



DQN 과대평가 완화 논문 재연

- LunarLander-v3환경 기반으로

과 목 명: 강화학습의 기초
담 당 교 수: 소정민 교수님
제 출 일: 2025.12.07
학 과: 데이터사이언스·인공지능
학 팀 명: 42조
성 명: A70080 이채원
GitHub : [RL-DDQN-Comparison\(click\)](#)

개요

본 프로젝트는 “Deep Reinforcement Learning with Double Q-learning” 의 아이디어를 기반으로 원 논문의 Atari + CNN 설정 대신 LunarLander-v3 환경에 Double DQN을 적용하여 과대평가 완화 효과를 실험적으로 재현하였다.

INDEX

01. 프로젝트 주제 및 목표
02. 논문 요약
03. 환경 및 데이터셋
04. State, Action, Reward
05. 알고리즘과 Hyperparameter
06. 실험 셋업
07. 실험 결과
08. 토의 및 결론

1. 프로젝트 주제 및 목표

1.1 프로젝트 주제 및 목표

◆ 프로젝트 주제

- Deep Reinforcement Learning with Double Q-learning 논문 재연으로 Double DQN의 Q-learning의 과대평가 문제의 완화 분석.

◆ 프로젝트 목표

- LunarLander-v3 환경에서 DQN vs DDQN 성능 비교
- Target network 업데이트 주기의 효과 분석
- 여러 random seed(10, 20, 30)를 통한 신뢰도 평가 및 신뢰구간 시각화

2. 논문 요약

2.1 논문 정보

◆ 제목

- Deep Reinforcement Learning with Double Q-learning

◆ 저자

- Hado van Hasselt, Arthur Guez, David Silver (Google DeepMind)

◆ 발표

- AAAI

2.2 논문 요약

◆ 연구 배경 및 문제 정의

- Q-Learning과 DQN 알고리즘은 특정 조건에서 실제보다 value를 높게 평가한다.
- 이 문제는 Target Q-value 계산 시 max연산자를 사용하여 오차가 존재할 때도 최대값을 선택하는 것에서 비롯된다.
- 과대평가는 학습을 불안정하게 만들고 최적의 결과를 학습하는데 방해가 된다.

◆ 제안 방법

- DDQN(Double DQN)으로 행동의 선택과 행동의 가치 평가를 분리하여 과대평가를 방지한다.
- 기존 DQN을 그대로 사용하면서 Target의 선택된 행동의 가치를 평가하도록 식을 변경한다.

◆ 실험 결과

- DQN과 비교하여 가치추정의 정확도가 높아지고, 과대평가가 감소함을 입증하였다.
- Atari 2600 게임에서 DDQN이 DQN보다 더 안정적이고 높은 점수를 보였다.

3. 환경 및 데이터셋

3.1 환경 및 데이터셋

◆환경 및 데이터셋의 변경

- 본 프로젝트에서는 원 논문과는 다른 환경과 상태 표현을 사용하였다. 원 논문은 Atari 2600 게임 화면 픽셀 이미지($84 \times 84 \times 4$)를 입력으로 하는 CNN 기반 Q-network를 사용하였으나 본 프로젝트에서는 아래와 같은 이유로 다른 데이터셋을 사용하였다.
 1. DDQN 알고리즘 로직 검증에 집중
 2. 이미지 전처리와 CNN 설계에 따른 복잡도 배제
 3. 제한된 시간과 GPU 자원 내에서 안정적 실험 수행
- 본 프로젝트에서는 알고리즘의 제어 성능을 객관적으로 검증하기 위하여 강화학습에서 표준 벤치마크 환경인 Gymnasium LunarLander-v3을 활용하였다.

◆환경 및 데이터셋 비교

구분	원 논문	본 프로젝트
환경	Atari 2600 게임	Gymnasium LunarLander-v3
입력	게임화면 픽셀 이미지 ($84 \times 84 \times 4$)	8차원 연속 상태 벡터
모델	CNN 기반 Q-network	MLP 기반 Q-network

4. State, Action, Reward

4.1 State, Action, Reward 설계

◆ 설계

- State, Action, Reward 설계는 Gymnasium LunarLander-v3 벤치마크 환경에 내장된 표준 설계 수치 채택

◆ State, Action, Reward

State	Action	Reward
<ul style="list-style-type: none"> - Launder의 x,y 위치 - x, y의 속도 - 착륙선의 각도 - 왼쪽/오른쪽 발이 지면에 닿는 여부 (Boolean) 	<ul style="list-style-type: none"> - Do nothing: 0 - Left engine: 2 - Right engine: 3 - Main engine: 1 	<ul style="list-style-type: none"> - 안전한 착륙: 100~140 - 추락: -100 - 엔진가동: -0.3/step - 에피소드 종료

5. 알고리즘과 Hyperparameter

5.1 DQN과 DDQN

◆ DQN

$$Y_t^{\text{DQN}} \equiv R_{t+1} + \gamma \max_a Q(S_{t+1}, a; \theta_t^-).$$

- Target Q-value 계산 시 max 연산자로 추정 오차가 큰 값을 지속적으로 선택하여 과대평가 유발

◆ DDQN

$$Y_t^{\text{DoubleDQN}} \equiv R_{t+1} + \gamma Q(S_{t+1}, \operatorname{argmax}_a Q(S_{t+1}, a; \theta_t), \theta_t^-).$$

- 행동의 선택은 online network(θ)로 가치의 평가는 Target network(θ^-)로 분리하여 과대평가를 감소

5.2 Hyperparameter 설정

◆ Hyperparameter

- 원 논문의 전략을 준수하여 알고리즘의 본질을 유지하되 실험 환경 변경으로 학습률과 배치사이즈 등 조정
- 업데이트 주기 변경을 추가하여 DDQN의 성능 변화 측정

항목	값	의미
학습률	5e-4	가중치의 조정의 크기
할인율	0.99	원 논문과 같이 장기보상 고려
배치크기	64	한 번의 업데이트에 사용하는 transition. 계산의 효율성을 위하여 원 논문보다 크게 설정
Replay Buffer	50000	과거의 경험 저장 개수이며 이미지보다 적은 메모리를 차지하므로 메모리 용량과 속도를 위해 축소
Target Update 주기	100, 1000, 10000	원 논문에서는 10000으로 고정되어 있지만 주기를 3가지로 나누어 DDQN의 안정성 확인
ϵ -greedy	1 ~ 0.01	무작위를 최대화에서 학습이 진행될 수록 단순하게 진행되도록 설정
Seeds	10, 20, 30	단일 시드로 인한 우연성 의존도를 낮추기 위한 설정

6. 실험 셋업

6.1 실험 셋업

◆ 실험 환경

- Google Colab(T4 GPU), PyTorch Framework

◆ 실험 횟수

- 각 설정마다 500 에피소드
- 3개의 seeds

◆ Evaluation Metric

- 하나의 에피소드의 총 reward
- 최근 100 에피소드 평균 보상
- 성공 기준: 평균 200 점 이상

◆ 실험 구성

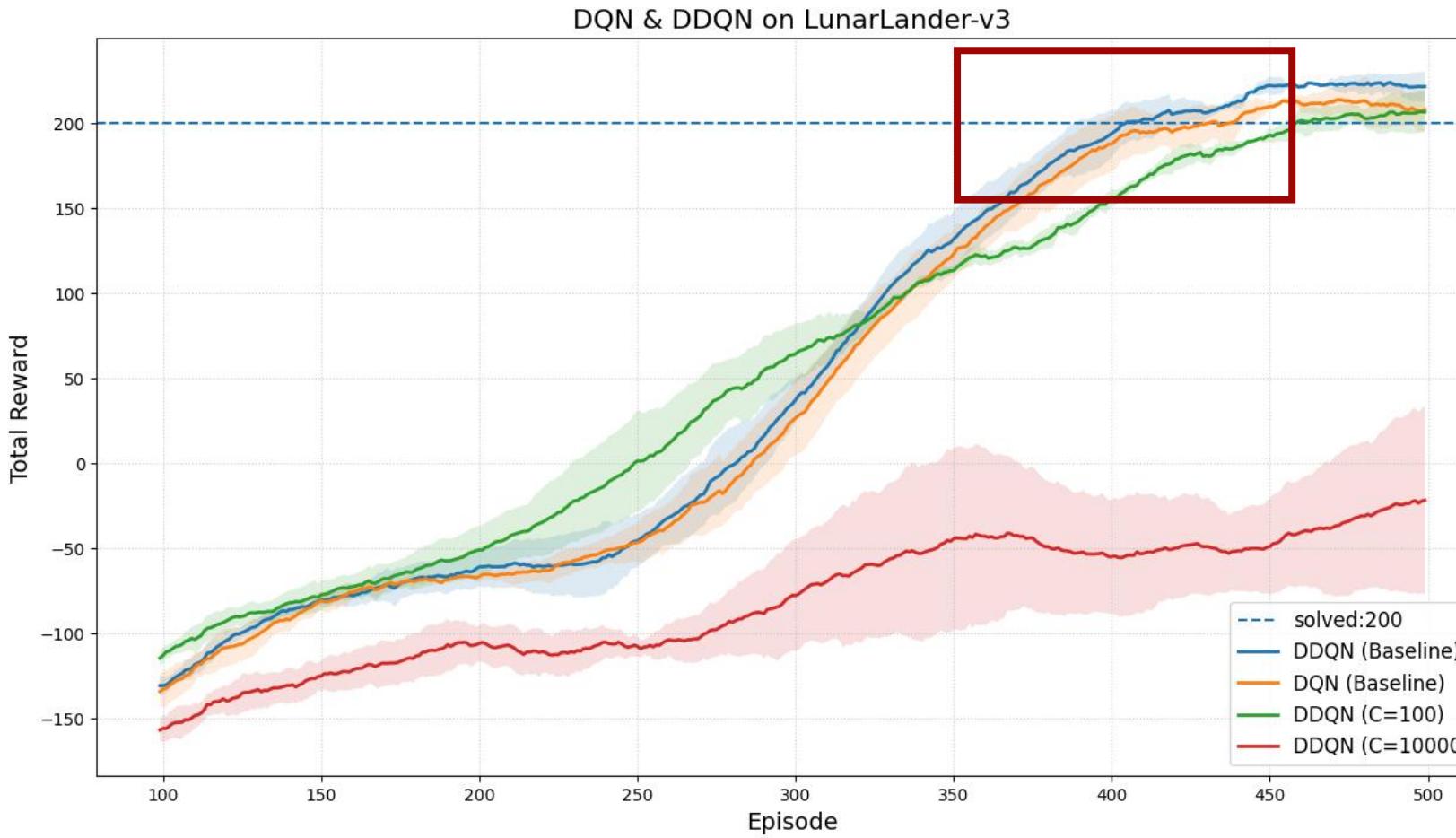
- Baseline 비교: 업데이트 주가가 1000 인 경우

◆ 신뢰도 실험

- 각 episode index별 수평으로 평균/표준편차를 그림으로 표현

7. 실험 결과

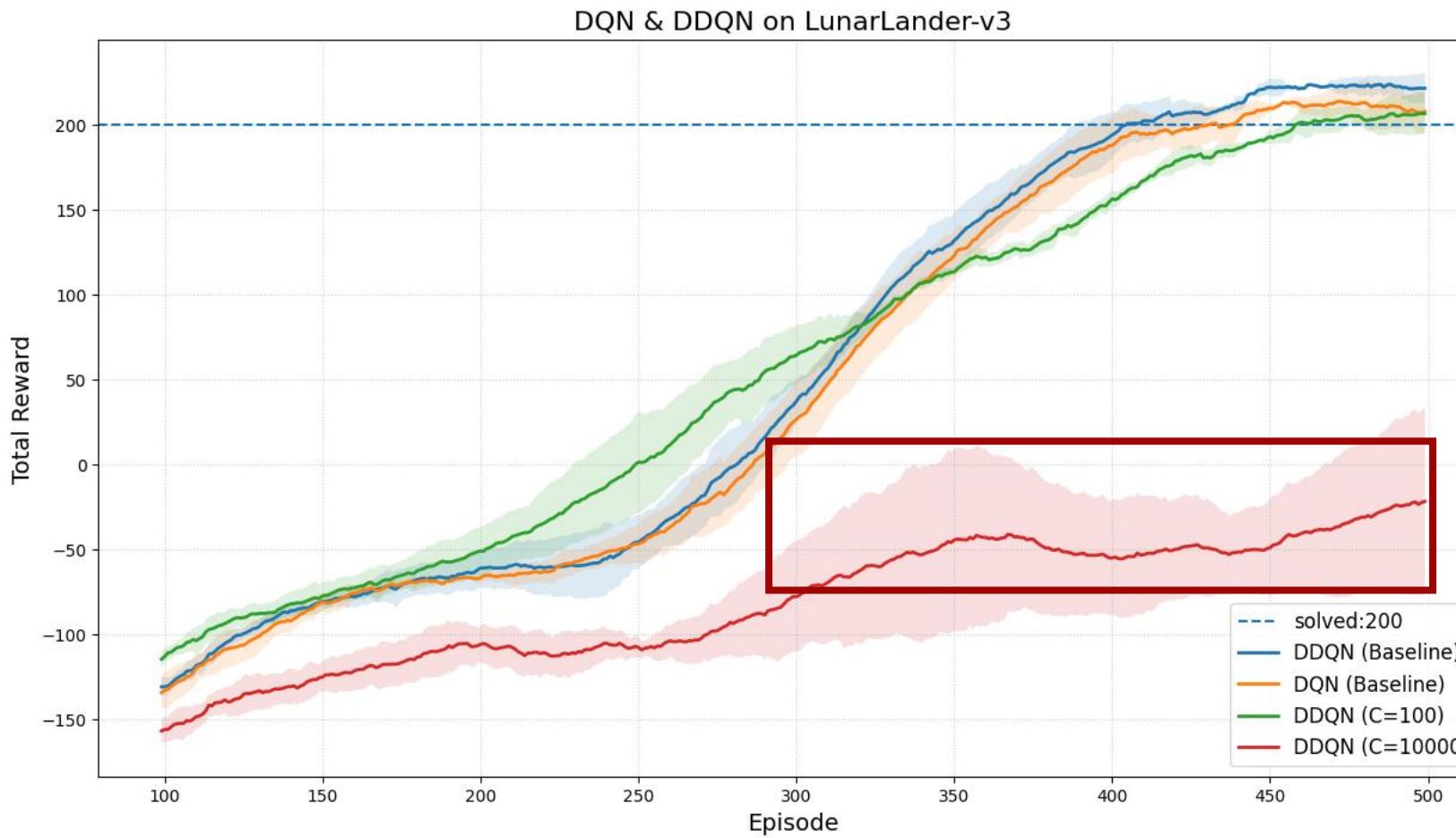
7.1 알고리즘 성능 비교



◆ 알고리즘의 성능 비교

- DDQN이 DQN보다 더 높은 최종 점수에 도달했다. 또한 DDQN의 신뢰 구간인 음영 영역이 좁으므로 학습이 안정적이고 과대평가 경향이 감소했다고 볼 수 있다. 그러나 원 논문과 비교하여서는 그 차이가 상대적으로 작게 나타난다.

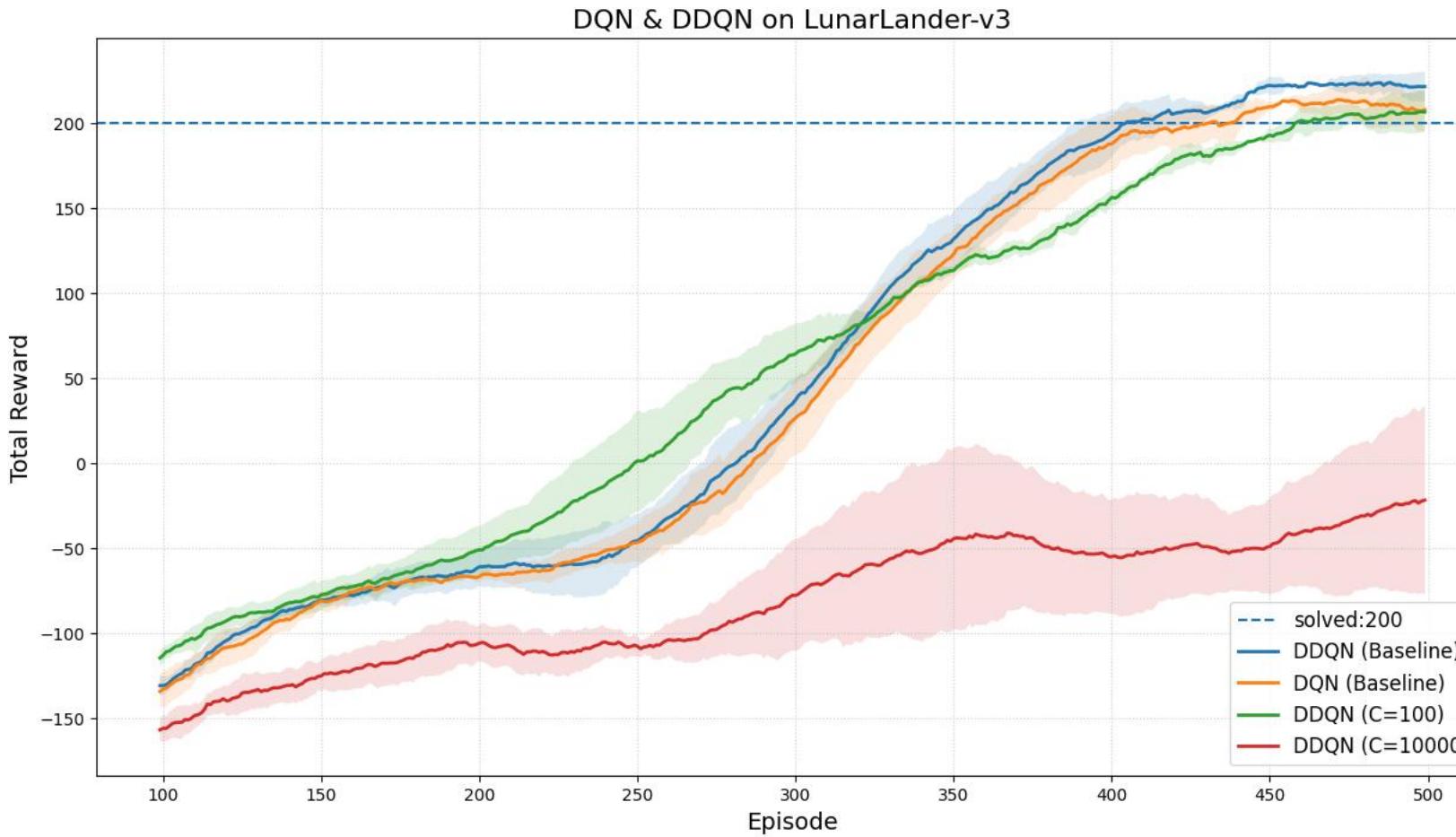
7.2 Target 업데이트 주기 변화 영향



◆ Target 업데이트 주기 변화

- 1000을 기준으로 target 갱신 주기를 더 빠르게 혹은 느리게 설정하였다.
- 너무 큰 주기인 10000은 target의 업데이트가 거의 없고 평균 보상이 200에 도달하지 못했다.

7.3 신뢰도 평가



◆ 신뢰도 평가

- 동일한 설정에서 다른 랜덤 방식으로 반복학습을 수행하여 신뢰도 구간을 표현하였다.
- 그래프에서 각 선 주변의 반투명 영역은 여러 seed에 대한 결과의 분산을 나타낸다.
- 이 영역이 좁을수록 seed 간의 성능이 비슷하다는 것을 의미함으로 결과가 안정적이고 신뢰도가 높은 것으로 해석할 수 있다.

7.3 신뢰도 평가

◆ 결과 3. 신뢰도 평가

- 동일한 설정에서 다른 랜덤 방식으로 반복학습을 수행하여 신뢰도 구간을 표현하였다.

8. 토의 및 결론

8.1 토의 및 결론

◆ 토의

- LunarLander-v3는 상태 차원이 낮고 비교적 간단한 환경이기 때문에 원 논문의 환경보다는 과대평가 문제가 덜한 것일 수 있다. 그러나 논문의 아이디어를 다른 환경에 적용해서 그 결과를 도출해낼 수 있었다는 점에서 의의가 있다.
- DDQN이 항상 우수한 결론을 나타내는 것이 아니고, Target 업데이트 주기가 너무 크면 학습이 어렵기 때문에 적절한 조정이 필요할 것이다.
-

◆ 결론

- LunarLander-v3 환경에서 DQN vs DDQN 모두 안정적으로 문제 해결
- Target network 업데이트 주기 빈도에 따라 성능과 안정성 확인

8.2 한계 및 향후연구

◆한계

- 픽셀 기반의 Atari 환경이나 CNN구조를 구현 못함
- 비교적 단순한 데이터를 활용하여 일반화에는 한계 존재
- 이미지

◆향후 연구

- 비교적 단순한 이미지 환경으로 실험 확장
- Dueling DQN 등의 추가적인 기법을 통한 비교

참조문헌

참조문헌

감사합니다
