Week-10 - RL

L1 Reinforce : MC Policy Gradient

$$\nabla J(\theta) \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \nabla \pi(a|s,\theta)$$

$$\theta_{t+1} = \theta_t + \alpha G_t \frac{\nabla \pi(A_t|S_t,\theta_t)}{\pi(A_t|S_t,\theta_t)}$$

- it uses the policy of comp. return from time t, which includes all the future rewards up until the end of the episode.
- the algo. computes an unbiased estimate of the gradient. Can be very slow due to high variance (occurs due to recurrence time).
- With Baseline

$$\nabla J(\theta) \propto \sum_s \mu(s) \sum_a (q_\pi(s,a) - b(s)) \nabla \pi(a|s,\theta)$$

Update rule,

$$\theta_{t+1} = \theta_t + \alpha \left(G_t - b(S_t)\right) \frac{\nabla \pi(A_t|S_t,\theta_t)}{\pi(A_t|S_t,\theta_t)}$$

12 Actor Critic Methods - I

- ACM → learn both a policy & a state value func^n simultaneously. The policy is referred to as the actor that suggests action given a state.
- The estimated value func^n is called the critic. It evaluates the ac^{ns} taken by the actor based on the given policy.

$$\theta_{t+1} = \theta_t + \alpha \left(G_{t:t+1} - \hat{v}(s_t, w) \right) \frac{\Delta \pi(A_t | S_t, \theta_t)}{\pi(A_t | S_t, \theta_t)}$$

introducing TD error,

$$\theta_{t+1} = \theta_t + \alpha \delta_t \frac{\nabla \pi(A_t | S_t, \theta_t)}{\pi(A_t | S_t, \theta_t)}$$

- 1 step actor critic method, is like an episodic task for estimating the $\pi_\theta \approx \pi$.
- comparing the current method to baseline

1. The G_t term although unbiased causes high variance.
2. since, $E_\pi[G_t | S_t, A_t] = q_\pi(S_t, A_t)$ if somehow, we had an estimate of $q_\pi(S_t, A_t)$ with less variance then we can use it instead of G_t.
3. In 1 step AC, we use $\hat{v}$ for both estimation of $q_\pi(S_t, A_t)$ & the baseline
4. The bootstrapping method in the update introduces bias but reduces variance.
5. reduced variance helps in increased learning.

L3 ACM-2

- actor - computes the policy π_θ and updates θ.
- critic - computes the estimate $\hat{v}(s, w)$ of the state value func^n. Also, updates the parameter w.
- basic algo. for actor critic —
  1. take ac^n $a \sim \pi_\theta(a|s)$ & receives (s, a, s', r)
  2. update w using data $(s, r + \gamma \hat{v}(s', w))$

3. calculate,
$\hat{\delta}(s,a) = r + \gamma \hat{v}(s', w) - \hat{v}(s, w)$
4. $\theta \leftarrow \theta + \alpha \cdot \hat{\delta}(s,a) \cdot \nabla_\theta \log \pi_\theta(a|s)$
5. repeat

- how to update the critic?
→ we will have multiple data points of the form (s, y)
→ min. the squared loss

$$L(w) = \frac{1}{N} \sum_i \| \hat{v}(s_i, w) - y_i \|^2$$

↳ is the batch size

- Advantage func^n → the diff. btw q value & value func^n.

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

- A3C → asynchronous advantage actor critic. Training happens parallely
- A2C → synchronous adv. actor critic.
↳ presence of a coordinator, The updates of the global parameters are executed only after the threads have finished their computa^n.

L4 DPG & DDPG ⟶ deep DPG

↳ deterministic policy gradient

- policy gradient over continuous spaces!

$$\nabla J(\theta) = E_\pi \left[\sum_a q_\pi(s_t, a) \nabla \pi(a | s_t, \theta) \right]$$

↳ for discrete acⁿs

$$\nabla_\theta J(\theta) = E_{s \sim \mu^\pi(s), a \sim \pi_\theta} \left[\nabla_\theta \log \pi_\theta(a|s) \, q_{\pi_\theta}(s, a) \right]$$

↳ for continuous acⁿs

- it is often hard to implement the differentiable continuous controllers in many problems. Expectaⁿ over both states & acⁿs.
- deterministic policies

$$\pi_\theta : S \to A \quad (\text{det.})$$

$$J(\theta) = \int_S \mu^\pi(s) \, q^\theta_{\pi_\theta}(s, \pi_\theta(s)) \, ds$$

↳ the continuous acⁿ spaces allow us to think of Δ in acⁿ w.r.t the policy parameter. Now, we will have expectaⁿ only over states.

- DPG - it is a limiting case of stochastic policy gradient for a very wide class of stochastic policies.
  → If the pol. is stochastic, then not a problem. Otherwise, we will use an off-policy actor critic, where the behavioral policy differs from the estimation policy.
- DDPG = DPG + DQN (off-policy)

$$\mu'(s) = \mu_\theta(s) + N \quad \text{(noise)}$$

  → use replay buffer
  → maintains seperate target network parameters θ', w' & uses soft updates.
  → use batch normalization method to normalize input state features & minimize covariate shift.