Multi-Agent Systems

Introduction to Reinforcement Learning

Multi-Agent RL

Eric Pauwels (CWI & VU)

December 6, 2021

Outline

Advantage Actor-Critic (A2C)

Actor-Critic combines valued-based and policy-based learning.

Actor-Critic algorithms therefore have two components that are learned jointly:

- Actor learns a parametrised policy
- Critic learns value function to evaluate state-action pairs;

Advantage function: Select action based on how it performs relative to other actions in that state:

$$a_\pi(s, a) := q_\pi(s, a) - v_\pi(s)$$

Quick Recap

Policy gradient along trajectory τ

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} R_t(\tau) \nabla_\theta \log \pi_\theta(a_t \mid s_t) \right]$$

Restricting the trajectory to the part starting at s_t:

$$R_t(\tau) = R(s_t, a_t, \ldots, s_T) \quad \Longrightarrow \quad \mathbb{E}_{\tau \sim \pi_\theta} [R_t(\tau)] = q_{\pi_\theta}(s_t, a_t);$$

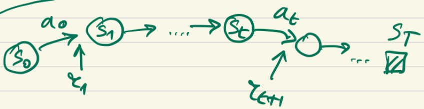
$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} q_{\pi_\theta}(s_t, a_t) \nabla_\theta \log \pi_\theta(a_t \mid s_t) \right]$$

Policy Gradient

$$\nabla_\theta J(\theta) = E_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T} R_t(\tau)\,\nabla_\theta \log\pi_\theta(a_t|s_t)\right]$$

Single sample = single trajectory τ

$$\nabla_\theta J(\theta) \simeq \sum_{t=0}^{T} \underbrace{R_t(\tau)}\,\nabla_\theta \log\pi_\theta(a_t|s_t)$$



$$R_t(\tau) = r_{t+1} + r_{t+2} + \dots + r_T \;\hat{=}\; q_{\pi_\theta}(s_t, a_t)$$

generated

1-Sample.

$s_t \xrightarrow{a_t} s_{t+1}$

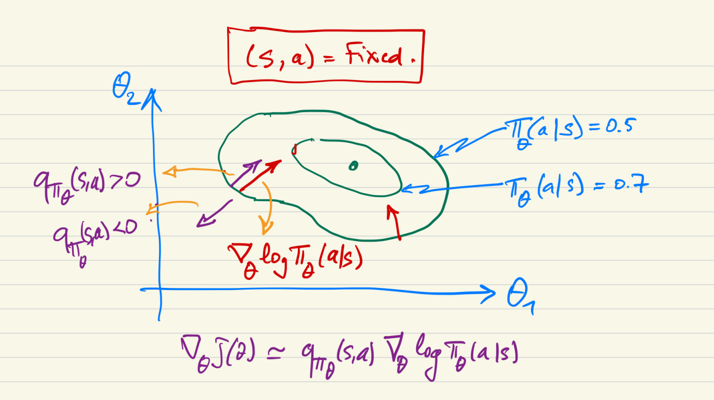
Policy gradient: Quick Recap

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \mathbb{E}_{\tau \sim \pi_\theta}\left[\sum_{t=0}^{T} R_t(\tau) \nabla_\theta \log \pi_\theta(a_t \mid s_t)\right] \\
&= \mathbb{E}_{\tau \sim \pi_\theta}\left[\sum_{t=0}^{T} q_{\pi_\theta}(s_t, a_t) \nabla_\theta \log \pi_\theta(a_t \mid s_t)\right]
\end{aligned}
$$

For N sampled paths $\tau_i = \{s_{i,0}, a_{i,0}, s_{i,1}, r_{i,1}, \ldots\}$:

$$
\begin{aligned}
\nabla_\theta J(\theta) &\approx \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{t=0}^{T} R_t(\tau_i) \nabla_\theta \log \pi_\theta(a_{i,t} \mid s_{i,t})\right] \\
&= \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{t=0}^{T} q_{\pi_\theta}(s_{i,t}, a_{i,t}) \nabla_\theta \log \pi_\theta(a_{i,t} \mid s_{i,t})\right]
\end{aligned}
$$

Gradient policy theorem



Changing θ in the direction of $\nabla_\theta \log \pi(a \mid s)$ makes it more likely that action a will be chosen by policy π. That is a good thing if $q(s, a)$ positive/large!

Actor-Critic Methods

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau \left[\sum_{t=0}^{T} \nabla_\theta \log \underbrace{\pi_\theta(a_t \mid s_t)}_{ACTOR} \underbrace{q_{\pi_\theta}(s_t, a_t)}_{CRITIC} \right]$$

- **Critic** estimates the value function (could be action-value q or state-value v function).
- **Actor** updates the policy distribution in the direction suggested by the critic.

Introducing baselines: A2C

- Policy gradient theorem:

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t) q(s_t, a_t) \right]$$

- Problem: value of $q(s, a)$ is not very informative;
- We need a reference point or baseline: natural choice $= v(s)$
- Advantage: Relative value of an action as compared to other actions in that state:

$$A(s, a) := q(s, a) - v(s)$$

- **Advantage actor-critic (A2C)**

$$\nabla_\theta J(\theta) \propto \mathbb{E}_\tau \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t \mid s_t) (q_{\pi_\theta}(s_t, a_t) - v_{\pi_\theta}(s_t)) \right]$$

Estimating Advantage

- Typically two neural networks to estimate
  1. policy $\rightarrow \pi_\theta$
  2. value functions and advantage: $\rightarrow v_w, q_w$
  3. Network weights: θ and w
- Computational strategy:
  - Estimate $v(s)$
  - Estimate $q(s, a)$ using Bellman eqs:

$$q(s_t, a_t) = \mathbb{E}\left[r_{t+1} + \gamma v(s_{t+1})\right]$$

- Along sampled trajectory:

$$\hat{q}(s_t, a_t) = r_{t+1} + \gamma \hat{v}(s_{t+1})$$
$$\hat{A}(s_t, a_t) = r_{t+1} + \gamma \hat{v}(s_{t+1}) - \hat{v}(s_t)$$

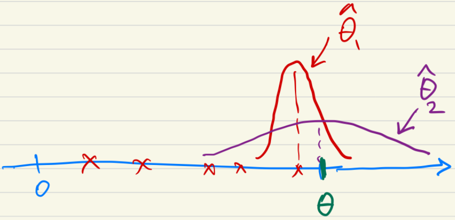
Estimating Advantage (2)

- *n*-step returns along sampled trajectory:

$$\hat{q}(s_t, a_t) = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t_3} + \ldots + \gamma^{n-1} r_{t+n} + \gamma^n \hat{v}(s_{t+n})$$

- Combination of biased and unbiased estimate:
  - Actual returns: unbiased but high variance
    (sample paths can be very different!)
  - Bias due to inclusion of estimate $\hat{v}$, but lower variance
    (average over all actions);

Mathematical aside (1): Bias vs. Variance



$$X_i \sim U(0, \theta)$$

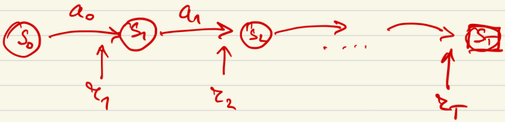
?

$$\hat{\theta}_1 = \max_i X_i \longrightarrow \boxed{\text{BIASED Small var}}$$

$$\hat{\theta}_2 = 2\bar{X} = \frac{2}{n}\sum_{i=1}^{n} X_i \longrightarrow \boxed{\text{UNBIASED Large VAR.}}$$

Mathematical aside (2a): Discount factor

Discount factor. $\boxed{\gamma = \text{Prob of Continuing}}$



$$\mathbb{E}T = \sum_{k=1}^{\infty} k\, \mathbb{P}(T=k)$$

$$= \sum_{k=1}^{\infty} k \cdot \gamma^k (1-\gamma)$$

$$= (1-\gamma)\gamma \underbrace{\sum_{k=1}^{\infty} k \gamma^{k-1}}$$

Mathematical aside (2b): Discount factor

$$\sum_{k=1}^{\infty} k\gamma^{k-1} = \frac{d}{d\gamma}\left(\sum_{k=1}^{\infty}\gamma^k\right)$$

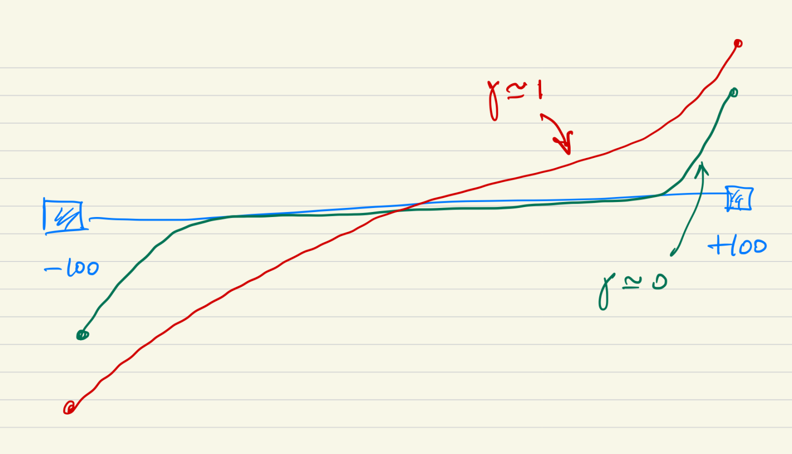
$$= \frac{d}{d\gamma}\left(\frac{\gamma}{1-\gamma}\right)$$

$$= \frac{(1-\gamma)-\gamma(-1)}{(1-\gamma)^2} = \frac{1}{(1-\gamma)^2}$$

$$\boxed{ET = (1-\gamma)\gamma\frac{1}{(1-\gamma)^2} = \frac{\gamma}{1-\gamma}}$$

Eg:
$\gamma = 0.9$
$ET = 9.$

Mathematical aside (2c): Discount factor



Further reading

https://lilianweng.github.io/lil-log/2018/04/08/policy-gradient-algorithms.html