Playing Atari With Deep Reinforcement Learning

Yeachan Heo

소개

Playing Atari With Deep Reinforcement(이하 DQN) 논문은 딥마인드에서 발표되었다. 당시 아주 조금의 전처리를 거친 게임 화면만으로 게임을 학습할 수 있어 큰 주목을 받았던 논문이다. Q함수에 대한 Temporal-Difference Error를 사용하는 Q러닝을 많이 개선한 버전이라고 생각하면 될 것 같다.

Quick Facts

1.DQN 알고리즘은 Off-Policy Algorithm이다
2.DQN 알고리즘은 Value-Base RL에 속한다
3.DQN 알고리즘은 넓은 observation space를 커버하기 위해 CNN(Convolution Neural Network) 를 사용한다
4.DQN 알고리즘은 Replay Buffer를 사용해 자신의 경험을 되돌아본다
5.DQN 알고리즘은 Q러닝에 기반해 있다
6.DQN 알고리즘은 Discrete한 Action Space에서만 동작한다

Exploration vs Exploitation

DQN 알고리즘은 기본적으로 $\epsilon - greedy$ 정책을 사용한다. ϵ의 확률로 랜덤한 액션 $a \in A$를 선택하고, 랜덤한 액션을 하지 않을 상황이라면 $\gamma max_a\ Q(s,a)$(상태 s에서의 Q함수의 값이 최대가 되는 행동 a, greedy action) 를 선택한다는 것이다. 본 논문에서는 ϵ를 바꾸지 않는 하이퍼파라미터로 고정하는 pseudo code를 작성해 놓은 것 같다. 실제 구현에서는 ϵ을 학습이 진행됨에 따라 낮추도록 하는 경우가 많다.

DQN Loss Function

DQN의 오차함수에 대해 알아보자. $(r + \gamma max_a Q(s',a') - Q(s,a))^2$ 에 대해 알아보자는 뜻이다. 눈치가 빠른 사람들은 알아봤을 것이다. target이 $r + \gamma max_a Q(s',a')$이고 prediction이 $max_a Q(s,a)$인 MSE(Mean Squared Loss) 이다. 지금부터 저 loss가 왜 유효한지에 대하여 알아보자. 먼저 뉴럴넷이 근사하는 대상이 무엇인지부터 확실히 할 필요가 있다. 뉴럴넷은 어떤 (s,a)에 대해 $Q(s,a)$를 근사한다. 즉, 어떤 상태와 행동에 대해 t~T(현재 시간부터 에피소드 종결) 까지의 기댓값을 근사한다. 그렇다면 target와 prediction을 다시 풀어서 적어보자. target는 $r_t + \sum_{t=t+1}^{T} p(r_t)$으로 다시 적을 수 있다. prediction은 $\sum_{t=t}^{T} p(r_t)$라고 적을 수 있다. 이제 두개의 차를 구해 보자. $r_t - p(r_t)$가 된다. 실제 보상- 예측 보상이 되니 mse가 유효하다. 이 오류를 최소화시키면 점점 $Q(s,a)$를 정확히 예측할 수 있고 보상을 많이 받을 행동이 무엇인지 알 수 있다.

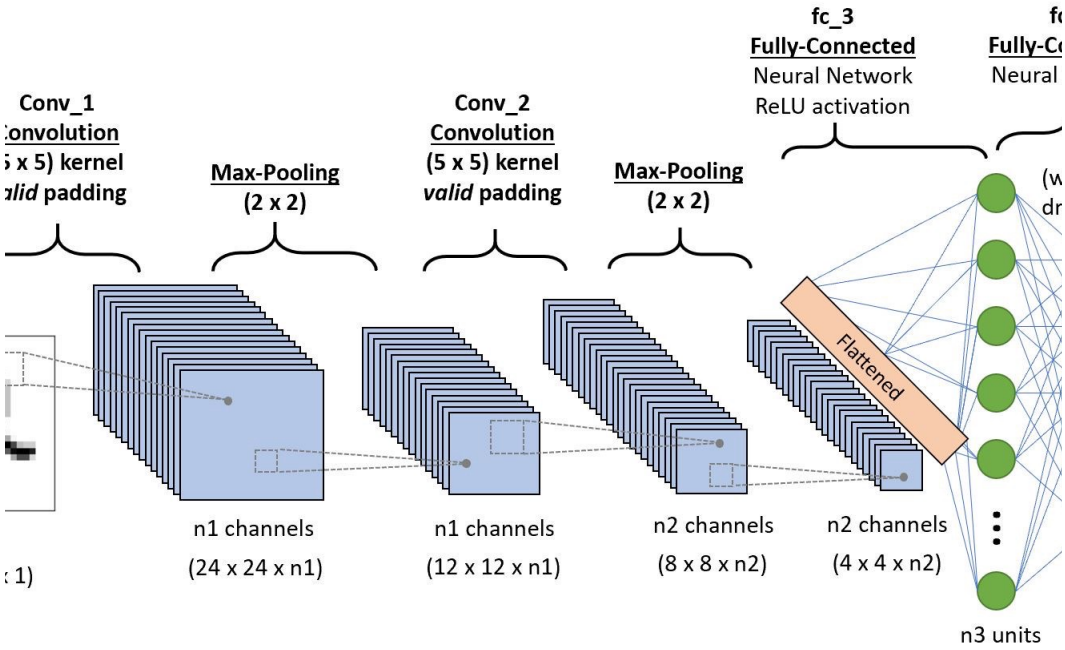
*$p(r_t)$는 보상의 예측값이라는 뜻이다.

Replay Buffer

DQN은 Replay Buffer라는 개념을 사용한다. 쉽게 말해서 (상태, 행동, 보상, 다음 상태) 를 저장해 놨다가 추후 Q함수를 업데이트 하는 데에 사용하는 것이다. 사람으로 비유해서 쉽게 말해보자. 상태로 썩은 우유를 가정하고, 행동은 그 우유를 먹는 것으로 생각한다. 배가 아픈 것은 분명히 좋지 않은 보상일 것이다. 이러한 것을 기억해 두었다가 다음에 같은 상태(썩은 우유) 가 나온다면 먹는 행동을 하지 않는 것과 같다고 보면 된다. 다시 본론으로 돌아가서 , Replay Buffer를 사용할 수 있는 이유는 DQN이 오프 폴리시 알고리즘이기 때문이다. DQN의 로스를 살펴보면 다음 행동 a'를 해본 다음 $Q(s', a')$을 구하는 것이 아니라 $max_{a'}Q(s', a')$을 사용한다. 즉, (상태, 행동, 보상, 다음 상태) 만 있어도 업데이트 할 수 있다. 그것이 언제의 것이든 상관없이 말이다. 비슷한 오차함수를 쓰지만 온폴리시 알고리즘인 SARSA의 오차함수를 살펴보자. $(r + \gamma Q(s', a') - Q(s, a))^2$ 인데, 실제로 다음 행동 a'을 해야 업데이트 할 수 있다. 그러므로 리플레이 버퍼를 사용할수 없다. $Q(s, a)$의 값은 Q함수를 근사하는 신경망이 업데이트 될 수록 달라지기 때문이다. 학습되지 않은 신경망과 학습된 신경망의 같은 입력에서의 출력은 당연히 다르다. 그러므로 같은 상태라도 신경망이 업데이트 됨에 따라 결정하는 행동이 달라질 수 있다. 따라서 실제 액션을 해본 다음 업데이트 타겟을 만드는 살사에는 Replay Buffer를 사용할 수가 없다.

컨볼루션 신경망(CNN) 도입

DQN 논문에서는 아타리 게임을 DQN 알고리즘으로 풀기 위해서 컨볼루션 신경망을 도입했다고 한다.



CNN, 합성곱 신경망의 구조

그림에서 볼 수 있듯이 합성곱 신경망은 사진 등의 특징을 추출하는 효과를 낸다고 한다. 아타리와 같이 화면을 observation 으로 가지고 있는 태스크의 경우 최적의 선택이 될 것 같다.

질문

1.DQN의 converge proof는 보지 못한 것 같다. 어떻게 증명할 수 있을까?
2.DQN 오차함수의 문제점에는 어떤 것들이 있을까?