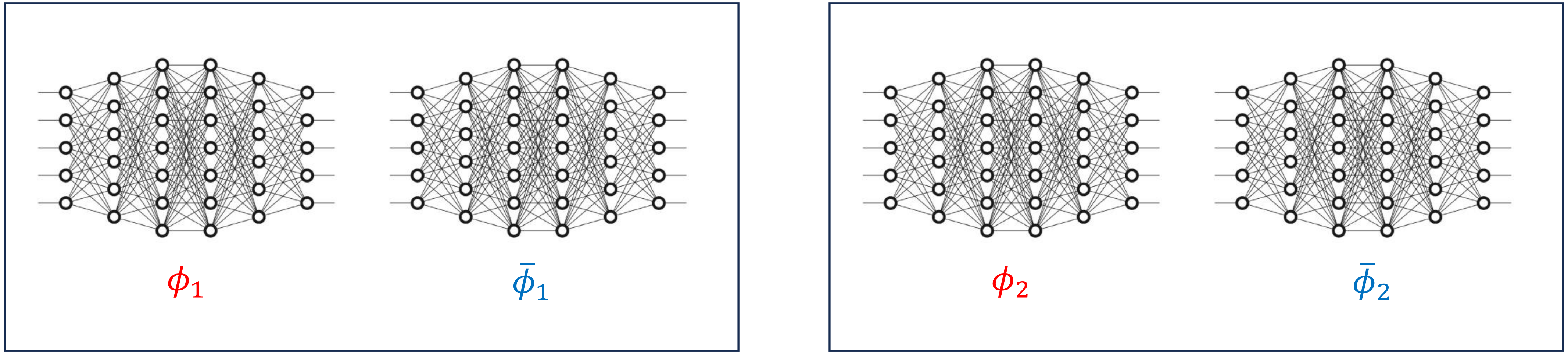
$$\bar{\phi} \leftarrow \tau\phi + (1-\tau)\bar{\phi}, \qquad \bar{\theta} \leftarrow \tau\theta + (1-\tau)\bar{\theta}$$

The bandit version of this algorithm: Page 11 here

Lillicrap et al., Continuous control with deep reinforcement learning. 2015.

Further Stabilizing DDPG (1/3): Twin Delayed DDPG

- Double Q-learning



Double Q-learning: When training ϕ_1, instead of using $Q_{\bar{\phi}_1}$ to evaluate the regression target, use $\cancel{Q_{\bar{\phi}_2}}$

TD3: $\min\left\{Q_{\bar{\phi}_1}, Q_{\bar{\phi}_2}\right\}$

Double Q-learning: Use independent samples to train ϕ_1 and ϕ_2

TD3: Use the same set of samples
(the independence between ϕ_1 and ϕ_2 only comes from random initialization)

Further Stabilizing DDPG (2/3): Twin Delayed DDPG

- Target policy smoothing

DDPG: use $Q_{\bar{\phi}}(s', \mu_{\bar{\theta}}(s'))$ as the regression target

TD3: sample $a' = \mu_{\bar{\theta}}(s') + \mathcal{N}(0, \sigma^2)$

use $Q_{\bar{\phi}}(s', a')$ as the regression target

Further Stabilizing DDPG (3/3): Twin Delayed DDPG

- Delayed policy updates: running multiple steps of value updates before running one step of policy update

Remark: all three changes make it harder for the policy $\mu_\theta(s)$ to exploit the error of the Q function $Q_\phi(s, a)$

Twin Delayed DDPG (TD3)

For $k = 1,\ 2, \ldots$

Use $\mu_\theta(s) + \mathcal{N}(0, \sigma^2)$ to collect samples and place them in replay buffer

Sample a batch $\{(s_i, a_i, r_i, s_i')\}_{i=1}^{n}$ from the replay buffer

For each sample i, draw $a_i' \sim \mu_{\bar{\theta}}(s_i{}') + \mathcal{N}(0, \sigma^2 I)$

$$\phi_j \leftarrow \phi_j - \lambda \nabla_{\phi_j} \sum_{i=1}^{n} \left(Q_{\phi_j}(s_i, a_i) - r_i - \gamma \min_{\ell=1,2} Q_{\bar{\phi}_\ell}(s_i', a_i') \right)^2 \qquad \forall j = 1,2$$

If $k \bmod M = 0$:

$$\theta \leftarrow \theta + \eta \sum_{i=1}^{n} \nabla_\theta Q_\phi\big(s_i, \mu_\theta(s_i)\big)$$

$$\bar{\theta} \leftarrow \tau\theta + (1 - \tau)\bar{\theta}$$

$$\bar{\phi}_j \leftarrow \tau\phi_j + (1 - \tau)\bar{\phi}_j \qquad \forall j = 1,2$$

Fujimoto et al., Addressing Function Approximation Error in Actor-Critic Methods. 2018.

Soft Actor-Critic (SAC)

- TD3 / DDPG: modeling $\mu_\theta(s)$ + additional noise for exploration
- SAC: modeling $\mu_\theta(s)$ and $\sigma_\theta(s)$ + adding entropy as an exploration bonus

Entropy Bonus (≈ Boltzmann Exploration)

Bandit

$$\pi = \underset{\pi}{\operatorname{argmax}} \sum_a \pi(a) R(a) + \alpha\, H(\pi) = \underset{\pi}{\operatorname{argmax}}\, \mathbb{E}_{a\sim\pi}[R(a) - \alpha \log \pi(a)]$$

MDP

$$\pi = \underset{\pi}{\operatorname{argmax}}\ \mathbb{E}^{\pi}\left[\sum_{h=0}^{\infty} \gamma^h \left(\sum_a \pi(a|s_h) R(s_h, a) + \alpha\, H(\pi(\cdot\,|s_h))\right)\right]$$

$$= \underset{\pi}{\operatorname{argmax}}\ \mathbb{E}^{\pi}\left[\sum_{h=0}^{\infty} \gamma^h\ (R(s_h, a_h) - \alpha \log \pi(a_h|s_h))\right]$$

Bellman Equation with Entropy Bonus

TD3 vs. SAC

- Value update

TD3: Sample $a' \sim \mu_\theta(s') + \mathcal{N}(0, \sigma^2)$

Use $Q_{\bar{\phi}}(s', a')$ as the regression target

SAC: Sample $a' \sim \pi_\theta(\cdot\,|s') = \mu_\theta(s') + \mathcal{N}(0, \sigma_\theta^2(s'))$

Use $Q_{\bar{\phi}}(s', a') - \alpha \log \pi_\theta\,(a'|s')$ as the regression target

Soft Actor-Critic (SAC)

For $k = 1,\ 2, \ldots$

Use $\mu_\theta(s) + \mathcal{N}(0, \sigma_\theta^2(s))$ to collect samples and place them in replay buffer

Sample a batch $\{(s_i, a_i, r_i, s_i')\}_{i=1}^n$ from the replay buffer

For each sample i, draw $a_i' \sim \mu_\theta(s_i{}') + \mathcal{N}(0, \sigma_\theta^2(s_i{}'))$

$$\phi_j \leftarrow \phi_j - \lambda \nabla_{\phi_j} \sum_{i=1}^{n} \left(Q_{\phi_j}(s_i, a_i) - r_i - \gamma \left(\min_{\ell=1,2} Q_{\bar{\phi}_\ell}(s_i', a_i') + \alpha \log \pi_\theta(a_i' | s_i{}') \right) \right)^2 \quad \forall j = 1,2$$

Perform Policy (θ) Update (to be specified later)

$$\bar{\phi}_j \leftarrow \tau \phi_j + (1 - \tau) \bar{\phi}_j \quad \forall j = 1,2$$

Haarnoja et al., Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. 2018.

TD3 vs. SAC

- Policy update

TD3: Do not view $-\alpha \log \pi_\theta\,(a|s)$ as part of the reward

Only train $\mu_\theta(s)$

$$\theta \leftarrow \theta + \eta \nabla_\theta Q_\phi(s, \mu_\theta(s))$$

SAC: View $-\alpha \log \pi_\theta\,(a|s)$ as part of the reward

Train both $\mu_\theta(s)$ and $\log \sigma_\theta(s)$

Sample $a_\theta(s) = \mu_\theta(s) + \epsilon\sigma_\theta(s)$ where $\epsilon \sim \mathcal{N}(0,1)$

$$\theta \leftarrow \theta + \eta \nabla_\theta \big(Q_\phi(s, a_\theta(s)) - \alpha \log \pi_\theta(a_\theta(s)|s)\big)$$

Soft Actor-Critic (SAC)

Further using $\pi_\theta(a|s) = \frac{1}{(2\pi\sigma_\theta(s)^2)^{d/2}} \exp\left(-\frac{\|a-\mu_\theta(s)\|^2}{\sigma_\theta(s)^2}\right)$

For $k = 1,\ 2, \ldots$

Use $\mu_\theta(s) + \mathcal{N}(0, \sigma_\theta^2(s))$ to collect samples and place them in replay buffer

Sample a batch $\{(s_i, a_i, r_i, s_i')\}_{i=1}^n$ from the replay buffer

For each sample i, draw $a_i' \sim \mu_\theta(s_i') + \mathcal{N}(0, \sigma_\theta^2(s_i'))$

$$\phi_j \leftarrow \phi_j - \lambda \nabla_{\phi_j} \sum_{i=1}^{n} \left(Q_{\phi_j}(s_i, a_i) - r_i - \gamma \left(\min_{\ell=1,2} Q_{\bar{\phi}_\ell}(s_i', a_i') + \alpha \log \pi_\theta(a_i' | s_i') \right) \right)^2 \quad \forall j = 1,2$$

Let $a_\theta(s_i) = \mu_\theta(s_i) + \epsilon \sigma_\theta(s_i)$ where $\epsilon \sim \mathcal{N}(0, I)$

$$\theta \leftarrow \theta + \eta \sum_{i=1}^{n} \nabla_\theta \left(Q_\phi(s, a_\theta(s_i)) - \alpha \log \pi_\theta(a_\theta(s_i) | s_i) \right)$$

$$\bar{\phi}_j \leftarrow \tau \phi_j + (1 - \tau) \bar{\phi}_j \quad \forall j = 1,2$$

Haarnoja et al., Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. 2018.