11. DDPG 알고리즘 분석

김호철

Contents

1 소개

- **DQN**
  - 고차원(high-dimensional) 연속적인(continuous) 상태 공간을 가능하게 하였음
  - 하지만, 액션공간이 저차원(low-dimensional)이고, 이산적(discrete)임
  - 이산화(Discretization) : 연속공간을 이산공간으로 변경하는것
  - 이산화하여 사용할 수 있지만 상태공간이 크고 정교한 로보틱스 같은 분야에서는 불가능
- **DPG(Deterministic Policy Gradient)**
  - DPG는 연속적 액션 공간에서 액터로 결정적(deterministic) 정책 $a = \mu(s)$와 크리틱으로 액션-벨류 함수 $Q(s, a)$를 학습한다.
  - 폴리시 그레디언트 알고리즘은 보통 액터로 확률적(stochastic) 정책 $\pi(a|s)$을 학습한다.
- **DDPG(Deep Deterministic Policy Gradient)**
  - DDPG는 액터-크리틱 알고리즘이다. 액터는 정책을 추정하고, 크리틱은 Q-펑션을 추정한다.
  - (정책 추정을 하는 액터 관점) 연속 액션 공간을 핸들링하기 위해 결정적 정책을 학습하는 DPG를 기반으로 한다.
  - (Q-펑션을 추정하는 크리틱 관점) 경험 리플라이와 타겟 네트워크같은 DQN을 사용한다.
  - DDPG의 어려운 점은 가끔 성능이 개선되지 않는 경우(네트워크는 학습되지만 성능 개션은 되지 않는 경우)가 있다. (TRPO, PPO에서 해결)

2 DPG(Deterministic Policy Gradient)

2.1 DPG 개요

- DPG는 연속 액션 학습을 위해 결정적 정책을 사용한다.
- 확률적(stochastic) 정책 기울기 이론과 유사한 결정적 정책 기울기 이론을 기반으로 한다.
- 결정적 정책은 액션을 반환 $a = \mu(s) \leftarrow$ 확률적 정책은 확률을 반환 $\pi(a|s)$
- 확률적 폴리시 그레디언트(SPG)는 **상태와 액션의 쌍**을 사용해서 파라메터들을 업데이트한다.

$$\Delta\theta = Q^{\pi}(s,a)\nabla_{\theta}\log\pi_{\theta}(a|s) \tag{1}$$

- 결정적 폴리시 그레디언트(DPG)는 오직 **상태**만 사용해서 파라메터들을 업데이트한다.

$$\Delta\theta = \nabla_{a}Q^{\mu}(s,a)\mid_{a=\mu_{\theta}(s)}\nabla_{\theta}\mu_{\theta}(s) \tag{2}$$

- 그러므로 DPG는 SPG이 비해 적은 샘플이 요구되어진다.

2.2 상태 방문 빈도(State visitation frequency)

- 상태 방문 빈도는 어떤 정책 π에 따라 주어진 상태를 방문할 확률의 할인된 합계이다.

$$\begin{aligned}\rho_{\pi}(s) &= \sum_{t=0}^{\infty}\gamma^{t}P(s_t = s|\pi)\\ &= \int_{S}\sum_{t=0}^{\infty}\gamma^{t}P_0(s')P(s' \to s|t,\pi)ds'\end{aligned} \tag{3}$$

- 여기서 $P_0(s')$는 초기 상태 방문 확률이고, $P(s' \to s|t,\pi)$는 정책 π를 따라 t단계에서, 상태 s'에서 상태 s로의 방문 확률이다.
- 상수 계수(Constant Factor)까지, ρ_{π}는 상태 공간에 대한 확률 분포로 간주된다.

$$\begin{aligned}\sum_{s\in S}\rho_{\pi}(s) &= \sum_{t=0}^{\infty}\gamma^{t}\sum_{s\in S}P(s_t = s|\pi)\\ &= \sum_{t=0}^{\infty}\gamma^{t} = \frac{1}{1-\gamma}\end{aligned} \tag{4}$$

- 그러므로 ρ_{π}에 $1-\gamma$를 곱하면 확률 분포가 된다.

2.3 SPG vs DPG 목적 함수

- **확률적 폴리시 그레디언트 이론**(1999년 서튼 등)
  - SPG 목적 함수는,

$$\begin{aligned} J(\theta) &= \mathbb{E}_{s\sim\rho_\pi, a\sim\pi_\theta}[Q^\pi(s,a)] \\ &= \int_{\mathcal{S}} \rho_\pi(s) \int_{\mathcal{A}} \pi_\theta(a|s) Q^\pi(s,a) da ds \end{aligned} \tag{5}$$

  - 폴리시 그레디언트를 위해 미분을 적용하면,

$$\begin{aligned} \nabla_\theta J(\theta) &= \int_{\int} \rho_\pi(s) \int_{\mathcal{A}} Q^\pi(s,a) \nabla_\theta \pi_\theta(a,s)\ ds\ ds \\ &= \mathbb{E}_{s\sim\rho_\pi, a\sim\pi_\theta}[Q^\pi(s,a)\ \nabla_\theta \log \pi_\theta(a|s)] \end{aligned} \tag{6}$$

  - 확률적 폴리시 그레디언트 업데이트는(크리틱 액터),

$$\Delta\theta = Q^\pi(s,a) \nabla_\theta \log \pi_\theta(a|s) \tag{7}$$

- **결정적 폴리시 그레디언트 이론**(2014년 실버 등)
  - DPG 목적 함수는,

$$\begin{aligned} J(\theta) &= \mathbb{E}_{s\sim\rho_\pi}[Q^\mu(s,a)] \\ &= \int_{\mathcal{S}} \rho_\mu(s) Q^\mu(s,a) da \quad where \quad a = \mu_\theta(s) \end{aligned} \tag{8}$$

  - 폴리시 그레디언트를 위해 미분을 적용하면,

$$\begin{aligned} \nabla_\theta J(\theta) &= \int_{\int} \rho_\pi(s) \nabla_a Q^\mu(s,a)|_{a=\mu_\theta(s)} \nabla_\theta \mu_\theta(s)\ ds \\ &= \mathbb{E}_{s\sim\rho_\pi}[\nabla_a Q^\mu(s,a)|_{a=\mu_\theta(s)} \nabla_\theta \mu_\theta(s)] \end{aligned} \tag{9}$$

  - 결정적 폴리시 그레디언트 업데이트는(크리틱 액터),

$$\Delta\theta = \nabla_a Q^\mu(s,a)|_{a=\mu_\theta(s)} \nabla_\theta \mu_\theta(s) \tag{10}$$

3 DDPG(Deep Deep Deterministic Policy Gradient)

3.1 DDPG

- DDPG는 DPG 기반의 액터-크리틱알고리즘이다.
- 연속적인 액션 공간을 핸들링하기 위해, 크리틱은 Q-함수 $Q(s,a)$를 학습하고, 액터는 결정적 정책 $a = \mu(s)$를 학습한다.
- DDPG는 또한 DQN의 혁신적 두가지 아이디어(경험 리플라이, 타겟 네트워크)를 사용한다.

- 손실 함수(Loss Function, 최소화)은,

$$L(\phi) = \mathbb{E}_{s\sim\rho_\pi}\left[\left(r + \gamma\hat{Q}_{\hat{\phi}}(s', \hat{\mu}_{\hat{\theta}}(s')) - Q_\phi(s,a)\right)^2\right] \ \ where \ \ a = \mu_\theta(s) \tag{11}$$

- 크리틱 $Q_\phi(s,a)$ 업데이트는 타겟 네트워크를 사용하여 다음과 같다.

$$-\Delta(\theta) = \left(r + \gamma\hat{Q}_{\hat{\phi}}(s', \hat{\mu}_{\hat{\theta}}(s')) - Q_\phi(s,a)\right)\nabla_\theta Q_\phi(s,a) \tag{12}$$

- 목적 함수(Target Function, 최대화)는,

$$J_\theta = \mathbb{E}_{s\sim\rho_\mu}[Q_\phi(s,a)] \ \ where \ \ a = \mu_\theta(s) \tag{13}$$

- 액터 $\mu_\theta(s)$ 업데이트는 다음과 같다.

$$\Delta(\theta) = \nabla_a Q_\phi(s,a)|_{a=\mu_\theta(s)}\nabla_\theta\mu_\theta(s) \tag{14}$$

- 액터와 크리틱 모두 행위(behavior)와 타겟에서 동일한 네트워크를 공유하지만, 각자 자체 가중치 $\phi, \theta, \hat{\phi}, \hat{\theta}$를 사용한다.
- 정책의 결정적으로 인해 더 많은 탐색을 위해 추가 노이즈가 필요하다. 추가 노이즈 처리는,

$$\mathcal{N} : a_t = \mu_\theta(s_t) + \mathcal{N}_t \tag{15}$$

- 타겟 네트워크 업데이트는 소프트 업데이트를 적용한다.

$$\begin{aligned} \hat{\phi} &\leftarrow \tau\phi + (1-\tau)\hat{\phi} \\ \hat{\theta} &\leftarrow \tau\theta + (1-\tau)\hat{\theta} \end{aligned} \tag{16}$$

3.2 DDPG 알고리즘

크리틱 네트워크 $Q(s, a; \phi)$와 액터 네트워크 $\mu(s; \theta)$를 무작위 초기화
타겟 네트워크 $\hat{Q}, \hat{\mu}$를 가중치 $\hat{\phi} = \phi, \hat{\theta} = \theta$로 초기화
리플라이 버퍼 $\mathcal{R}$을 초기화
에피소드 1에서 M까지 반복
  액션 탐색을 위해 노이즈 프로세스 $\mathcal{N}$을 무작위 초기화
  초기 관측 상태 s_1을 받는다.
  t를 1에서 T까지 반복
    액션 $a_t = \mu(s_t; \theta) + \mathcal{N}$을 선택 (DQN에서는 $a_t = \mathrm{argmax}_a\, Q(s_t, a; \theta)$)
    a_t를 실행하고, r_{t+1}, s_{t+1}을 관측한다.
    $\mathcal{R}$에서 트랜지션 $(s_t, a_t, r_{t+1}, s_{t+1})$를 저장한다.
    $\mathcal{R}$에서 B개의 트랜지션 $(s_t, a_t, r_{t+1}, s_{t+1})$ 미니배치를 샘플링한다.
    각 샘플에 대해서 타겟을 계산한다.
      $y_i = r_{i+1} + \gamma \hat{Q}(s_{i+1}, \hat{\pi}(s_{i+1}; \hat{\theta}); \hat{\phi})$
    손실함수를 최소화하도록 크리틱 네트워크를 업데이트한다.
      $L = \frac{1}{B} \sum_i (y_i - Q(s_i, a_i; \phi))^2$
    결정적 폴리시 그레디언트를 사용하여 액터 네트워크를 업데이트 한다.
      $\nabla_\theta J \approx \frac{1}{B} \sum_i \nabla_a Q(s_i, a; \phi)|_{a=\mu(s_i;\theta)} \nabla_\theta \mu(s_i; \theta)$
    타겟 네트워크를 업데이트 한다.
      $\hat{\phi} \leftarrow \tau\phi + (1-\tau)\hat{\phi}, \hat{\theta} \leftarrow \tau\theta + (1-\tau)\hat{\theta}$