DPG and DDPG

B. Ravindran

Policy gradient over continuous action spaces

- Policy Gradient Theorem:

$$\begin{aligned}\nabla_\theta J(\theta) &= \int_{\mathcal{S}} \mu^\pi(s) \int_{\mathcal{A}} \nabla_\theta \pi_\theta(a|s) q_{\pi_\theta}(s,a) da ds \\ &= \mathbb{E}_{s\sim\mu^\pi(s), a\sim\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a|s) q_{\pi_\theta}(s,a)\right]\end{aligned}$$

Problems with continuous action spaces.

- Hard to implement differentiable continuous controllers in many problems.
- Expectation over both states and actions. If we can reduce the expectation to only over states, this might simplify gradient estimation.

Deterministic Policies

- Let the policy $\pi_\theta : \mathcal{S} \to \mathcal{A}$ be deterministic.
- That is, $\pi_\theta(s)$ will output the action to be taken at state s.
- The performance objective:

$$J(\theta) = \int_{\mathcal{S}} \mu^{\pi}(s) q_{\pi_\theta}(s, \pi_\theta(s))\, ds$$

- Notice how there is only one integral now as we have made the policies deterministic.

Deterministic policy gradient(DPG)

- What is the notion of gradient of deterministic policies? Continuous actions spaces allow us to think of change in actions w.r.t the policy parameter.

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \int_S \mu^\pi(s) \nabla_a q_\pi(s,a) \nabla_\theta \pi_\theta(s)|_{a=\pi_\theta(s)} ds \\
&= \mathbb{E}_{s\sim\mu^\pi}\left[\nabla_a q_\pi(s,a) \nabla_\theta \pi_\theta(s)|_{a=\pi_\theta(s)} ds\right]
\end{aligned}
$$

- We now have expectation only over states.
- Avoided max over actions by moving in the direction of the gradient of q_π.

Deterministic policy gradient

- For a wide class of stochastic policies, the deterministic policy gradient is a limiting case of the stochastic policy gradient
- Update equations in actor critic framework:

$$\begin{aligned}
\delta_t &= r_t + \gamma \hat{q}(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}, \mathbf{w}) - \hat{q}(\mathbf{s}_t, \mathbf{a}_t, \mathbf{w}) \\
\mathbf{w} &\leftarrow \mathbf{w} + \alpha_{\mathbf{w}} \delta_t \nabla_{\mathbf{w}} \hat{q}(\mathbf{s}_t, \mathbf{a}_t, \mathbf{w}) \\
\theta &\leftarrow \theta + \alpha_\theta \nabla_a \hat{q}(\mathbf{s}_t, \mathbf{a}_t, \mathbf{w}) \nabla_\theta \pi_\theta(\mathbf{s}_t)|_{a=\pi_\theta(s)}
\end{aligned}$$

Deterministic policy gradient

- How do we ensure exploration?
  - If the environment is stochastic, then not an issue.
  - Otherwise use off-policy actor critic, where the behaviour policy differs from the estimation policy
- Requires compatible parametrizations.

DDPG(Deep DPG)

- DDPG combines DPG with DQN.
- The algorithm is off-policy. Behaviour policy:

$$\pi'(s) = \pi_\theta(s) + \mathcal{N} \text{ ——— Noise}$$

- Uses replay buffer: alleviates moving target problem.
- Maintains separate target network parameters $\theta', \mathbf{w}'$ and uses soft updates
- Uses Batch Normalization to normalize input state features and minimize covariate shift.

Update Equations

Target
network
params

- Critic Network:

$$\delta_t = r_t + \gamma \hat{q}(\mathbf{s}_{t+1}, \pi_{\theta'}(\mathbf{s}_{t+1}), \mathbf{w}') - \hat{q}(\mathbf{s}_t, \mathbf{a}_t, \mathbf{w})$$
$$\mathbf{w} \leftarrow \mathbf{w} + \alpha_{\mathbf{w}} \delta_t \nabla_{\mathbf{w}} \hat{q}(\mathbf{s}_t, \mathbf{a}_t, \mathbf{w})$$

- Policy Network:

$$\theta \leftarrow \theta + \alpha_\theta \nabla_a \hat{q}(\mathbf{s}_t, \mathbf{a}_t, \mathbf{w}) \nabla_\theta \pi_\theta(\mathbf{s}_t)|_{a=\pi_\theta(s)}$$

DDPG Algorithm

Algorithm 1: DDPG algorithm

Randomly initialize critic network $\hat{q}(s, a, \mathbf{w})$ and actor $\pi_\theta(s)$ with weights $\mathbf{w}$ and θ
Initialize target network parameters $\mathbf{w}' \leftarrow \mathbf{w}$, $\theta' \leftarrow \theta$
Initialize replay buffer $\mathcal{R}$
for *episode=1, M* **do**
  Initialize a random process $\mathcal{N}$ for action exploration
  Receive initial observation s_1
  for t*=1,T* **do**
    Select action $a_t = \pi_\theta(s_t) + \mathcal{N}$ according to the current policy and exploration noise
    Execute action a_t and observe reward r_t and observe new state s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{R}$
    Sample a random minimatch of N transitions (s_i, a_i, r_i, s_{i+1}) from $\mathcal{R}$
    Set $y_i = r_i + \gamma \hat{q}(s_{i+1}, \pi_{\theta'}(s_{i+1}), \mathbf{w}')$
    Update critic by minimizing the loss: $L = \frac{1}{N}\sum_i (y_i - \hat{q}(s_i, a_i, \mathbf{w}))^2$
    Update the actor policy using the sampled policy gradient:

$$\nabla_\theta J \approx \frac{1}{N}\sum_i \nabla_a \hat{q}(s_i, a_i, \mathbf{w}) \nabla_\theta \pi_\theta(s_i)|_{a=\pi_\theta(s)}$$

    Update the target parameters:

$$\mathbf{w}' \leftarrow \tau \mathbf{w} + (1-\tau)\mathbf{w}'$$
$$\theta' \leftarrow \tau\theta + (1-\tau)\theta$$

Soft updates

  end
end