Lecture 4b: Actor Critic
CS885 Reinforcement Learning

2025-01-16

Complementary readings: [SutBar] Sec. 13.4-13.5, [Sze] Sec. 4.4, [SigBuf] Sec. 5.3

Pascal Poupart
David R. Cheriton School of Computer Science

UNIVERSITY OF WATERLOO

Outline

- Policy gradient with a baseline
- Actor Critic algorithms
- Deterministic policy gradient

UNIVERSITY OF WATERLOO

Actor Critic

- Q-learning
  - **Model-free value-based method**
  - No explicit policy representation
- Policy gradient
  - **Model-free policy-based method**
  - No explicit value function representation
- Actor Critic
  - **Model-free policy and value-based method**

Stochastic Gradient Policy Theorem

- Stochastic Gradient Policy Theorem

$$\nabla_\theta V_\theta(s_0) \propto \sum_s \mu_\theta(s) \sum_a \nabla_\theta\, \pi_\theta(a|s)\, Q_\theta(s,a)$$

- Equivalent Stochastic Gradient Policy Theorem with a baseline $b(s)$

$$\nabla_\theta V_\theta(s_0) \propto \sum_s \mu_\theta(s) \sum_a \nabla_\theta\, \pi_\theta(a|s)\, [Q_\theta(s,a) - b(s)]$$

since $\sum_a \nabla_\theta\, \pi_\theta(a|s)\, b(s) = b(s)\nabla_\theta \sum_a \pi_\theta(a|s) = b(s)\nabla_\theta 1 = 0$

Baseline

- Baseline often chosen to be $b(s) \approx V^{\pi}(s)$

Advantage function: $A(s,a) = Q(s,a) - V^{\pi}(s)$

Gradient update: $\theta \leftarrow \theta + \alpha\, \gamma^{n} A(s_n, a_n) \nabla_{\theta} \log \pi_{\theta}(a_n|s_n)$

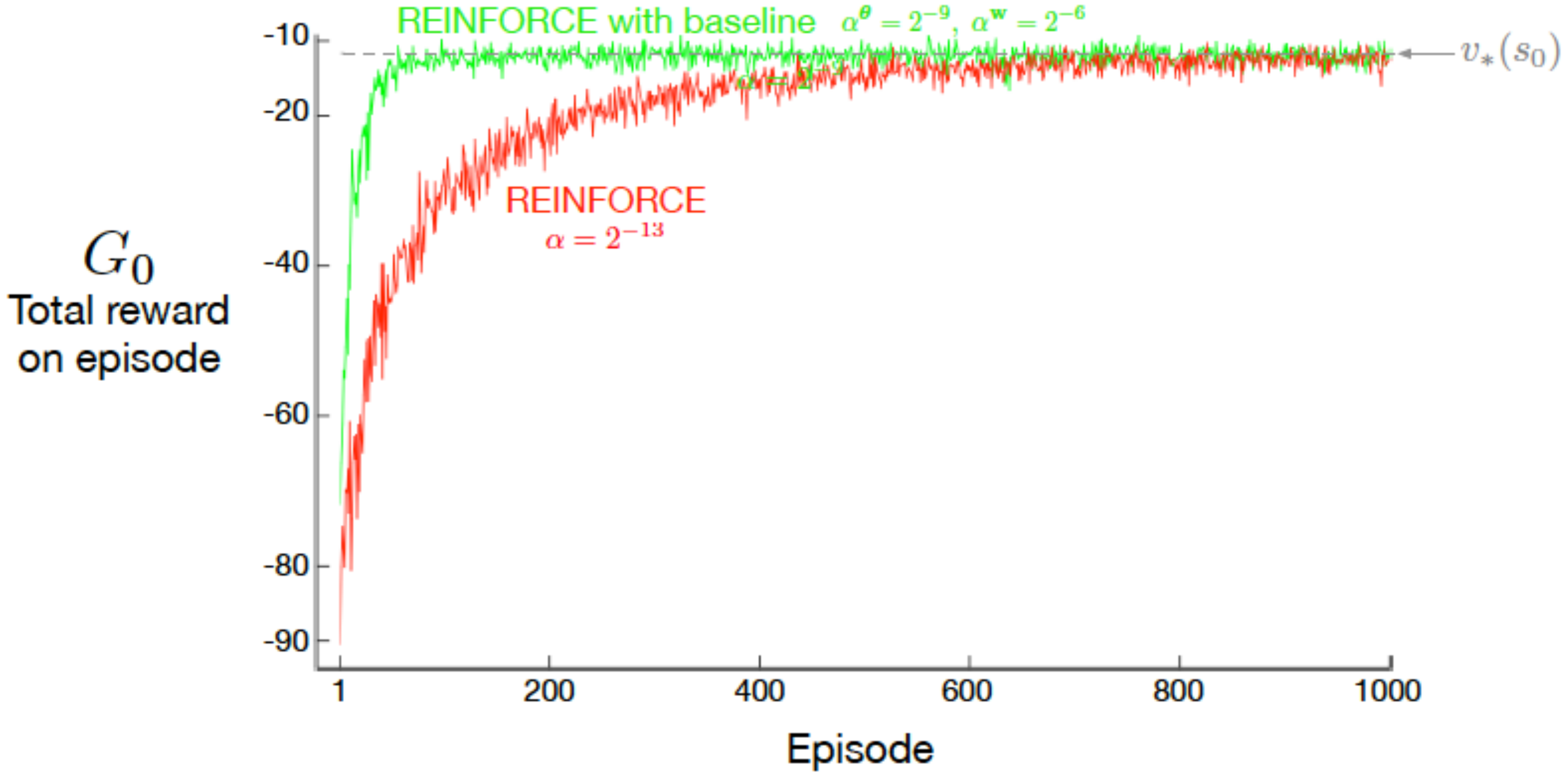
Benefit: faster empirical convergence

REINFORCE Algorithm with a Baseline

REINFORCEwithBaseline(s_0)
  Initialize π_θ to anything
  Initialize V_w to anything
  Loop forever (for each episode)
    Generate episode $s_0, a_0, r_0, s_1, a_1, r_1, \dots, s_T, a_T, r_T$ with π_θ
    Loop for each step of the episode $n = 0, 1, \dots, T$
      $G_n \leftarrow \sum_{t=0}^{T-n} \gamma^t \, r_{n+t}$
      $\delta \leftarrow G_n - V_w(s_n)$
      Update value function: $w \leftarrow w + \alpha_w \, \delta \nabla_w V_w(s_n)$
      Update policy: $\theta \leftarrow \theta + \alpha_\theta \gamma^n \delta \nabla_\theta \log \pi_\theta(a_n|s_n)$
  Return π_θ

UNIVERSITY OF WATERLOO

Performance Comparison



UNIVERSITY OF WATERLOO

Temporal Difference Update

- Instead of updating $V(s)$ by Monte Carlo sampling

$$\delta \leftarrow G_n - V_w(s_n)$$

Bootstrap with temporal difference updates

$$\delta \leftarrow r_n + \gamma V_w(s_{n+1}) - V_w(s_n)$$

- Benefit: reduced variance (faster convergence)

Actor Critic Algorithm

ActorCritic(s_0)

  Initialize π_θ, Q_w to anything

  Loop forever (for each episode)

    Initialize s_0 and set $n \leftarrow 0$

    Loop while s is not terminal (for each time step n)

      Sample $a_n \sim \pi_\theta(a|s_n)$

      Execute a_n, observe s_{n+1}, r_n

      $\delta \leftarrow r_n + \gamma V_w(s_{n+1}) - V_w(s_n)$

      Update value function: $w \leftarrow w + \alpha_w \, \delta \nabla_w V_w(s_n)$

      Update policy: $\theta \leftarrow \theta + \alpha_\theta \gamma^n \delta \nabla_\theta \log \pi_\theta(a_n|s_n)$

      $n \leftarrow n + 1$

  Return π_θ

Advantage Update

- Instead of doing temporal difference updates

$$\delta \leftarrow r_n + \gamma V_w(s_{n+1}) - V_w(s_n)$$

- Update with the advantage function

$$A(s_n, a_n) \leftarrow r_n + \gamma \max_{a_{n+1}} Q(s_{n+1}, a_{n+1}) - \sum_a \pi_\theta(a|s_n) Q(s_n, a)$$

$$\theta \leftarrow \theta + \alpha_\theta \gamma^n A(s_n, a_n) \nabla_\theta \log \pi_\theta(a_n|s_n)$$

- Benefit: faster convergence

Advantage Actor Critic (A2C)

A2C(s_0)

Initialize π_θ, Q_w to anything

Loop forever (for each episode)

Initialize s_0 and set $n \leftarrow 0$

Loop while s is not terminal (for each time step n)

Sample $a_n \sim \pi_\theta(a|s_n)$, execute a_n, observe s_{n+1}, r_n

$\delta \leftarrow r_n + \gamma \max_{a_{n+1}} Q_w(s_{n+1}, a_{n+1}) - Q_w(s_n, a_n)$

$A(s_n, a_n) \leftarrow r_n + \gamma \max_{a_{n+1}} Q_w(s_{n+1}, a_{n+1}) - \sum_a \pi_\theta(a|s_n) Q_w(s_n, a)$

Update Q_w: $w \leftarrow w + \alpha_w \delta \nabla_w Q_w(s_n, a_n)$

Update π_θ: $\theta \leftarrow \theta + \alpha_\theta \gamma^n A(s_n, a_n) \nabla_\theta \log \pi_\theta(a_n|s_n)$

$n \leftarrow n + 1$

Return π_θ

Continuous Actions

- Consider a deterministic policy $\pi_\theta(s) \to a$

- Deterministic Gradient Policy Theorem

$$\nabla V_\theta(s_0) \propto E_{s\sim\mu_\theta(s)}\left[\nabla_\theta \pi_\theta(s) \nabla_a Q_\theta(s,a)\Big|_{a=\pi_\theta(s)}\right]$$

Proof: see Silver et al. 2014

- Stochastic Gradient Policy Theorem

$$\nabla V_\theta(s_0) \propto \sum_s \mu_\theta(s) \sum_a \nabla_\theta\, \pi_\theta(a|s)\, Q_\theta(s,a)$$

Deep Deterministic Policy Gradient (DDPG)

Initialize π_θ, $\pi_{\bar{\theta}}$, Q_w, $Q_{\bar{w}}$ to anything
Loop forever (for each episode)
  Initialize s_0 and set $n \leftarrow 0$
  Loop while s is not terminal (for each time step n)
    Select a_n, execute a_n, observe r_n, s_{n+1}
    Add (s_n, a_n, r_n, s_{n+1}) to experience buffer
    Sample mini-batch of experiences from buffer
    For each experience $(\hat{s}_{\hat{n}}, \hat{a}_{\hat{n}}, \hat{r}_{\hat{n}}, \hat{s}_{\hat{n}+1})$ in mini-batch
      $\delta \leftarrow \hat{r}_{\hat{n}} + \gamma Q_{\bar{w}}\left(\hat{s}_{\hat{n}+1}, \pi_{\bar{\theta}}(\hat{s}_{\hat{n}+1})\right) - Q_w(\hat{s}_{\hat{n}}, \hat{a}_{\hat{n}})$
      Update Q_w: $w \leftarrow w + \alpha_w \delta \nabla_w Q_w(\hat{s}_{\hat{n}}, \hat{a}_{\hat{n}})$
      Update π_θ: $\theta \leftarrow \theta + \alpha_\theta \gamma^{\hat{n}} \nabla_\theta \pi_\theta(\hat{s}_{\hat{n}}) \nabla_a Q_w(\hat{s}_{\hat{n}}, a)|_{a=\pi_\theta(\hat{s}_{\hat{n}})}$
    $n \leftarrow n + 1$
    Update target networks: every c steps $\bar{w} \leftarrow w$, $\bar{\theta} \leftarrow \theta$
Return π_θ

Comparison

	A2C	DDPG
Policy	Stochastic	Deterministic
Learning mode	On policy ($a \sim \pi(a\|s)$)	Off policy (select a according to any policy)
Experience buffer	No	Yes (greater data efficiency)
Target networks	No	Yes (greater stability)

DDPG in Robotics

Lillicrap, Hunt, Pritzel, Heess, Erez, Tassa, Silver, Wierstra (2016) **Continuous Control with Deep Reinforcement Learning**, *ICLR*.