강화학습, Q-Learning

이은후

강화학습 - 용어

[상태]
Agent: 주변 상태에 따라 어떤 행동을 할지 판단을 내리는 주체
Environment: 에이전트가 속한 환경

[동작]
Action: 에이전트가 취하는 행동
State: 행동에 따라 변화하는 상태
Reward: 행동으로 인한 보상

*Agent: 상황에서 행동하는 주체. (여러 에이전트를 동시에 다루는 문제는 Multi-agent 세팅이라고 부름)
*Policy: Agent의 판단 방식
*Environment: 문제가 세팅된 그 상황 자체. 즉 액션, state, 리워드 등이 모두 환경이 됨.
*State: 현 시점에서 상황을 설명하는 값들의 집합
*Action: 현 시점에서 agent가 취할 수 있는 행동들
*Reward: Agent가 취한 Action으로 인해 따라오는 보상/이득 (높을수록 좋음)
*Q-value: 취한 Action마다의 Quality value (얼마나 좋은지를 측정하는 값)

강화학습 - Markov Decision Process (MDP)

- 강화학습은 '마르코프 결정 과정'을 따름.
  마르코프 결정 과정의 핵심 문제 = 의사결정자(agent)의 의사 결정 정책(policy)인 π를 구하는 것

- π = 현재 state에서 agent가 취할 action을 지정하는 함수
  목표는 보상(reward)의 누적합을 최대화 시키는 정책 π를 구하는 것!

강화학습 - 가치함수

- 가치함수
  : reward는 단기적인 보상(이득)만을 나타냄.
  agent가 취하는 각각의 action이 얼마나 가치 있는지를 판단하려면 누적 보상값을 예측할 수 있어야 하며,
  이를 수학적으로 정의한 것이 가치함수

0.787	1.45	2.291
1.45	2.844	5.15
2.291	5.15	7.186

  가치함수는 모든 상태공간을 가치로 치환함.
  → 가치함수를 구한 다음, 각 state에 대한 가치를 비교해서 값이 높은 곳으로 움직이는 action을 취함.

강화학습 - 가치함수

- 가치함수의 정의 – 1) state-value 함수 (상태 가치 함수)
  2) action-value 함수 (상태-행동 가치 함수, Q-function)

- 1) state-value 함수 (상태 가치 함수)
  : state에만 의존하여 value를 평가하는 방식.
  agent가 이 함수를 통해 다음으로 갈 수 있는 state들의 가치를 보고 높은 가치를 가진 state로 이동.

  즉, 현재 상태 s에서 정책 π를 따랐을 때의 가치를 반환

$$G_t = R_{t+1} + \gamma R_{t+2} + r^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

$$v(s) = \mathrm{E}[\ G_t \mid S_t = s\ ]$$

강화학습 - 정리

- ‘강화학습’ 이란?

  : 달성하고자 하는 목표에 대해 각 state에서 action에 대한 최적의 value를 학습하는 것.
  이는 현재 state에서 미래 보상의 합을 최대화 하게끔 학습하면서 이루어짐.
  즉, 모든 state에 대해서 각 value function을 최대로 하는 action을 선택하도록 하는 것

Q-Learning

- Q-Learning?
  : MDP(마르코프 결정과정)모델 없이 하는 강화 학습 기법 중 하나 (Model-Free)

- Model-Free 알고리즘
  : 환경에 대해 알지 못하고 있는 상태에서 수행하는 방식.
  MDP 모델을 미리 알 수 없기 때문에 '예측'과 '제어'를 반복하여 학습한 가치함수를 기반으로 정책을 학습함.

  (Model based: 환경에 대한 모든 설명을 알고 수행하는 방식.
  직접 action을 취하기 전에, 가능한 미래 상황들을 고려해서 다음 상태와 보상을 예측함.
  MDP에서 상태 전이 확률과 보상함수를 미리 정해서 할 수 있는 경우를 말함)

- 앞서 말한 state-value함수는 state만을 고려하였지만,
  Q함수(action-value함수)는 state에 action까지 고려하여 결정됨.

Q-Learning - 가치함수

- [action-value 함수]
  : 현재 상태 s에서 정책 π를 따라 행동 a를 수행했을 때의 가치 반환

  방정식: $Q(s,a) \simeq R+\gamma \max Q(s',a')$
  (s: state, a:action)

  현재 state의 action에 대한 Q-value =
  당장의 reward + 다음 state가 가질 수 있는 max Q-value값에 discount factor(γ)를 적용한 것.
  이 과정을 무수히 반복하면 결국 최선의 policy(π)을 얻을 수 있음

*discount factor: 당장이 아니라 먼 미래에 더 좋은 결과를 가져오는 action을 선택하게 하기 위한 요소.
즉, 현재 state가 받는 reward = 당장 받을 수 있는 reward+미래 reward들의 합
(현재 action을 통해 받는 reward + 다음 state에서 받을 reward*discount factor …)

미래의 보상은 현재 보상보다 가치가 낮기 때문에, 보상에 미래 시점의 time step만큼 곱해주는 것. 0.0~1.0사이의 값으로 나타남

Q-Learning – 가치함수

- $Q(s,a) \simeq R+\gamma \max Q(s',a')$
  → 목표는 Q(s,a)값이 $R+\gamma \max Q(s',a')$에 가까워지도록 하는 것임
  그러기 위해선 Q(s,a)를 계속 업데이트 해줘야 함

- 이용(Exploitation)과 탐험(Exploration)
  : 이용은 현재 알고있는 한 가장 최적의 행동을 선택하는 것(greedy),
  탐험은 다양한 경험을 쌓기 위해 아무 행동을 선택해 보는 것

  Agent는 기본적으로 Q값이 높은 action을 선택하는데, 이로 인해 충분히 탐색하지 않은 채 '적당해 좋은' 해결책을 찾아내면 그것에 수렴해버리는 문제가 발생함.
  → $\epsilon(epsilon)$ – greedy 탐색법 사용
  : ϵ 와 0.0~1.0사이 랜덤값을 비교하여 ϵ이 더 크면 무작위행동 반환, 아니면 Q함수에 따라 움직임.
  (ϵ 은 보통 탐색이 진행됨에 따라 값을 서서히 낮춤. (epsilon decay))

Q-Learning

- Q-function 단점
  : 특정 조건에서 action-value를 overestimate(과평가)하는 경향이 있음.

  즉, Q-Learning은 reward에 따라 다음 action이 결정된다고 할 수 있는데,
  잘못 선택한 action에 대한 value가 과평가 된다면 잘 된 방향으로 학습이 진행될 수 있다는 것임
  → max Q-value만을 선택함에 따라 overestimate 문제가 발생하는 것.

  Q-Learning이 좋은 성능을 보이지 못한 경우가 있다면 overestimate때문.
  이를 보완하기 위해 나온 것이 Double Q Network (DQN)
  (DQN은 많은 양의 데이터를 효과적으로 저장하고, 효율적인 함수 출력으로 바꾸는 신경망 네트워크를 쓰는 방법임)