1 Q-러닝 설명

1.1 Q-러닝 개요

목표: Q-러닝의 목표는 시간에 따른 누적 보상을 최대화하는 최적의 정책(즉, 상태에서 행동으로의 매핑)을 찾는 것입니다.

1.2 수학적 기초

1. **Q-함수 (행동-가치 함수):** Q-함수 $Q(s, a)$는 상태 s에서 행동 a를 취하고 그 이후에 최적의 정책을 따를 때 기대되는 누적 보상을 나타냅니다.

2. **벨만 방정식:** Q-값은 벨만 방정식을 기반으로 업데이트되며, 이는 Q-값의 재귀적 분해를 제공합니다:

$$Q(s, a) = \mathbb{E}\left[r + \gamma \max_{a'} Q(s', a') \mid s, a\right]$$

   여기서:

   - s: 현재 상태
   - a: 현재 행동
   - r: 상태 s에서 행동 a를 취한 후 받은 보상
   - s': 다음 상태
   - a': 다음 행동
   - γ: 할인 인자 (미래 보상에 비해 즉시 보상을 얼마나 가치 있게 여기는가)

1.3 Q-러닝 알고리즘

1. **초기화:**

   - 상태가 행을, 행동이 열을 나타내는 Q-테이블을 0 (또는 작은 임의의 값)으로 초기화합니다.
   - 하이퍼파라미터 설정: 학습률 (α), 할인 인자 (γ), 탐험률 (ϵ).

2. **행동 선택 (탐험-활용 트레이드오프):**

   - 엡실론-탐욕 정책을 사용하여 행동을 선택합니다:
     - 확률 ϵ로 임의의 행동을 선택합니다 (탐험).
     - 확률 $1 - \epsilon$로 현재 상태에서 가장 높은 Q-값을 가진 행동을 선택합니다 (활용).

3. **Q-값 업데이트:**

   - 선택한 행동을 실행하고, 보상과 다음 상태를 관찰합니다.

- 벨만 방정식을 사용하여 Q-값을 업데이트합니다:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left(r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right)$$

여기서 α는 학습률이고, γ는 할인 인자입니다.

4. **엡실론 업데이트:**
   - 시간에 따라 탐험률 ϵ을 점진적으로 감소시켜 탐험을 줄입니다.

1.4 상세 단계

1. **초기화:**
   - **q_table:** 각 셀이 $Q(s,a)$인 2D 배열로 초기화는 0으로 합니다. 테이블의 크기는 state_size x action_size입니다.
   - α: 새로운 정보가 오래된 정보를 얼마나 덮어쓰는지를 결정하는 학습률.
   - γ: 미래 보상의 중요도를 결정하는 할인 인자.
   - ϵ: 에이전트가 환경을 탐험하도록 초기 탐험률을 설정.
2. **행동 선택:**
   - act 메소드는 엡실론-탐욕 전략을 사용하여 탐험 또는 활용을 결정합니다.
   - 임의의 숫자가 ϵ보다 작으면 임의의 행동을 선택합니다 (탐험).
   - 그렇지 않으면 현재 상태에서 가장 높은 Q-값을 가진 행동을 선택합니다 (활용).
3. **Q-값 업데이트:**
   - update_q_value에서 에이전트는 벨만 방정식을 사용하여 Q-테이블을 업데이트합니다.
   - **best_next_action**은 다음 상태에서 취할 최적의 행동을 식별합니다.
   - **td_target**은 즉시 보상과 할인된 미래 보상을 고려한 목표 Q-값입니다.
   - **td_error**는 목표 Q-값과 현재 Q-값의 차이입니다.
   - Q-값은 학습률 α에 따라 td_target을 향해 업데이트됩니다.
4. **엡실론 감쇠:**
   - 업데이트 후 탐험률 ϵ을 감소시켜 탐험에서 활용으로 점진적으로 전환합니다.

1.5 계산 설명

- **탐험 대 활용:** 에이전트가 충분히 환경을 탐험하도록 초기 탐험률을 설정하고, ϵ이 감소함에 따라 학습된 정책을 활용합니다.
- **Q-값 업데이트:** 벨만 방정식을 사용하여 최적의 Q-값 추정을 점진적으로 개선합니다.
- **할인 인자 (γ):** 즉시 보상과 미래 보상의 중요도를 조정합니다.
- **학습률 (α):** 새로운 정보에 기반하여 Q-값이 얼마나 빠르게 업데이트되는지를 조절합니다.

이 원칙을 따름으로써 Q-러닝 알고리즘은 에이전트가 환경과 상호 작용을 통해 최적의 정책을 학습하고, 누적 보상을 최대화하기 위해 의사 결정을 점진적으로 개선할 수 있게 합니다.