휴리스틱 전략을 이용한 Q 러닝의 학습 간단화

박종철*, 김현철**
고려대학교 컴퓨터학과
e-mail : .grega87@naver.com, harrykim@korea.ac.kr

Simple Q-learning using heuristic strategies

Jong-cheol Park *, Hyeon-cheol Kim**
Dept. of Computer Science, Korea University

요 약

강화학습은 게임의 인공지능을 대체할 수 있는 수단이지만 불완전한 게임에서 학습하기 힘들다. 학습하기 복잡한 불완전한 카드게임에서 휴리스틱한 전략을 만들고 비슷한 상태끼리 묶으면서 학습의 복잡성을 낮추었다. 인공신경망 없이 Q-러닝만으로 게임을 5 만판을 통해서 상태에 따른 전략을 학습하였다. 그 결과 동일한 전략만을 사용하는 대결보다 승률이 높게 나왔고, 다양한 상태에서 다른 전략을 선택하는 것을 관찰하였다.

1. 서론

최근 강화학습은 게임의 인공지능으로 대체할 수 있는 주요 수단으로 부각되고 있다. 이미 Alphazero 에 의해서 체스, 바둑, 쇼기는 정복되었다[1]. 앞과 같은 완전 정보 게임들에 강화학습의 성능은 이미 인간을 뛰어넘었지만, 불완전 게임에서는 현재 진행형이다. 불완전 게임들은 대부분 강화학습으로 훈련하기에 상태와 행동의 복잡도가 높아서 적용하기 힘들다. 예를 들어 카드게임과 같은 장르는 매번 다른 상태에서 수많은 선택지들이 존재한다. 카드게임이 어려워질수록 나올 수 있는 경우의 수가 계속해서 커지고, 많은 선택지들 속에서 가장 최적의 수를 학습한다는 것은 힘들다. 이 논문에서는 휴리스틱 전략과 Q 러닝을 사용하여 전략을 구사할 수 있는 인공지능을 제안한다.

2. 관련 연구

강화학습은 어떠한 환경속에서 에이전트가 환경을 인식하고 선택지들 중에서 보상을 최대화하는 정책을 학습시키는 과정이다. 환경과 상호작용을 통해서 학습하며 주로 게임 분야에 적용됐었다. Q 러닝은 강화학습의 기법 중 하나로 주어진 상태에서 가능한 행동들 중에 기대값을 예측하는 Q 함수를 학습을 통해서 정책을 찾는다.

불완전 게임은 최소한 한명의 플레이어가 다른 사람의 선택에 대한 정보가 확실치 않을 때이다[2]. 상대방의 선택에 대한 정보가 없거나 플레이어에게 감추어진 정보가 있을 경우 모두 해당한다. 가장 간단한 게임의 예로는 포커가 있고, 상대방의 패를 보지 못하기 때문에 불완전한 게임이다. 인공지능은 상황에 맞는 적절한 수를 계산할 수 있어도, 상대방의 플레이가 허세인지 아닌지를 학습하는데 오래 걸린다. 최근에 유행하는 다양한 카드 전략 게임들은 기본적으로 불완전 게임들이다. 게임의 상태는 주어진 카드들과 덱에 따라서 무수히 많아진다. 이보다 더 복잡한 게임들을 학습하는데 쓰는 주된 방법은 심층 강화학습이다. 인공신경망을 이용해서 상태 정보를 입력으로 받고, 출력으로 행동을 받아온다. 처음에 Atari 와 같은 고전게임들에 적용하기[3] 시작해서 점점 어려운 게임들에 도전하고 있다.

3. 전략을 이용한 학습 간단화

최근에 유행하는 다양한 카드 전략 게임들은 대부분 서로의 패를 볼 수 없는 불완전 게임이다. 기존의 연구에서는 정해진 덱과 보드의 상태를 백터화를 통해서 표현하였다[4]. 행동을 묶어서 메타화를 통한 부분 정책을 학습해서 이용한 연구도 진행됐었다[5].

기본적인 접근 방식은 q 러닝 과정에서 행동과 상태를 더 높은 계층으로 표현한다. 각 상태마다 나올 수 있는 많은 상태를 전부다 학습하지 않고, 사람이 만드는 전략을 이용한다. 카드게임에서의 전략은 현제 상황에서 최선의 수를 찾지 못할 수 있지만, 탐색하는 수를 줄여서 강화학습에 필요한 학습량을 크게 줄일 수 있다. 또한 복잡한 게임에서 같은 상태는 다시 나올 확률이 극히 적다. 비슷한 상태들끼리 같은 상태로 표현하고, 행동을 전략으로 바꿈을 통해서 Q 러닝을 이용한 짧은 훈련으로도 인공지능을 만들 수 있다.

게임의 규칙은 블리자드의 게임인 하스스톤과 유사하다. 양쪽 플레이어는 18 장의 덱과 30 의 체력을 가지고 시작하며, 상대방의 체력을 0 이하로 만들면 승리한다. 자원은 1 로 시작하고 차례마다 1 씩 올라가며 최대 10 까지 증가한다. 자신의 차례에서 패에서 자신의 자원을 넘지 않는 카드들을 필드에 둘 수 있다. 필드에 있는 카드마다 공격력과 체력을 가지며

낸 차례에 공격할 수 없다. 카드에 공격하면 서로 상대의 체력을 자신의 공격력만큼 감소시키며, 0 이하일경우에 파괴한다. 영웅을 공격하면 공격력만큼 체력을 감소시킨다.

덱은 비용이 1~9 인 카드들로 이루어져 있고 비용이 1 인 카드의 공격력과 체력은 [1,1,2]로 나타낸다. 비용마다 공격력과 체력이 반대인 카드가 쌍으로 18 장을 구성한다. 비용이 1 인 카드들의 공격력과 체력이 1 씩 증가해서 비용이 9 까지 총 18 장을 구성한다. ([1,1,2],[1,2,1],[2,2,3],[2,3,2],[3,3,4]…) 무작위로 섞인 덱에서 선공은 3 장, 후공은 5 장을 가진 채 시작한다

상태는 현재 필드의 상태, 자신의 체력, 상대에 체력의 정보를 가지고 상태를 표현한다. 필드의 상태는 자신이 보유하는 필드의 카드 숫자와 상대방의 카드 숫자를 사용한다. 게임의 시작은 같은 상태에서 시작하고(30, 30, 0, 0) 학습 도중에 만개 정도의 다른 상태들을 거쳤다.

기본적인 전략을 크게 3 가지인 공격적, 수비적, 교환적으로 나눈다. 공격적인 전략은 필드에 있는 모든 카드들을 상대방 영웅을 향해서 공격한다. 수비적인 전략은 상대방의 필드를 비우는데 우선시하고 나올 수 있는 최고의 상태를 선택한다. 교환적인 전략은 상대의 필드를 최소화하고 자신의 필드를 최대화하는 최고의 상태를 선택한다. 공격적인 전략을 제외한 나머지는 탐색을 통해서 모든 경우의 수를 고려한 뒤 판단을 내린다. 전략은 사람이 휴리스틱 정의해야 하고 세분화될수록 학습하는 에이전트의 성능이 올라간다. 사람의 개입은 수많은 선택지들을 어느정도 군집화를 통해서 나누고, 에이전트는 상황마다 어떤 군집에 있는 수가 제일 맞는지를 학습할 수 있다. 하지만 전략을 통해서 선택되지 않은 행동들은 영원히 제외된다.

각 플레이어는 공격이 끝나고나서 패에 있는 카드들 중에서 비용이 가장 큰 카드부터 자원이 넘지 않은 선까지 필드에 차례대로 낸다.

```
Trade algorithm
P: 나올 수 있는 모든 상태
M: 자신의 필드에 있는 카드의 숫자
E: 상대의 필드에 있는 카드의 숫자
D: 상대한테 줄 수 있는 피해의 합
define score:
   score = M*30 – E*12 + D*3
   for card in m:
      score = score + card.life + card.attack
   for card in E:
      score = score – card.life – card.attack
bestscore = -999
for board in P:
   bestscore < score(board) then
      bestscore = score
      bestboard = board
return bestboard
```

(그림 1) 교환적 알고리즘

```
Defensive algorithm
P: 나올 수 있는 모든 상태
M: 자신의 필드에 있는 카드의 숫자
E: 상대의 필드에 있는 카드의 숫자
D: 상대한테 줄 수 있는 피해의 합
define score:
   score = M*10 – E*40 + D
   for card in m:
      score = score + card.life + card.attack
   for card in E:
      score = score – card.life – card.attack
bestscore = -999
for board in P:
   bestscore < score(board) then
      bestscore = score
      bestboard = board
return bestboard
```

(그림 2) 수비적 알고리즘

4. 실험 결과

전략을 학습하는 Q 러닝 에이전트를 후공으로 한 가지 전략만 사용하는 상대로 5 만번 학습을 한다. 입실론 값은 0.1 로 시작해서 100 번마다 0.005 씩 증가시키고, 시간에 대해서 점수 페널티를 0.99 로 설정하였다.

<표 1> 전략들 간의 대결에서 후공의 승률

	공격(선)	교환(선)	수비(선)	AI(선)
공격(후)	42.06	12.83	13.02	8.31
교환(후)	70.36	33.35	33.21	31.82
수비(후)	70.73	32.73	32.84	31.75
AI(후)	72.8	34.24	33.63	32.94

동일한 전략만 사용한 경우에 선공의 승률이 58%에서 67%로, 후공보다 유리하다. 시작하는 카드가 2 장 더 적음에도 불구하고 먼저 카드를 내고 공격할 수 있어서 우세하였다.

공격적인 전략은 필드를 계속 유지하는 전략들에 비해서 크게 불리하고, 에이전트가 학습한 Q 테이블 역시 상대방의 체력이 낮은 경우에 공격적인 전략을 골랐다. 교환적 전략과 수비적 전략은 서로 같은 필드에서 비슷하게 교환을 해서 큰 차이를 보이지 않았다. 교환적 전략은 자신의 카드를 아끼면서 교환을 통해서 불리한 판을 역전하는 판이 존재하였다. 수비적 전략은 상대의 카드들을 최대한 제거해서 상대방의 선택지를 최소화 해서 상대의 전략의 차이가 크지 않았다. 3 가지의 전략을 모두 구사하는 에이전트는 한쪽으로 기울어진 판들을 제외한 나머지 비등한 판들을 더 이길 수 있었다. 특히 공격적인 전략을 상대로 유리할 경우에 공격적인 전략을 구사해서 길어져서 역전의 기회를 차단하였다. 상태들 중에서 Q-table 이 제대로 학습되지 않은 구간이 많아서 기본값인 공격적인 전략이 자주 두었다.

5. 결론

복잡한 게임에서 나올 수 있는 무수히 많은 상태와 행동들을 줄임으로써 Q 러닝만으로도 학습이 가능하였다. 그 결과 동일한 전략을 사용하는 것보다 더 뛰어난 인공지능을 만들 수 있었고, 상태마다 전략을 바꿔 사용해서 승률을 더 높였다. 이 연구를 통해서 상태와 전략의 정의로 가장 뛰어난 인공지능을 못하나, 적은 학습량으로도 전략을 구사하게 하는 것을 보였다.

6. 감사의 글

이 논문은 2017 년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2017R1A2B4003558)

참고문헌

[1] Silver, David, et al. "Mastering chess and shogi by self-play with a general reinforcement learning algorithm." arXiv preprint arXiv:1712.01815 (2017).
[2] Gibbons, Robert. A primer in game theory. Harvester Wheatsheaf, 1992.
[3] Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." arXiv preprint arXiv:1312.5602 (2013).
[4] Learning Artificial Intelligence in Large-Scale Video Games: A First Case Study with Hearthstone: Heroes of Warcraft
[5] Frans, Kevin, et al. "Meta learning shared hierarchies." arXiv preprint arXiv:1710.09767 (2017).