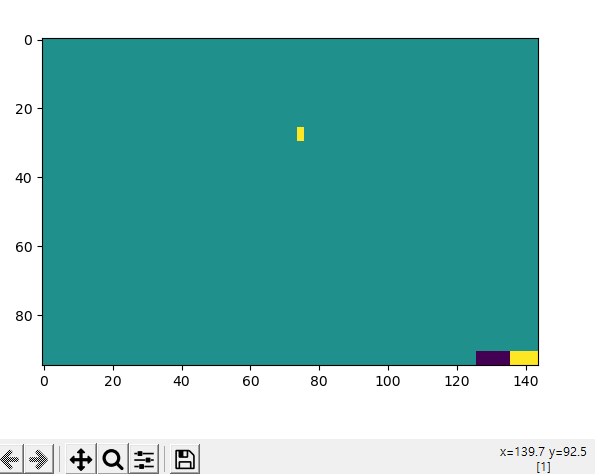
강화학습 기말과제 Breakout

소프트웨어학과 201920723 김동욱

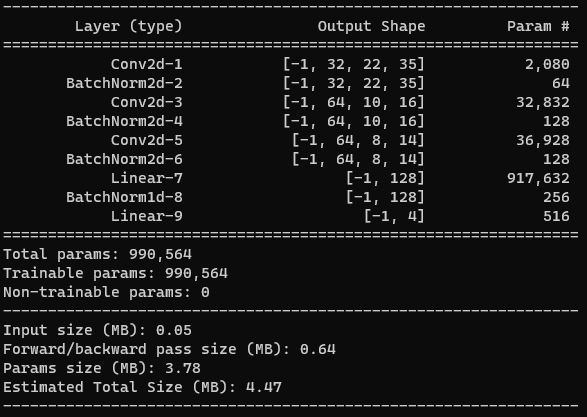
핵심 아이디어

1. 수업시간에 배운 알고리즘 중 DQN 기반 학습. AC기반의 경우 off-policy 문제로 학습이 어렵고, REINFORCE 기반의 방법도 value based on policy인 DQN보다 성능이 낮았다.
2. 단순 프레임을 입력으로 사용하지 않고 현 프레임과 이전 프레임의 차이를 통해 현 프레임 값은 1, 이전 프레임은 -1로 만들어 프레임의 변화량(ball과 bar의 움직임)을 표현하였다.
3. 또한 전체 프레임을 입력으로 사용하지 않고 block 부분을 전혀 사용하지 않고 block하단의 blank space와 그곳에 있는 ball, bar만 입력으로 사용하였다. 그 이유로 agent가 사용하는 딥러닝 모델에 convolution block을 사용하였는데 이는 데이터 패턴을 인식하는데 강력한 모듈로 단순 크기만 차이 있는 block 무리와 bar, ball의 차이를 구분하는 것이 의미 없어 보였기 때문이고 또한 실제 실험 결과 전체 이미지를 사용한 모델보다 성능이 훨씬 좋게 나왔다.



3의 입력 데이터 시각화. 청색은 0, 노란색은 1, 보라색은 -1을 나타내며 상단의 block들을 input으로 사용하지 않고 있다.

1. 수업에서 배운 강화학습 알고리즘을 최대한 활용하기 위해 agent에서 사용된 딥러닝 모델 설계 부분에서 다양한 기술없이 기초적인 방법과 적은 수의 파라미터를 사용하였다.



1. 좀 더 쉽고 explicit한 학습을 위해서 나만의 reward 함수를 추가하여 학습하였다. 추가된 reward로는 bar와 ball이 같은 선상에 있을 경우 가상의 점수인 fake\_reward를 추가로 받는다. 이 방법의 경우 bar가 ball을 따라가게 되어 최적 정책을 학습할 경우 절대 공을 떨어뜨리지 않을 것이다.

코드 설명(코드에도 주석을 작성했습니다)

1~12: 라이브러리 호출

14~37: 하이퍼 파라미터 설정. Load 여부만 바뀌고 나머지는 학습 당시와 일치.

40~42: save, load 경로 설정

48~62: 프레임의 변화량 계산 및 입력 자르기 등 state에 사용 될 input pre-process 함수

66~91: 간단한 convolution과 Fully connected layer만 사용한 DQN함수

94~163: DQN함수를 사용하는 DQNAgent함수로, get\_action, append\_sample, train\_model, save, summary write을 수행한다.

167~183: Main 함수 시작과 Env, agent, 학습에 사용될 values initialization

185: train start

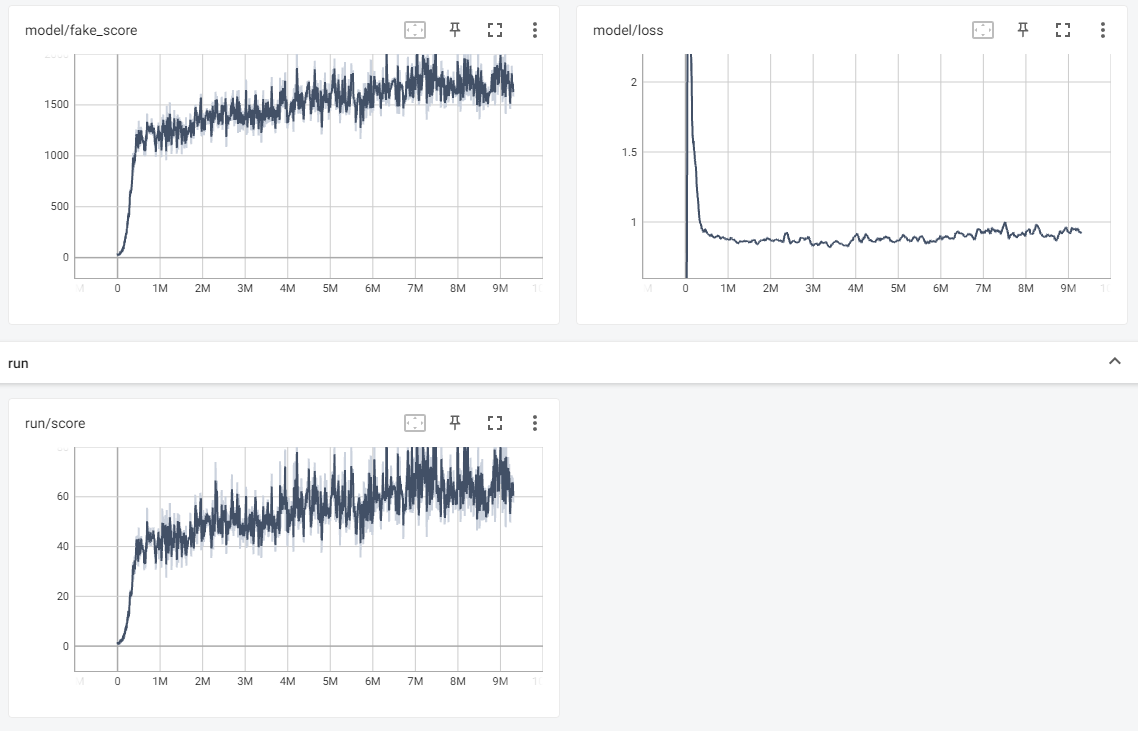
186~198: env, state, action 계산

200~208: real\_reward(score)와 fake\_reward(score) 계산

210~216: memory append, loss 계산

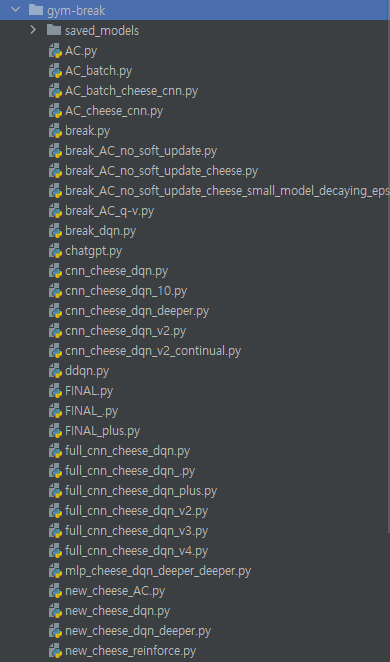
218~245: 게임 끝날 경우 초기화 및 모델, 데이터 저장

학습 텐서보드 로그



추가:

Batch 학습을 제외한 병렬 학습이 없어 적고 빠른 연산을 하지만 결국 학습이 오래 걸려 optimal 정책에 도달하지 못하였다. 하지만 병렬 학습을 구현하거나 계속해서 학습할 경우 텐서보드에서 볼 수 있듯 최적 정책에 수렴할 것이다. 또한 입력 state를 하단의 절반정도만 사용하여 게임 후반에 많은 block을 부수고 ball이 agent가 볼 수 있는 state밖으로 나가게 되어 학습이 정체되는 것으로 추측된다. 그럼에도 학습 성능은 여전히 지속적으로 좋아지긴 하였다.



도전했던 다른 모델들