10. A3C 알고리즘 분석

김호철

Contents

---

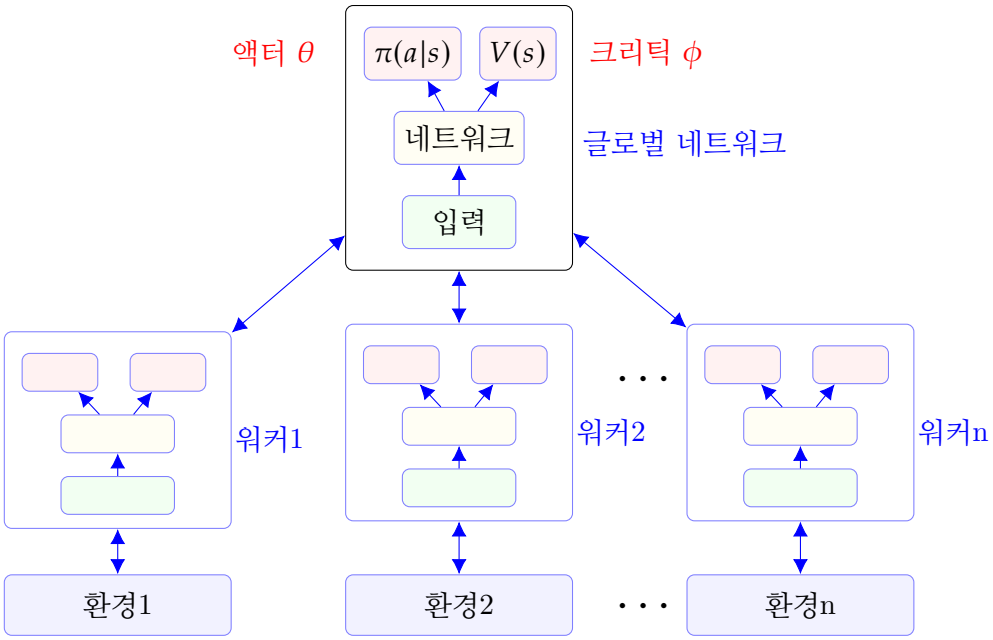
1 개요

- Asynchronous Methods for Deep Reinforcement Learning, Google DeepMind 2016
- A3C(Asynchronous Advantage Actor-Critic)는 다수개의 작업자(Worker) 에이전트들과 하나의 글로벌 네트워크로 구성된 액터-크리틱 알고리즘이다.
- 작업자 에이전트들은 병렬(parallel)로 작업하며, 글로벌 네트워크 파라메터를 비동기(asynchronously)로 업데이트한다.
- 이 병렬 처리는 주어진 시간 단계에서 여러 에이전트가 다양한 상태를 경험하기 때문에 각 에이전트의 시간적 상관(temporal correlation) 문제(DQN의 경험 리플라이 필요없음)를 줄인다.
- 크리틱 네트워크는 실제로 $Q(s,a)$ 대신 n-스텝 리턴 $G_t^{(n)}$을 사용함으로써, 어드밴티지(Advantage)$A(s,a) = Q(s,a) - V(s)$를 추정하는데 $G^{(n)} - V(s)$를 사용한다.
- 이산(discrete)과 연속(continuous) 액션 공간 모두에서 작동하므로, 복잡한 상태/액션(입력/출력) 공간의 새로운 도전적 문제에 DRL 알고리즘을 많이 사용한다.

2 A3C(Asynchronous Advantage Actor-Critic)

2.1 구조

- A3C는 다중 네트워크를 활용한다.
  - 하나의 글로벌 네트워크와 여러 작업자 에이전트가 병렬로 작동한다.
  - 각 작업자 에이전트는 서로 다른 네트워크 파라메터를 가지며 자체 환경(copy)과 상호작용한다.
- 작업자 에이전트는 각 에이전트에서 계산된 누적 기울기에 의해 글로벌 네트워크의 파라메터를 비동기(asynchronously)으로 업데이트한다.



- 단일 RL 에이전트에는 일반적으로 non-stationary와 temporal correlation 문제가 발생한다.
- 그러나 다중 작업자 에이전트가 있는 비동기식 RL은 다음을 제공한다.
  - Temporal Correlation 감소 : 각 에이전트의 경험은 서로 독립적이다.
  - DQN에서와 같이 경험 리플라이가 필요하지 않다. (더 적은 계산 및 메모리 필요)
  - 서로 다른 탐사 정책은 다양한 경험을 제공한다. (고정식, 더 견고함)
  - 온-폴리시 RL이 가능하다. (살사, 액터-크리틱 등)
  - 대부분의 딥러닝 방법과 달리 A3C는 GPU 대신 단일 멀티 코어 CPU에서 실행할 수 있다.

2.2 비동기 구현

- 각 작업자 에이전트는 병렬 학습을 위해 별도로 실행된다.
  - 글로벌 네트워크는 공유(shared) 액터 파라메터 θ와 크리틱 파라메터 ϕ를 가진다.
  - 그리고, 여러 작업자 에이전트는 이 공유 파라메터에 비동기로 업데이트한다.
  - 각 에이전트는 작업이 완료되면 신속히 공유 파라메터를 업데이트한다.
- 각 업데이트 사이에 각 에이전트는 글로벌 네트워크의 공유 파라메터 사본을 가져온다.
  - 에이전트의 파라메터는 $\theta' = \theta$ 및 $\phi' = \phi$이고,
  - 로컬 에이전트 정책을 통해 작동하는 시뮬레이터를 t_{max} 단계만큼 실행한다.
- t_{max} 단계 동안, 각 에이전트는 자체 프로세스에서 누적 기울기(accumulated gradient)를 계산하고, 최종적으로 공유 파라메터를 업데이트한다.
  - 에이전트 액터 :

$$\Delta\theta \leftarrow \Delta\theta + (G_t^{(n)} - V_{\phi'}(s_t))\nabla_{\theta'} \log \pi_{\theta'}(a_t|s_t) \tag{1}$$

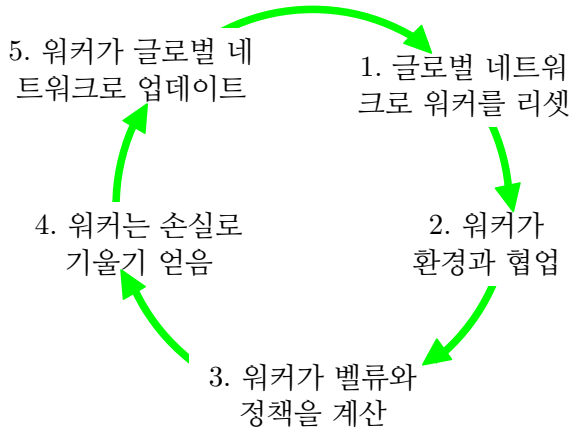
  - 에이전트 크리틱 :

$$\Delta\phi \leftarrow \Delta\phi + (G_t^{(n)} - V_{\phi'}(s_t))\nabla_{\theta'} V_{\phi'}(s_t) \tag{2}$$

  - 글로벌 네트워크 :

$$\begin{aligned} \theta &\leftarrow \theta + \alpha\Delta\theta \\ \phi &\leftarrow \phi - \beta\Delta\phi \end{aligned} \tag{3}$$

- 이 병렬 학습은 에이전트 수에 따라 선형적으로 속도 향상이 된다.



2.3 어드밴티지

- 폴리시 그레디언트 업데이트는 에이전트에게 어떤 행동이 '좋았고' '나빴는지' 알려주기 위해, 트라젝토리(trajectories:궤적)의 미니배치에서 나온 리턴 G_t를 사용한다.
- 그런 다음 액션을 적절하게 장려(encourage)하거나 억제(discourage)하기 위해 네트워크가 업데이트된다.
- 리턴 G_t 대신, 어드밴티지 $A(s,a) = Q(s,a) - V(s)$를 사용하면, 에이전트가 액션이 얼마나 좋은지 뿐만 아니라, 예상보다 얼마나 더 나은 결과를 얻었는지 결정할 수 있도록 한다.
- A3C 크리틱 네트워크는 실제에서는 어드밴티지 $A(s,a) = Q(s,a) - V(s)$를 추정하는 $G_t^{(n)} - V(s)$를 사용한다.
  - $A_{\phi_1\phi_2}(s,a) = Q_{\phi_2}(s,a) - V_{\phi_1}(s)$를 사용하면 두개의 네트워크가 필요하다.
  - $Q(s,a)$ 대신 n-스텝 리턴 $G_t^{(n)}$을 네트워크 $Q_{\phi_2}(s,a)$가 필요하지 않다.
- 또한 n-스텝 리턴을 통해 부트스트래핑 정도를 유연하게 선택함으로써, PG의 리턴 G_t로 인한 그래디언트 분산을 줄이고 훈련 속도를 가속화할 수 있다.

2.4 A3C 알고리즘

* 각 개별 워커 에이전트에서의 알고리즘(액터:θ, 크리틱:ϕ)

글로벌 네트워크의 공유 파라메터 θ, ϕ, 하나의 워커 파라메터 θ', ϕ'

스텝 카운트를 1로 초기화

true 이면 다음을 반복

기울기 $d\theta \leftarrow 0, d\phi \leftarrow 0$ 을 리셋

파라메터들 $\theta' = \theta, \phi' = \phi$를 동기화(Copy) 시킨다.

$t_{start} = t$로 세팅하고, 상태 s_t를 얻는다.

s_t가 종료 상태가 아니고, $t \leq t_{start} + t_{max}$ 이면 다음을 반복 : 트라젝토리 생성

정책 $\pi(a_t|s_t;\theta')$에 따라 액션 a_t를 선택

a_t를 실행하고, 보상(r_{t+1})과 다음 상태(s_{t+1})를 관측

$t \leftarrow t + 1$

$R = V(s_t;\theta')$ (s_t가 종료 상태이면 0)

$t-1$에서 t_{start}까지 i를 반복 : n-스텝 리턴 적용을 위해 리버스 반복

$R \leftarrow r_{i+1} + \gamma R$: n-스텝 리턴

기울기 θ' 누적 : $d\theta \leftarrow d\theta + (R - V(s_i;\phi'))\nabla_{\theta'} \log \pi(a_i|s_i;\theta')$

기울기 ϕ' 누적 : $d\phi \leftarrow d\phi + (R - V(s_i;\phi'))\nabla_{\theta'} \log V(s_i;\phi')$

비동기 업데이트 : $\theta_G \leftarrow \theta_G + \alpha d\theta_W, \phi_G \leftarrow \phi_G + \beta d\phi_W$

* 실제에서는 $\pi(a_t|s_t;\theta)$의 소프트맥스 출력과, $V(s_i;\phi)$의 선형 출력을 제외한 모든 파라메터를 CNN을 사용하여 공유한다.

3 A2C

- A2C는 A3C의 동기화(synchronous) 버전이다.
- A3C에서 각 작업자 에이전트는 글로벌 네트워크 매개변수를 비동기식으로 업데이트하므로 때때로 이러한 에이전트가 서로 다른 정책을 사용하여 집계된 업데이트가 최적이 아닐 수 있다.
- A2C에서 이러한 불일치를 해결하기 위해 글로벌 네트워크는 모든 병렬 작업자 에이전트가 현재 반복 작업을 완료할 때까지 파라메터 업데이트를 기다린다.
- 그리고, 다음 반복에서 병렬 에이전트는 동일한 정책으로 시작한다.
- 코디네이터(Coordinator)를 사용하여 동기화된 그래디언트 업데이트를 통해 훈련이 더 응집력이 있고 잠재적으로 더 빠르게 수렴할 수 있다.
- A2C는 GPU를 더 효율적으로 활용할 수 있으며, A3C와 같거나 더 나은 성능을 달성하면서 큰 배치 크기에서 더 잘 작동한다.