국민대학교 KPSC & AIM 스터디 – 강화학습을 이용한 체스 AI 만들기

Introduction to RL, Week 11

Chris Ohk
utilForever@gmail.com

Actor-Critic 이론

- 빠른 복습
  - 그리드월드 예제에서 폴리시 그레이디언트의 일종인 REINFORCE를 구현했다.
  - REINFORCE 알고리즘은 일종의 몬테카를로 폴리시 그레이디언트로서 에피소드마다만 학습할 수 있다는 단점이 있다.
  - 또한 에피소드가 길어지면 길어질수록 특정 상태(s, a)에 대한 반환값의 변화가 커진다.
  - 또한 반환값은 온전히 에이전트가 환경으로부터 받은 보상만으로 구하기 때문에 분산이 큰 경향이 있다.

Actor-Critic 이론

- 액터-크리틱(Actor-Critic)
  - REINFORCE 알고리즘의 단점을 해결하기 위해 다이내믹 프로그래밍의 정책 인터레이션 구조를 사용해 매 타임스텝마다 학습할 수 있도록 한다.
  - 정책 이터레이션은 크게 정책 발전과 정책 평가로 나눌 수 있다.
  - 정책 이터레이션에서는 정책이 따로 존재하며 정책을 가치함수를 통해 평가하고 평가를 토대로 정책을 발전시킨다.

Actor-Critic 이론

- 정책 이터레이션 vs 폴리시 그레이디언트
  - 정책 이터레이션에서는 가치함수에 대한 탐욕 정책을 그대로 발전시켰지만
    폴리시 그레이디언트에서는 정책 신경망의 업데이트로 이 과정을 대체한다.
  - 문제는 정책 평가다. 정책 이터레이션에서 정책 평가는
    다이내믹 프로그래밍을 통해 정책에 대한 가치 함수를 구하는 과정이었다.

Actor-Critic 이론

- 정책 이터레이션 vs 폴리시 그레이디언트
  - 폴리시 그레이디언트의 정책 신경망 업데이트 식

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta \mathrm{J}(\theta) \approx \theta_t + \alpha[\nabla_\theta \log \pi_\theta(a|s) q_\pi(a|s)]$$

  - 이 식에서 $q_\pi(s, a)$가 정책 평가의 역할을 한다.
    큐함수가 정책대로 행동했을 때 에이전트가 받으리라 기대하는 가치를 나타내기 때문이다.
  - 하지만 문제는 테이블 형태로 가치함수 또는 큐함수를 저장하는 것이 아니라서
    $q_\pi(s, a)$를 알 수가 없다는 거다.
  - 따라서 $q_\pi(s, a)$를 반환값인 G_t로 치환해서 정책 신경망을 업데이트하는 것이
    REINFORCE 알고리즘이며, 그 식은 다음과 같다.

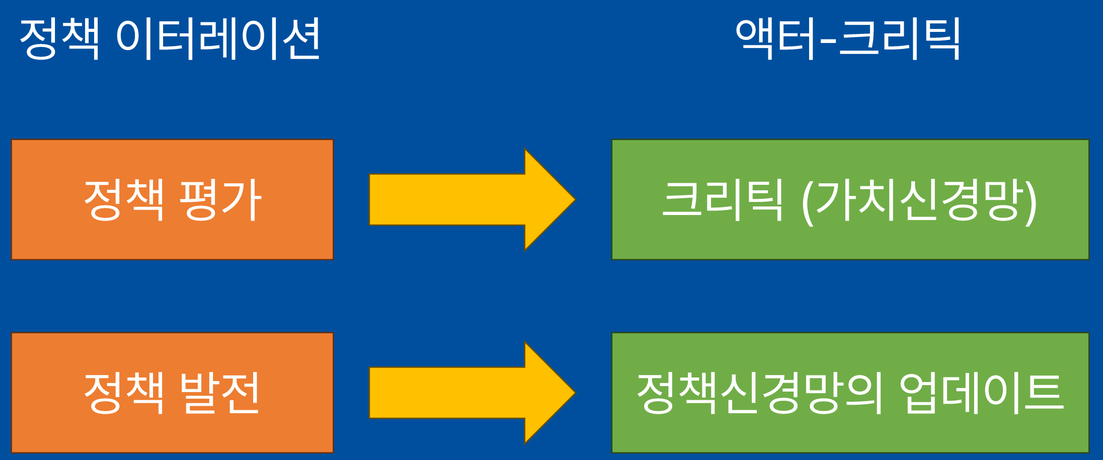
$$\theta_{t+1} \approx \theta_t + \alpha[\nabla_\theta \log \pi_\theta(a|s) G_t]$$

Actor-Critic 이론

- 정책 이터레이션 vs 폴리시 그레이디언트
  - 하지만 반환값을 사용하지 않고 큐함수를 사용하는 방법이 있다.
    바로 큐함수 또한 근사하는 방법이다.
  - 인공신경망을 2개 만들어서 하나는 정책을 근사하고, 다른 하나는 큐함수를 근사하는 거다.
    큐함수를 근사하는 신경망을 **가치신경망**이라고 한다.
  - 가치신경망의 역할은 정책을 평가하는 것이기 때문에 **크리틱(Critic)**이라는 이름이 붙는다.

Actor-Critic 이론

- 정책 이터레이션과 액터-크리틱의 관계
  - 정책의 발전은 정책신경망의 업데이트로 발전
  - 정책의 평가는 크리틱이라는 가치신경망을 사용해서 진행



- Actor-Critic의 업데이트 식 (가치신경망의 가중치는 w)

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta \mathrm{J}(\theta) \approx \theta_t + \alpha [\nabla_\theta \log \pi_\theta(a|s) Q_w(s,a)]$$

Actor-Critic 이론

- Actor-Critic의 오류함수
  - REINFORCE에서는 정책신경망 출력의 크로스 엔트로피와 반환값의 곱이었다.
  - 딥살사나 DQN에서는 MSE 오류함수를 사용하기 때문에 큐함수가 그대로 오류함수로 들어가는 것이 아니라 MSE 식을 통해 정답과 예측의 차이가 오류함수의 값으로 사용된다.
  - 하지만 Actor-Critic에서는 오류함수가 다음과 같다.

    오류함수 = 정책신경망 출력의 크로스 엔트로피 × 큐함수(가치신경망 출력)
  - 즉, 큐함수가 그대로 크로스 엔트로피에 곱해진다.
    이 경우 큐함수의 값에 따라 오류함수의 값이 많이 변화하게 된다 (즉, 분산이 크다).

Actor-Critic 이론

- Actor-Critic의 오류함수
  - 따라서 큐함수의 변화 정도를 줄여주기 위해 **베이스라인(Baseline)**을 사용한다.
  - 모든 상태에서 모든 행동의 큐함수의 평균을 구해서 그 값을 베이스라인으로 사용할 수도 있겠지만, 가치함수를 활용한다면 더 좋은 베이스라인이 될 수 있다.
  - 가치함수는 상태마다 값이 다르지만 행동마다 다르지는 않기 때문에 효율적으로 큐함수의 분산을 줄일 수 있다.
  - 따라서 Actor-Critic에서는 가치함수를 베이스라인으로 사용한다.
    가치함수 또한 근사해야 하는데 새로운 v라는 변수를 사용해서 근사할 수 있다.

Actor-Critic 이론

- Actor-Critic의 오류함수
  - 가치함수를 베이스라인으로 큐함수에서 뺀 것을 **어드밴티지(Advantage) 함수**라고 한다.

$$A(S_t, A_t) = Q_w(S_t, A_t) - V_v(S_t)$$

  - 위 함수는 큐함수와 베이스라인인 가치함수를 따로 근사하기 때문에 비효율적이다.
  - 따라서 큐함수를 가치함수를 사용해서 표현하면 가치함수만 근사해도 정의할 수 있다.
    가치함수만 근사해서 정의한 어드밴티지 함수는 다음과 같다.

$$\delta_v = R_{t+1} + \gamma V_v(S_{t+1}) - V_v(S_t)$$

  - 이제 어드밴티지 함수를 사용한 Actor-Critic의 업데이트 식을 다음과 같이 변형할 수 있다.

$$\theta_{t+1} \approx \theta_t + \alpha[\nabla_\theta \log \pi_\theta(a|s)\, \delta_v]$$

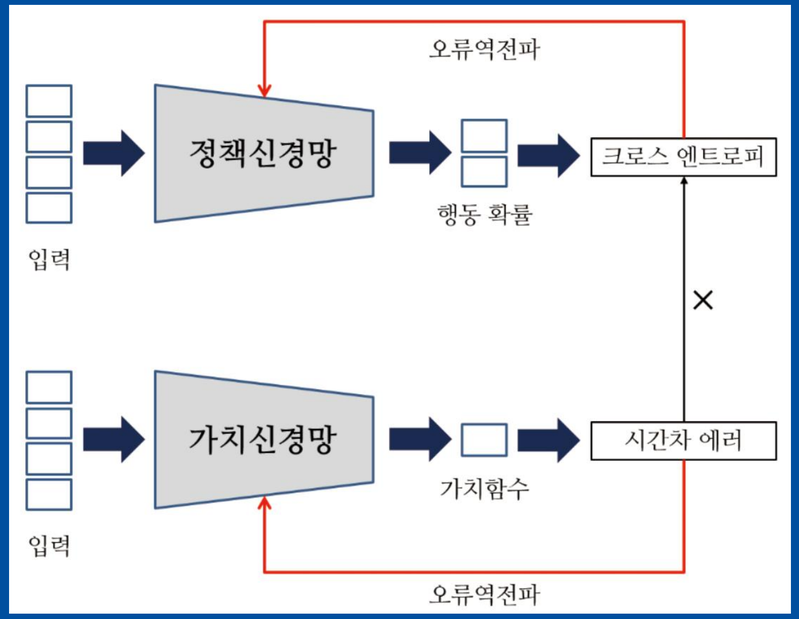
Actor-Critic 이론

- Actor-Critic의 오류함수
  - 가치신경망은 가치함수를 근사하는 인공신경망이므로 이 또한 학습이 필요하다.
  - 가치신경망의 학습은 시간차 오류를 통해 진행한다.
    현재 상태와 다음 상태, 그리고 보상을 통해 오류함수를 계산해서
    이 오류함수를 최소화하는 방향으로 업데이트를 진행한다.

$$\mathrm{MSE} = (\text{정답} - \text{예측})^2 = \left(\mathrm{R_{t+1}} + \gamma \mathrm{V_v}(\mathrm{S_{t+1}}) - \mathrm{V_v}(\mathrm{S_t})\right)^2$$

Actor-Critic 이론

- 정리
  - Actor-Critic은 2개의 신경망을 가지고 있다.
    정책신경망은 정책을 근사하며 가치신경망은 가치함수를 근사한다.
  - 카트폴의 입력인 상태는 4개의 원소를 가지며 행동은 2가지가 있으므로
    두 신경망의 구조는 다음과 같다.

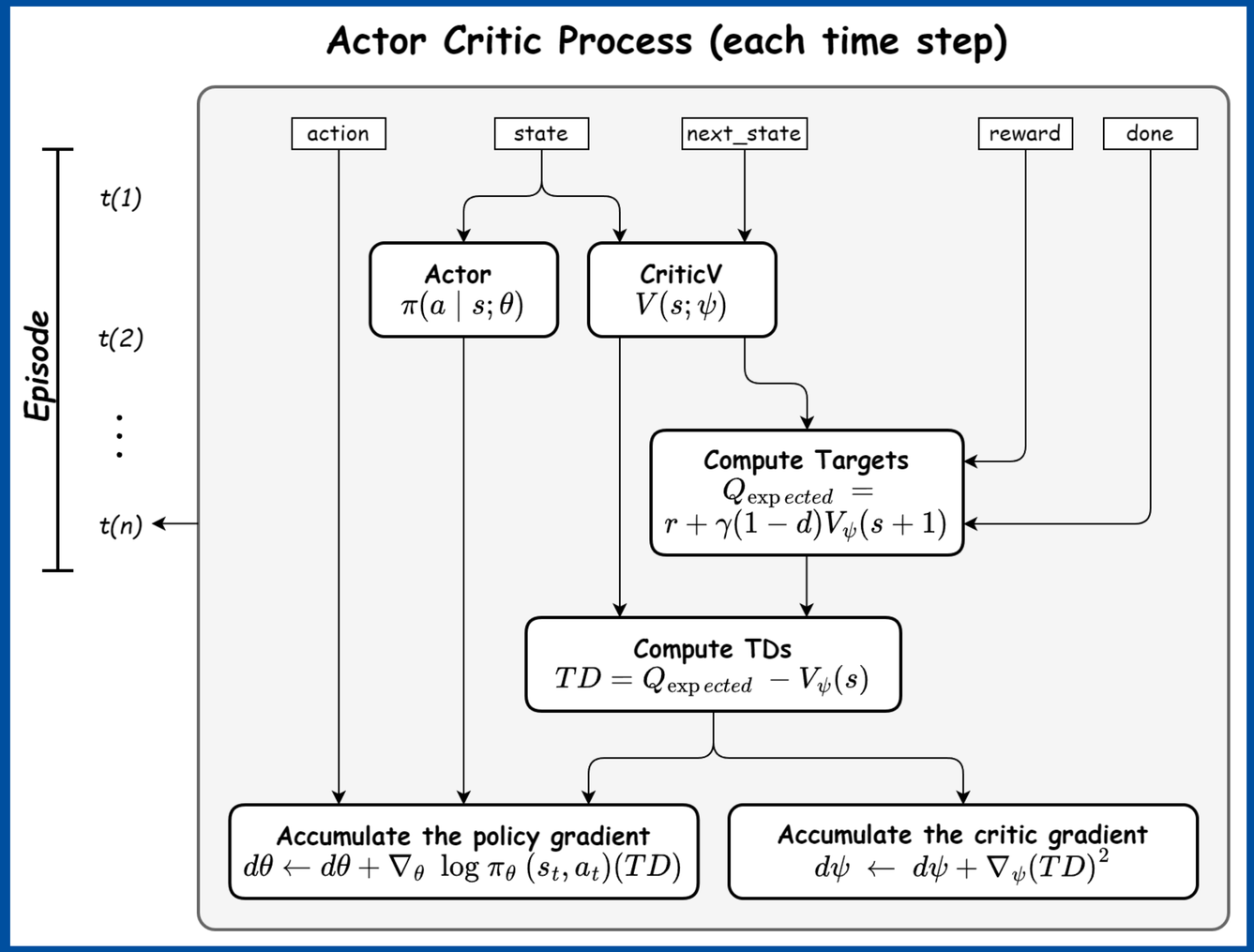


Actor-Critic 이론

- 정리
  - 정책신경망의 출력으로부터 크로스 엔트로피 오류함수를 계산할 수 있다.
  - 또한 가치신경망의 출력으로부터 시간차 오류를 계산할 수 있다.
  - 가치신경망은 시간차 오류를 가지고 딥살사에서와 같은 방식으로 신경망을 업데이트한다.
  - 정책신경망은 크로스 엔트로피 오류함수와 시간차 오류의 곱으로
    새로운 오류함수를 정의하고 이 오류함수로 정책신경망을 업데이트한다.
  - 이렇게 Actor-Critic이 어드밴티지를 사용하기 때문에 다른 이름으로는
    A2C(Advantage Actor-Critic)이라고도 한다.

Actor-Critic 이론

- 정리



감사합니다!

스터디 듣느라 고생 많았습니다.