심층 강화학습 기술 동향

Research Trends on Deep Reinforcement Learning

장수영 (S.Y. Jang, sy.jang@etri.re.kr) 도시·공간ICT연구실 연구원
윤현진 (H.J. Yoon, hjyoon73@etri.re.kr) 도시·공간ICT연구실 선임연구원
박노삼 (N.S. Park, siru23@etri.re.kr) 도시·공간ICT연구실 책임연구원
윤재관 (J.K. Yun, jkyun@etri.re.kr) 도시·공간ICT연구실 책임연구원
손영성 (Y.S. Son, ysson@etri.re.kr) 자율형IoT연구실 책임연구원

ABSTRACT

Recent trends in deep reinforcement learning (DRL) have revealed the considerable improvements to DRL algorithms in terms of performance, learning stability, and computational efficiency. DRL also enables the scenarios that it covers (e.g., partial observability; cooperation, competition, coexistence, and communications among multiple agents; multi-task; decentralized intelligence) to be vastly expanded. These features have cultivated multi-agent reinforcement learning research. DRL is also expanding its applications from robotics to natural language processing and computer vision into a wide array of fields such as finance, healthcare, chemistry, and even art. In this report, we briefly summarize various DRL techniques and research directions.

KEYWORDS 심층 강화학습, 멀티 에이전트 강화학습, 분산 강화학습 프레임워크, 가상 학습 환경, 강화학습 응용 분야

Ⅰ. 서론

강화학습은 시행착오를 통해 수집되는 수많은 데이터 속에 숨어 있는 패턴을 학습을 통해 찾아내는 방법으로, 그 역사는 1950년대까지 거슬러 올라갈 정도로 깊다. 강화학습은 기본적으로 마코프 결정 과정(MDP: Markov Decision Process)을 이용해 수학적으로 모델링되는 순차적 의사결정 문제를 풀기 위한 하나의 최적화 방법이다. MDP는 (S, A, P, R, γ) 튜플로 정의된다. S와 A는 각각 에이전트의 상태 공간과 행동 공간을 의미한다. P와 R은 각각 전이 확률과 보상 함수로 에이전트가 상태 $s \in S$에서 행동 $a \in A$를 수행했을 때, P는 다음 상태의 확률 분포를, R은 다음 상태에서 에이전트가 받을 보상을 정의한다. 보상은 에이전트가 한 행동의 좋고 나쁨을 판단하기 위한 지표이다. γ는 할인

* DOI: https://doi.org/10.22648/ETRI.2019.J.340401
* 본 연구는 한국전자통신연구원 연구운영비지원사업의 일환으로 수행되었음[19ZH1100, 초연결 공간의 분산 지능 핵심원천 기술].

율을 의미하며, 누적 보상 계산 시 이 할인율을 이용해 미래 보상을 감가상각해 준다. 이는 에이전트가 목표를 빠르게 달성하도록 도우며 누적 보상의 발산을 막음으로써 학습이 안정적으로 이루어지도록 한다. 위 MDP로 정의된 순차적 의사결정 문제에 대한 최적 정책을 강화학습을 통해서 찾을 수 있다. 여기서 정책이란 상태 값을 입력으로 받아 에이전트가 수행해야 할 행동을 결정하는 함수이다. 강화학습 에이전트는 자신의 센서나 다른 에이전트로부터의 정보 등을 토대로 자신 및 주변 환경의 상태를 관측하고, 그 관측한 상태 값과 정책을 토대로 수행할 행동을 결정하며, 때때로 그 행동에 대한 보상을 받게 된다. 이때 강화학습 에이전트는 누적 보상의 기대치를 최대화하고자 자신의 정책을 학습하게 되며, 누적 보상의 기대치를 최대화하는 정책이 바로 최적 정책이 된다. 이는 보상 가설(Reward hypothesis)에 근거를 둔다. 보상 가설이란 어떤 목표(예, 문제 해결)를 에이전트의 누적 보상을 최대화하는 것으로 설명할 수 있다는 것이다. 이는 보상 함수 설계의 중요성을 의미하기도 한다. 강화학습은 에이전트가 보상이라는 신호에 의지해서 시행 착오를 통해 환경과 상호작용하며 자신의 정책을 학습하는 방법이기에 잘 못 된 보상 함수는 에이전트 학습을 어렵게 할 뿐만 아니라 설사 학습이 어느 정도 되었다고 하더라도 예기치 못한 부작용을 초래하기도 한다.

MDP는 다양한 종류의 게임뿐만 아니라 로봇 제어, 주식 거래, 자원 할당, 추천 시스템, 자연어 처리 등 현실 세계의 광범위한 문제들을 표현할 수 있는 굉장히 일반적인 수학적 모델링 방법이다. 즉, 강화학습이 적용될 수 있는 분야가 무궁무진하다는 이야기이다. 적용 분야가 무궁무진한 것에 비해 연구자들은 한동안 강화학습을 이용해 현실 세계의 문제들을 해결하는 데 어려움을 겪었다. 어느 정도 성과를 냈던 GridWorld나 Tic-Tac-Toe 같은 단순한 상태 공간과 행동 공간을 가지는 문제들에 비해 현실 세계의 문제들은 너무 복잡했기 때문이다. 복잡한 상태 공간과 행동 공간을 가지는 문제를 해결하기 위한 정책 함수를 근사하고 학습하는 데 신경망을 도입한 연구들이 수행되기 시작했다. 그러다가 심층신경망 학습 기술 그리고 이미지 인식을 위한 합성곱 신경망 기술의 발전 등이 이루어지며, 강화학습에 딥러닝을 결합한 심층 강화학습 기술 연구가 이루어지기 시작한다. 2013년 영국의 작은 스타트업인 딥마인드에서 심층 강화학습 기술을 이용해 다양한 Atari 게임에서 사람과 같이 게임 화면만을 이용해서 사람보다 게임을 더 잘하는 Deep Q-Network(DQN)[1]를 개발하며 심층 강화학습 기술은 많은 연구자들의 관심을 받게 된다. 딥마인드는 2013년 다양한 Atari 게임(DQN), 2016년 바둑(알파고)에 이어 2019년 스타크래프트2(알파스타)까지 사람보다 더 잘하는 인공지능을 개발해내며 엄청난 기술 발전을 이끌어가고 있다. 현재, 현실 세계의 광범위한 문제들을 해결하기 위해 심층 강화학습 기술을 적용한 많은 연구들이 이루어지고 있다. 최신 강화학습 연구에서 다루고 있는 문제들은 다음과 같은 특징들을 가지고 있다.

- 에이전트가 자신이 처한 환경에 대한 모든 상태 정보를 알 수가 없다.
- 에이전트의 상태 공간이 매우 크다(예, DQN에서 정의한 Atari 게임의 상태 공간은 28,224($84 \times 84 \times 4$, 84×84 픽셀 크기 이미지 4장), 바둑의 상태 공간은 우주의 원자 수인 10^{80}보다 10^{172}많은 (3^{19^2}).
- 에이전트의 행동 공간이 매우 크다(예, 연속적인 행동 공간을 갖는 로봇 관절 제어).
- 에이전트는 거의 실시간으로 매 시간 간격마다 어떤 행동을 수행해야 하는지를 결정해야

한다.

- 에이전트는 주어진 문제를 해결하기 위해서 다른 에이전트들과 공존하며, 그들과 협업하거나 경쟁하거나 혹은 협업과 경쟁을 동시에 해야 한다.

이는 사람이 일반적으로 문제를 해결하기 위해 처하는 상황과 매우 흡사하다. 그렇기에 이전과 같은 상황에서 에이전트 스스로 시행착오를 통해서 지능을 학습할 수 있는 강화학습을 강 인공지능(Strong AI), 즉 AGI(Artificial General Intelligence)를 위한 주요 기술로 생각하는 연구자들이 많다.

Ⅱ. 싱글 에이전트 강화학습

싱글 에이전트 강화학습 기술은 크게 Q-러닝 기반 강화학습과 정책 경사(PG: Policy Gradient) 기반 강화학습으로 분류할 수 있다.

1. Q-러닝 기반 강화학습

Q-러닝 기반 강화학습은 매 상태-행동 pair에 대한 Q 값을 근사한 후, 이를 기반으로 어떤 상태에서 어떤 행동을 취할 지 결정하는 방법이다. 탐험을 위해서 대개 ε-탐욕 정책(ε-greedy policy)을 많이 이용한다. ε-탐욕 정책이란 특정 상태에서 확률 ε로 랜덤한 행동을, (1 − ε)의 확률로 제일 높은 Q 값을 가지는 행동을 선택하는 방법이다. 대표적으로 DQN 그리고 DQN에 6가지 DQN 개선 알고리즘을 결합한 Rainbow[2] 등이 있다.

DQN의 기술적 특징은 크게 1) 이미지 인식을 위한 합성곱 신경망 이용; 2) 샘플 간 상관 관계를 없애고, 샘플 효율성을 높이기 위한 경험 리플레이 도입; 3) 학습의 안정성을 위해서 에이전트의 행동을 결정하는 online Q-network와 target Q 값 계산에 사용되는 target Q-network 분리로 요약될 수 있다.

Rainbow는 DQN에 다음 6가지 DQN 개선 알고리즘을 결합한 기법이다. 1) Double Q-learning - DQN은 현재 상태에서 target Q 값을 계산할 때 target Q-network의 최대값을 취하기에 target Q 값이 과대평가되어 학습 성능이 저하되는 경우가 발생한다. Double Q-learning은 online Q-network를 최대화하는 행동 값을 target Q-network의 입력으로 하여 target Q 값을 계산함으로써 target Q 값의 과대평가를 방지한다; 2) Prioritized experience replay - DQN은 경험 리플레이로부터 경험을 균일하게 추출하여 학습한다. 우선순위 기반 경험 리플레이(Prioritized experience replay)는 학습에 도움이 되는 샘플을 더 높은 확률로 추출하는 방법이다; 3) Dueling networks - DQN은 Q 값을 바로 계산한다. Q 값은 상태와 행동을 동시에 고려하기에 특정 상태에서의 가치를 평가할 때 행동에 영향을 크게 받을 수 있다. Dueling networks는 Q 값을 특정 상태에서의 가치(Value)와 그 상태에서 취할 수 있는 다양한 행동에 대한 이득(Advantage)으로 분리함으로써 행동의 가치를 고려하면서도 특정 상태에서의 가치를 더 강건하게 계산할 수 있게 된다; 4) Multi-step learning - DQN은 target Q 값을 계산할 때 바로 다음 상태, 즉 1-step 부트스트랩 후의 보상을 사용한다. 이를 확장하여 n-step 부트스트랩 후의 보상 정보를 이용하여 학습하게 되면, 즉 multi-step learning, 학습 안정성과 속도가 개선된다; 5) Distributional RL - DQN은 Q 값의 기대치를 이용한다. 이 경우, MDP에 내재된 랜덤성을 활용하기 어렵다. Distributional RL은 하나의 평균값 대신 보상의 분포를 이용하는 방법으로 학습 성능의 개선뿐만 아니라 에이전트가 위험이 큰 행동을 피할 수 있게 하기 때문에 안전한 에이전트 설계가

가능해진다; 6) Noisy Nets – DQN은 ε–탐욕 정책을 이용한다. 하지만 ε–탐욕 정책은 현재 에이전트가 처한 상황에 관계없이 랜덤 행동을 출력하기에 종종 비효율적인 탐험을 할 뿐만 아니라 ε 값 설정에 대한 문제도 존재한다. Noisy Nets는 정책 신경망 학습 시 정책 신경망의 가중치와 편향에 잡음을 부여하는 신경망(Noisy Nets)을 함께 학습함으로써 에이전트 행동 랜덤성이 에이전트가 처한 상태에 따라 그리고 학습 진행 시간에 따라 자동으로 적응되는 이점(예, 학습이 진행됨에 따라 랜덤성이 감소하며 greedy 선택을 촉진함)을 얻게 된다.

2. 정책 경사 기반 강화학습

정책 경사 기반 강화학습은 어떤 상태에서 어떤 행동을 취할지를 결정하는 정책을 직접 구한다. 정책이란 관측 값을 입력으로 받아서 행동 값을 출력하는 함수로 매개변수 벡터 θ로 정의될 수 있다. 심층 강화학습에서는 이 함수를 심층신경망을 이용해서 근사하며, 이때 매개변수 벡터 θ는 신경망의 가중치와 편향 값이다. 정책 경사 기반 강화학습은 이 매개변수 벡터 θ를 구하기 위해서 정책 경사 기법을 이용한다. 정책 경사 기법은 θ를 구하기 위해서 경사상승법을 이용하는 방법이다. 즉, 특정 θ에 대한 목적 함수의 경사(Gradient)를 구한 후, 이 경사가 상승하는 방향으로 일정 거리만큼 θ를 업데이트하는 것을 경사가 수렴할 때까지 혹은 최대 타임 스텝만큼 반복하는 것이다.

목적 함수는 정책 π_θ에 따라 행동 시 얻게 되는 누적 보상의 기대치이며, 식 (1)과 같이 표현된다.

$$J(\theta) = E_{\pi_\theta}[r(s,a)] \quad (1)$$

목적 함수의 경사는 정책 경사 정리(Policy gradient theorem)[3]에 의해서 식 (2)와 같이 정의된다.

$$\nabla J(\theta) = E_{\pi_\theta}[Q_\pi(s,a)\nabla_\theta \ln \pi_\theta(a|s)] \quad (2)$$

정책 경사는 목적 함수를 최대화하는 방향으로 정책의 매개변수를 식 (3)과 같이 업데이트한다.

$$\theta \leftarrow \theta + \alpha \nabla J(\theta) \quad (3)$$

정책 기반 강화학습의 대표 예로 PPO(Proximal Policy Optimization)[4]가 있다.

정책 기반 강화학습의 단점 중 한 가지는 정책 경사를 통해 매개변수를 점진적으로 갱신하는 것이, 이 매개변수로 이루어진 정책을 급격하게 변화시킬 수 있다는 것이다. 이는 학습의 불안정성을 야기하며 학습 속도 및 성능 저하의 원인이 된다. TRPO(Trust region policy optimization)[5]는 급격한 정책 갱신을 방지하기 위해서 정책 갱신 전후 간 KL(Kullback–Leibler) divergence를 일정 수준 이하로 제약하는 조건을 추가하였다. TRPO는 DQN으로는 해결하지 못 했던 연속적인 행동 공간을 갖는 로봇 제어 문제를 성공적으로 해결함으로써 연구자들의 관심을 끌게 된다. 하지만 TRPO는 위 제약 조건을 풀기 위해서 많은 계산이 필요할 뿐만 아니라 다양한 신경망 구조(예, 드랍아웃, 매개변수 공유)와 호환되지 않는 단점이 존재한다. PPO는 TRPO의 성능은 유지하면서도 이러한 단점들을 보완하기 위해 많은 계산량을 필요로 하는 KL divergence 제약 조건은 없애고, 단순히 TRPO의 목적함수에 존재하는 정책 갱신 전후의 비율을 클리핑(Clipping)함으로써 정책 갱신의 양을 간접적으로 제한하는 방식을 택하고 있다. PPO는 2017년에 공개되었지만 아직도 최상의 알고리즘일 정도로 성능, 계산 효율성 그리고 구현 용이성 등의 측면에서 매우 뛰어나다.

3. 내적 보상(Intrinsic Reward)을 이용한 강화학습

강화학습은 시행착오를 통해서 환경을 탐험하며

학습하는 방법이다. 이때 에이전트는 수행한 행동에 대한 결과로 주어지는 보상을 이용해 자신의 행동을 평가하며 정책을 업데이트하게 된다. 그렇기에 매 행동마다 보상이 주어지는 경우에는 학습이 잘 이루어질 수 있지만, 보상이 드물게 혹은 지연되어 주어지는 환경에서는 정책 학습이 성공적으로 이루어지기 어렵다. 몬테주마의 복수 게임의 경우, 장애물과 해골 등을 모두 피해서 열쇠를 먹을 때 보상이 한 번 주어진다. 이 경우, 열쇠를 먹기 위해서 특정 상태에서 행한 행동이 유익했는지, 유해했는지, 또는 무의미했는지를 판단하기가 어렵다. 대다수의 Atari 게임에서 사람보다 더 좋은 점수를 획득한 DQN의 경우, 몬테주마의 복수 게임에서는 점수 획득에 실패했다. 이러한 문제를 해결하기 위한 방법 중 하나로 내적 보상을 이용하는 방법이 있다.

내적 보상이란 환경으로부터 주어지는 보상이 아닌 에이전트 스스로 생성하는 보상을 의미한다. 사람의 내적 동기를 통한 학습 방법을 모방한 것이다. 대표적인 내적 보상 함수로 예측 오차가 있다. 예측 오차는 에이전트가 예측한 다음 상태와 실제 다음 상태 간 차이로 정의된다. 에이전트는 다음 상태를 예측하기 위해서 대개 하나 이상의 예측 모델(예, 신경망)을 정의하고, 이를 정책과 함께 학습한다. 에이전트가 환경을 탐험하며, 예측 모델의 학습이 이루어지기 때문에 에이전트에게 익숙한 상태에서는 이 예측 오차가 낮아지게 되고, 낯선 상태에서는 이 예측 오차가 높아지게 된다. 즉 에이전트의 탐험을 장려하는 효과가 있으며, 게임에서는 에이전트가 죽어서 초기 상태로 돌아가는 것을 억제하는 효과도 있다. 에이전트가 죽으면 초기 상태로 이동하기 때문에 초기 상태는 에이전트에게 익숙한 상태이기 때문이다. 내적 보상 함수는 에이전트가 매 행동을 수행할 때마다 보상을 얻을 수 있도록 설계되기에, 환경으로부터 드문 혹은 지연된 보상이 주어지는 환경에서도 좋은 학습 성능을 낼 수 있다. 또한, 매 환경 내지는 매 태스크마다 사람이 조밀한 보상 함수를 직접 설계해 줘야 하는 필요성을 낮출 수 있기에 강화학습 적용 분야를 넓히는 데 큰 도움이 된다. 하지만 에이전트가 달성해야 하는 태스크의 방향성과 내적 보상의 방향성 일치 여부, 학습이 진행됨에 따라서 동일한 상태에서 동일한 행동을 수행했을 때 얻는 보상의 비정상성(Non-stationarity), 그리고 환경마다 달라지는 보상의 스케일 문제 등을 고려해 주어야 한다. 대표적인 최신 연구인 RND(Random Network Distillation)[6]를 소개한다.

RND는 목표, 예측 및 정책, 세 가지 신경망으로 구성되어 있다. 정책 신경망은 에이전트의 행동을 결정하는 신경망이며, 목표 및 예측 신경망은 다음 상태 값을 입력으로 받아서 어떤 특징 값을 출력하는 신경망이다. 목표 신경망은 가중치가 랜덤하게 설정되어 고정되며, 예측 신경망은 목표 신경망과 동일한 구조를 갖는 신경망으로 목표 신경망과 동일한 출력을 내도록 정책 신경망과 함께 학습된다. 즉, 랜덤 신경망(Random network)을 예측 신경망에 증류(Distillation)하는 효과가 있기에 random network distillation이라 불린다. RND에서는 내적 보상을 위한 가치 함수와 외적 보상을 위한 가치 함수를 각각 구한 후 합치는 방식을 택하였으며, 정책 신경망 최적화를 위해서 PPO를 사용한다.

III. 멀티 에이전트 강화학습

멀티 에이전트 강화학습(MARL: Multi-Agent Reinforcement Learning)이란 주어진 환경에서 두 개 이상의 에이전트가 협업 또는 경쟁을 통해 높은 보상을 얻을 수 있는 행동 정책(Policy)을 학습하는 기

술로 정의할 수 있다. 기존 싱글 에이전트 중심의 강화학습 기술에서 고려하고 있는 탐색-이용 딜레마(Explore-exploit dilemma), 부분 관측 가능성(Partial observability)에 따른 문제들뿐만 아니라 멀티 에이전트 환경이 갖는 고유의 비정상성 특성과 에이전트 간의 신뢰할당(Credit-assignment) 문제까지 추가로 고려해야 하기 때문에 멀티 에이전트 학습은 더욱 복잡하고 어려운 것으로 알려져 있다.

최근 심층신경망을 이용하여 에이전트의 정책 함수나 가치 함수를 근사하는 심층 강화학습의 발전에 힘입어 이를 기반으로 하는 다양한 멀티 에이전트 심층 강화학습 알고리즘들이 제안되고 있다. 멀티 에이전트 심층 강화학습 알고리즘 역시 최적의 행동가치함수(Action-value function) 학습을 목표로 하는 Q-러닝 계열과 정책 함수를 경사상승법을 이용하여 직접 학습하는 정책 경사 계열로 구분할 수 있으며, 대표적인 최신 알고리즘으로는 Q-러닝 계열의 QMIX 알고리즘[7]과 정책 경사 계열의 MADDPG 알고리즘[8] 등이 있다.

1. Q-러닝 기반 MARL 알고리즘

Q-러닝은 주어진 상태에서 주어진 행동이 가져올 누적 보상의 기댓값을 예측하는 Q-함수(행동가치함수)를 벨만최적방정식을 이용하여 학습하고, 에이전트는 학습한 Q-함수의 예측값과 ε-탐욕 정책에 따라 각 상태에서 가장 높은 가치를 주는 행동을 선택하며 환경 탐색을 이어 나가는 방식으로 실행된다.

1990년대 초반 제안된 IQL(Independent Q-Learning) 알고리즘[9]과 최근 DQN을 적용한 멀티 에이전트 심층 강화학습 알고리즘[10]에서는 다수의 에이전트 각각이 Q-러닝 알고리즘을 이용하여 자신이 부분적으로 관측한 로컬 상황(Observation)을 기반으로 자신의 행동을 선택할 수 있는 로컬 Q-함수를 다른 에이전트에 대한 정보 없이 독립적으로 학습하는 방식을 제안하고 있다. 이러한 완전 분산형(Fully decentralized) 구조의 경우 학습과 탐색을 동시에 수행하고 있는 다른 에이전트들을 환경의 일부로 간주하기 때문에 주에이전트 관점에서는 Q-러닝 알고리즘이 수렴하기 위해 필요한 마코프 가정을 만족하지 못하게 되어 안정적인 학습이 어렵다는 단점을 갖는다. 또한, 비정상적(Non-stationary) 환경으로 인해 DQN 알고리즘의 경우 경험 리플레이(Experience replay) 메모리에 저장된 과거의 경험들을 그대로 활용하기 어렵다는 문제도 있다.

반대로, 에이전트 각각의 로컬 Q-함수를 학습하는 대신에 에이전트들의 모든 로컬 관측 상황과 전역 환경 상태(Global state)를 고려하여 모든 에이전트들의 공동행동(Joint action)이 가져올 누적 보상의 기댓값을 예측하는 한 개의 전역 Q-함수를 학습하는 방식도 가능하다. 하지만, 이러한 완전 집중형(Fully centralized) 학습 방식은 에이전트의 수가 증가함에 따라 공동행동 공간이 기하급수적으로 증가하기 때문에 에이전트의 수에 제약이 따른다는 단점을 갖는다.

QMIX 알고리즘은 이 두 가지 방식을 절충하여 다수의 에이전트가 각각 자신이 부분적으로 관측한 로컬 상황에서 개별 행동의 가치를 예측해 주는 로컬 Q-함수와 모든 로컬 Q-함수의 출력값들과 에이전트들의 개별 행동에 따른 전역 환경 상태를 모두 종합하여 에이전트들의 공동행동에 대한 가치를 예측하는 전역 Q-함수를 모두 사용하는 새로운 멀티 에이전트 학습 방식을 제안하고 있다. QMIX 알고리즘은 로컬 Q-함수를 근사하는 에이전트 신경망과 다른 에이전트들의 행동을 고려하는 전역 Q-함수를 근사하는 믹싱 신경망을 최적

의 공동행동 학습을 목표로 종단 간(end-to-end) 학습하기 때문에, 기존 완전 분리형 방식보다 안정적인 학습이 가능하다는 장점을 갖는다. 실제로 QMIX 알고리즘은 스타2 전략게임의 멀티유닛 제어 작업에서 기존 Q-러닝 기반의 MARL 알고리즘보다 개선된 성능을 보여주고 있다.

2. 정책 경사 기반 MARL 알고리즘

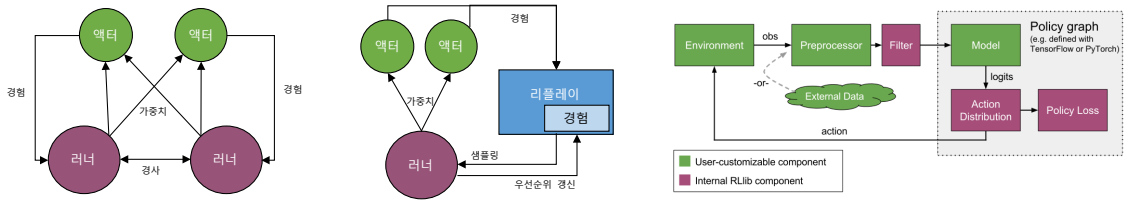
정책 경사 알고리즘은 주어진 상태에서 에이전트가 해야 하는 행동을 출력하는 정책 함수를 경사상승법을 이용하여 직접 학습하는 방식으로, 에이전트의 행동을 결정하는 정책 함수를 근사하는 정책 신경망(액터, actor)과 정책을 평가하는 역할을 하는 가치 함수(Q-함수)를 근사하는 가치 신경망(크리틱, critic)을 별도로 두고, 이 두 신경망을 종단 간 학습하는 액터-크리틱 알고리즘이 대표적이다. 액터-크리틱 알고리즘을 멀티 에이전트 학습으로 성공적으로 확장하기 위해서는 에이전트가 환경을 탐색하며 수집하는 경험에 따라 Q-함수의 예측값이 크게 변동하기 때문에 정책함수 학습이 느리다는 정책 경사 학습 고유의 취약점뿐만 아니라 비정상적 환경에서 수렴하기 어려운 Q-함수 학습의 문제점까지 모두 고려할 필요가 있다.

최근 제안된 MADDPG과 M3DDPG[11] 알고리즘은 N개의 에이전트들의 행동을 결정하는 N개의 분리된 액터(정책 신경망)와 정책 신경망 각각을 평가하는 N개의 크리틱(가치 신경망)을 경험 리플레이 메모리에 저장된 경험 데이터를 이용하여 학습하는 새로운 멀티 에이전트 학습 방식을 제안하고 있다. 특히 에이전트 각각은 다른 에이전트들이 선택한 행동들을 모두 고려하여 자신의 정책 신경망을 업데이트하는 방향을 결정하기 때문에 다른 에이전트들의 정책이 바뀌더라도 비교적 안정적인 학습이 가능하다. 학습 단계에서는 다른 에이전트들의 정보를 사용해야 하는 중앙집중형의 학습(Centralized learning)이 이루어지지만 학습이 완료된 후에는 독립된 정책 신경망(액터)을 이용하여 로컬 관측 상태에서 최적의 행동을 선택할 수 있는 분리된 실행(Decentralized execution)이 가능한 구조적 특징을 갖는다. 이러한, 액터-크리틱 기반의 멀티 에이전트 강화학습 기술들은 에이전트간 협업이 필요한 환경뿐만 아니라 경쟁이 필요한 환경에서도 우수한 성능을 보여주고 있다.

Ⅳ. 분산 강화학습 프레임워크

최근 강화학습 환경에서 최고의 성능을 보여주는 결과들은 대부분 병렬 연산을 이용하고 있다. 데이터를 분산하여 저장하는 딥러닝 분산 구조를 적용하는 것에서부터 환경에서 얻은 경험을 빠르게 많이 모으기 위한 것까지 분산 강화학습 구조를 적극적으로 활용하고 있다. 딥러닝 구조를 강화학습에 적용하기 위한 연구는 복수의 워커를 활용하는 비동기 분산 SGD[12], 분산 A3C[13], Gorila[14], 분산 DQN[15] 등이 있다. 이후, ES(Evolution Strategies)[16], 분산 BA3C[17], Ape-X[18] 등의 동기 방식 분산 경험 재현 방식의 연구가 진행되고 있다.

컴퓨팅 자원을 활용한 성능 개선 연구도 분산 강화학습의 중요한 요소이다. 예를 들어, OpenAI Five는 128,000 CPU 코어, 256 P100 GPU를 이용하여 하루에 900년 치 경험을 수집하기도 했다. 관련 연구로 RLlib[19], Mesh-TensorFlow, distributed TensorFlow, OpenAI Five, TF-Replicator[20] 등이 있다.



(a) IMPALA 멀티 러너 구조 (b) Ape-X 구조 (c) RLlib 구조

출처 Figure 1c: reprinted with permission from https://ray.readthedocs.io/en/latest/rllib-models.html

그림 1 **대표적 분산 강화학습 프레임워크**

1. IMPALA

IMPALA(Importance Weighted Actor-Learner Architecture)[21]는 2018년 딥마인드에서 발표한 프레임워크로서 빠른 학습을 위한 액터(Actor)-러너(Learner) 구성의 분산 강화학습 구조이다(그림 1a). IMPALA를 통해 단일 집합의 파라미터를 가진 하나의 강화학습 에이전트가 30가지(DMLab-30)의 서로 다른 태스크를 학습할 수 있음을 보여준다. 이를 위해서는 대량의 데이터와 길어진 훈련 시간이라는 문제를 해결해야 한다. IMPALA에서 액터는 별도의 변화도 계산 없이 자신의 경험을 중앙의 러너에게 전달하여 러너가 계산하는 방식으로 독립된 모델을 택하고 있다. 학습과 행위가 분리된 구조를 통해 전체 시스템 성능이 개선되며, 액터와 러너의 정책이 차이가 생길 수 있는 문제는 off-policy 보정 알고리즘을 활용한 큐 방식의 V-trace를 적용하여 해결한다. 딥마인드의 DMLab-30, Atari-57 등의 멀티 태스크 환경에서 우수한 결과를 보여주었다.

각 액터는 경험을 러너 중의 하나로 보내고, 러너는 이를 이용하여 학습하게 된다. 액터는 모든 러너에게서 가중치를 병렬로 얻고 병합한다. 가중치는 러너들에게 분산되고, 러너들 사이에 가중치 경사를 공유한다.

2. APE-X

Ape-X는 2018년 구글에서 발표했으며, IMPALA와 유사하게 경험을 공유하는 멀티 액터 방식으로 동작한다(그림 1b). 한편, Ape-X는 단일 러너가 중앙 집중 형태의 우선순위 기반 경험 리플레이로 정책 지연을 대응하는 방안을 제시한다. Ape-X는 액터들이 생성하는 데이터에서 가장 중요한 경험에 집중하며, TD Error라는 기준에 따라 샘플의 우선순위를 계산하여 경험을 재현한다. 또한 double Q-러닝 알고리즘, dueling DQN 네트워크 구조, 4프레임 스택 등을 활용함으로써 Atari-57에서 최상의 성능을 보여준다.

Ape-X 구조는 개별 환경을 가지고 있는 다수의 액터들이 경험을 생성하여 초기 우선순위를 계산하여 공유 메모리에 추가한다. 러너는 이 메모리에서 샘플링하여 학습하며, 네트워크와 메모리의 경험에 대한 우선순위를 업데이트한다. 액터의 네트워크는 러너의 네트워크 파라미터를 이용하여 주기적으로 업데이트된다. 기본적으로 액터와 러너 모두 분산 가능하지만, 실험 결과 수100개의 액터들이 CPU에서 동작하고, 단일 러너가 GPU에 있는 것이 가장 유용한 경험을 샘플링한다.

3. R2D2

R2D2(Recurrent Replay Distributed DQN)[22]는 RNN(Recurrent Neural Network)의 LSTM(Long short-term memory)과 Ape-X의 우선순위 경험 리플레이 및 분산 학습을 결합한 알고리즘이다. 데이터의 순서를 바꾸어 버리는 경험 리플레이와 RNN의 지연으로 인한 문제에 대한 해결책을 제시한다. 단일 네트워크 구조와 고정 하이퍼파라미터를 이용함으로써 Atari-57의 52 게임에서 인간 수준을 능가하는 성능을 R2D2는 보여준다.

R2D2는 RNN과 경험 리플레이를 동시에 사용하기 위한 기존의 방법—zero start state 방법과 전체 경험 리플레이 방법[23]—에서 발생하는 문제에 대한 해결 방안을 제시한다.

- Stored state: RNN의 상태도 함께 리플레이에 저장하고 이 상태를 이용하여 네트워크를 초기화한다. 이를 통해 초기 RNN 상태의 영향을 최소화한다.
- Burn-in: 'burn-in 구간(period)'이라는 개념을 이용하여, 리플레이의 일부는 트레이닝에 사용하지 않고 상태만 RNN으로 활용한다.

두 가지 방법 모두 zero state 방법과 비교하여 우수한 성능을 보여주며, 상태 불일치 등 분산 병렬 환경에서 발생할 수 있는 문제에 대한 방법을 제시한다.

4. RLlib

RLlib은 UC 버클리의 RISE 랩에서 개발되었으며, 파이썬 기반의 분산 병렬 머신러닝 프레임워크인 Ray상에서 동작하는 강화학습 라이브러리이다. RLLib은 1) 확장 가능한 분산 병렬 시스템의 고성능과 2) 다양한 강화학습 알고리즘 및 프레임워크에 대한 코드 재사용이라는 원칙 아래 다양한 애플리케이션을 위한 통합 API를 제공한다(그림 1c).

RLlib의 기본 알고리즘은 1) Ape-X, IMPALA와 같은 고성능 아키텍처 모델; 2) A2C, A3C, DDPG, DQN 등과 같은 경사 기반 모델; 3) ARS(Augmented Random Search), ES, Q-MIX와 같은 derivative-free 모델 등을 제공한다.

RLlib은 Open AI Gym 환경을 포함하여 멀티 에이전트와 다중정책(Multi-policy) 훈련 환경을 제공하며, 최신 강화학습 알고리즘을 지원한다. 대규모 클러스터로 확장 가능하도록 설계되어 있어, 간단한 파이썬 API를 이용하여 분산 및 멀티코어 학습을 가능하게 한다. 또한 텐서플로, 케라스, 파이토치 등의 다양한 머신러닝 프레임워크를 지원하여 호환성을 높여주며, 사용자 정의 컴포넌트를 위한 abstract 클래스를 제공하여 분산 병렬 알고리즘으로의 개발 및 확장성을 지원한다.

5. TF-Replicator

TF-Replicator는 2019년 2월 딥마인드에서 개발한 분산 머신러닝 프레임워크로서 텐서플로에서 동작할 수 있는 API를 제공한다. 데이터 및 모델 병렬 처리를 위한 단순한 모델을 제시하여, CPU/GPU/TPU 등을 포함하는 여러 클러스터 구조에서 동작하는 동기 또는 비동기 훈련 방식을 제공한다. TF-Replicator는 분류(Classification), 강화학습 등을 포함하는 범용 머신러닝 프레임워크로서, D4PG 학습 에이전트 시험을 통해 확장 가능한 강화학습 벤치마킹 성능을 제시한다.

TF-Replicator는 텐서플로의 replica의 구현상 어려움을 해결하기 위해, replica 정의를 위한 API와 내장 Replicator를 이용하여 다양한 시스템 구조의

분산 시스템 적용을 용이하게 한다. 다수의 액터들이 각자의 인스턴스를 통해 경험의 생성 및 계산을 수행하고 난 후, 리플레이 버퍼에 저장한다. 러너는 TF-Replicator로 구현되어 재현을 위한 샘플링 기능 등을 수행하며, 러너의 버퍼 갱신에 대한 요청을 처리한다.

TF-Replicator는 원래 딥마인드 개발자들이 TPU를 쉽게 활용하기 위한 목적에서 개발되었으며, 개발자들이 분산 시스템에 대한 전문적인 이해 없이도 GPU 또는 TPU 클러스터 환경의 성능을 활용할 수 있다는 장점이 있다.

V. 가상 학습 환경

본 절에서는 심층 강화학습을 위해 사용되는 가상 학습 환경(Environment)에 대해 설명한다.

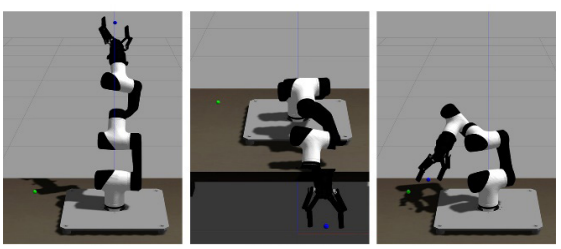
1. Gym-gazebo 2

Gym-gazebo는 ROS(Robot Operating System)와 Gazebo를 사용하여 강화학습으로 로봇을 훈련할 수 있는 환경이다. Gym-gazebo 2[24]는 Gym-gazebo의 실용성을 바탕으로 실세계의 작업에 강화학습을 적용할 수 있는 방법을 제공한다. 즉 강화학습을 전문적·산업적 레벨까지 끌어올리려고 하고 있으며, 기존의 전통적인 경로 계획 기법 대신 복잡하고 동적인 환경에서 적용될 수 있는 행동을 훈련하는 것을 목표로 하고 있다. 이를 달성하면 연구 개발 단계에서의 세팅을 쉽게 생산 환경으로 옮겨 갈 수 있다.

현재 Gym-gazebo 2는 로봇팔인 MARA(Modular Articulated Robotic Arm)를 이용하여 개발되고 있다. MARA는 각 액츄에이터, 센서, 모듈에 ROS 2가 있는 협업형 로봇팔이다. 각 모듈은 고유한 ROS 2를 지원하며 동기화, 결정론적(Deterministic) 통신 지연, ROS 2 소프트웨어 및 하드웨어 구성 요소 수명주기 등의 산업에 직접 적용될 수 있는 기능을 제공한다. Gym-gazebo2에서는 그림 2와 같이 강화학습 알고리즘을 신속하게 테스트할 수 있는 환경이 제공된다. 기본 환경은 테이블 중앙에 놓인 6 자유도(Degree of Freedom, DoF) MARA 로봇이고, 목표는 3D 공간의 한 목표지점에 로봇팔 집게의 중심점이 도달하는 것이다.

2. Unity ML-Agents

Unity는 3D 엔진 개발업체로 Unity에서 개발한 Unity ML-Agents SDK[25]를 사용하면 Unity 에디터로 제작한 게임이나 시뮬레이션을 Deep RL, ES 등의 알고리즘을 Python API를 통해 에이전트를 훈련시킬 수 있다. 그림 3은 Unity ML-Agents가 제공



(a) 초기화 상태 (b) 랜덤 동작 상태 (c) 학습에 의한 동작 상태

그림 2 MARA 가상 학습 환경

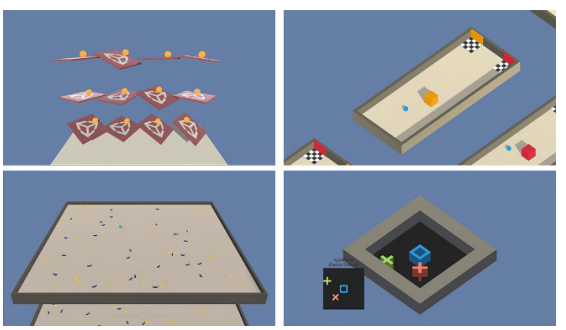


그림 3 Unity ML-Agents 제공 가상 학습 환경

하는 가상 학습 환경을 보여준다.

Unity ML-Agents의 학습 환경에는 크게 에이전트, 브레인, 아카데미의 3종류 객체가 있다. 각 에이전트는 자체의 상태 및 관측 값을 가지고 있고, 환경 내에서 고유한 행동을 하고 그에 따라 보상을 받는다. 에이전트들의 행동은 해당 에이전트의 브레인에 의해 결정된다. 각 브레인은 특정한 상태와 행동에 대한 공간을 정의하고, 에이전트의 행동을 결정한다. 특정 씬에 있는 아카데미 객체는 해당 환경에 포함된 모든 브레인을 자식으로 포함하고 있고, 각 환경은 환경의 범위를 정의하는 한 개의 아카데미를 포함한다.

Unity ML-Agents를 통해 다양한 훈련 시나리오가 연출될 수 있다. 하나의 브레인에 연결된 하나의 에이전트로 구성되는 싱글 에이전트 방식은 치킨 게임을 비롯한 모든 싱글 플레이어 게임에 적용할 수 있다. 그리고, 하나의 브레인에 연결된 여러 개의 독립된 에이전트로 구성되는 동시 싱글 에이전트는 훈련 시나리오를 병렬화하여 훈련을 가속화할 수 있다. 또한, 서로 대립되는 보상 함수를 갖고 있으며 하나의 브레인에 연결되어 있고 서로 상호작용하는 두 개의 에이전트인 적대적 자가 플레이와 하나 또는 여러 브레인에 연결되어 있으며, 보상 함수를 공유하고 서로 상호작용하는 멀티 에이전트인 협동형 멀티 에이전트도 지원한다. 경쟁형 멀티 에이전트는 대립되는 보상 함수가 있고 하나 또는 여러 브레인에 연결되어 상호작용하는 다중 에이전트이며, 생태계는 독립적인 보상 함수가 있고 하나 이상의 브레인에 연결되어 상호작용하는 멀티 에이전트이다.

3. OpenAI Gym

OpenAI Gym[26]은 강화학습 연구를 위한 툴킷이며, Gym은 공통 인터페이스가 제공되는 벤치마크 문제 모음과 사람들이 결과를 공유하고 알고리즘의 성능을 비교할 수 있는 웹사이트를 포함한다. Gym은 에이전트의 경험이 일련의 에피소드(Episode)로 분류되는 강화학습의 일시적인 설정에 중점을 둔다. 각 에피소드에서 에이전트의 초기 상태는 배포본에서 무작위로 샘플링되고 상호작용은 환경이 마지막 단계에 도달할 때까지 진행된다. 이 방식의 강화학습 목표는 에피소드당 총 보상 예상치를 최대화하고 최대한 적은 수의 에피소드에서 높은 수준의 성과를 달성하는 것이다.

Gym은 여러 자리 숫자를 추가하거나 시퀀스를 뒤집는 연산과 같은 알고리즘, 고전적인 Atari 게임, 보드 게임, 2D 및 3D 로봇과 같은 환경을 제공한다. 시뮬레이션에서 로봇을 제어하기 위해서는 MuJoCo 물리 엔진이 사용된다. 초기 릴리스 이후 오픈 소스 물리 엔진인 Box2D, Doom 게임 엔진을 기반으로 하는 환경이 추가되는 등 더 많은 환경이 만들어졌다. Gym을 이용하여 서로 다른 알고리즘들을 객관적으로 비교해 볼 수 있으며, 본인의 코드 외에 사이트에 있는 다른 사람들의 코드도 볼 수 있다.

4. DeepMind Lab

DeepMind Lab[27]은 일반 인공지능 및 기계학습 시스템의 연구 및 개발을 위해 설계된 1인칭 3D 게임 플랫폼이다. DeepMind Lab은 에이전트를 이용하여 크고, 부분적으로 관측되는 시각적으로 다양한 실세계에서 복잡한 작업을 학습하는 방법을 연구하는 데 사용될 수 있다. DeepMind Lab은 id 소프트웨어의 Quake III Arena 엔진 위에 구축되었으며, 주된 동작은 주위 둘러보기, 3D로 이동, 미로 탐색, 과일 수집, 위험한 통로 통과, 절벽에서

낙하 방지, 발사대를 사용하여 플랫폼 간 이동, 레이저 태그, 절차적으로 생성된 임의의 환경을 신속하게 학습 및 기억하는 작업, 신경 과학 실험에서 영감을 얻은 작업이 포함된다.

강화학습 API는 게임 엔진 상단에 구축되어 복잡한 에이전트에게 복잡한 관측을 제공하고 다양한 행동을 수행할 수 있게 한다. 플랫폼과의 상호작용은 사용자가 지정한 프레임 속도에 따라 엔진이 있는 다중 행동을 한 단계씩 밟을 수 있다. 따라서, 에이전트가 다음 행동을 제공할 때까지 관측이 제공된 후에 게임은 일시 정지된다. 각 단계에서 엔진은 보상(Reward), 픽셀 기반 관측 및 선택적으로 속도 정보를 제공한다. 에이전트는 움직임을 제어하는 동작(전방/후방, 왼쪽/오른쪽, 웅크리기, 점프), 쳐다보기(위/아래, 왼쪽/오른쪽), 레이저 태깅 등의 복수 개에서 동시 행동을 수행할 수 있다.

VI. 응용 분야

이 장에서는 로보틱스 분야를 중심으로 심층 강화학습이 적용된 다양한 응용 분야에 대해서 살펴보고자 한다.

로보틱스는 전통적인 강화학습 응용 분야이다. 앞서 말했듯이 심층 강화학습은 실제 환경에서 학습이 이루어지기 어렵기 때문에 주로 가상 학습 환경에서 학습이 이루어지게 된다. 하지만 가상 학습 환경과 실제 환경 간에는 모델링 오류로 인한 차이가 존재할 수밖에 없기에 가상 학습 환경에서 학습된 정책을 실제 기기에 바로 도입 시 제대로 동작하지 못 한다. 모델링 오류는 크게 에이전트 모델링 오류 및 에이전트가 행동하는 환경 모델링 오류로 구분할 수 있다. 자율 비행 드론을 예로 들면, 에이전트 모델 관련해서는 드론 무게, 프로펠러 마찰 계수, 모터 스펙, 드론 연식, 배터리에 따른 모터 출력 변화 등의 변수가 있을 수 있으며, 환경 모델 관련해서는 바람 세기 및 방향, 강수량, 지면 온도 등의 변수가 있을 수 있다. 이렇게 다양하며 시간에 따라 지속적으로 변하는 에이전트 및 환경을 정확하게 모델링하는 것은 불가능에 가깝기에 가상 환경과 실제 환경 간 에이전트 행동에 대한 상태 변화 차이가 발생하게 되고, 이는 가상 환경에서 학습된 정책의 성능을 저하시키는 주요한 원인이 된다. 이를 해결하기 위해서 참고문헌 [28]에서는 가상 환경의 모델링 정확도를 높이기 위한 연구와 모델링 오류에 강건한 정책을 학습하는 연구를 수행하였다. 모델링 오류에 강건한 정책을 학습하기 위해서 가상 환경에서 학습 시 에이전트의 역학에 무작위성을 부여하는 방법, 행동 값에 무작위한 작은 변동을 주는 방법 및 상태 공간을 압축하는 방법을 고려하였다.

이 외에도 심층 강화학습 기술은 자율 주행 시스템, 추천 시스템, 얼굴 인식 시스템, 데이터 센터 쿨링 시스템, 주식 거래 시스템, 질병 진단 시스템, 자연어 처리 시스템, 프로그램 작성 시스템, 수학 증명 시스템 등 광범위한 분야에서 그 효용성을 입증하고 있다. 참고문헌 [29]에서 이들 응용 분야에 대한 자세한 내용 및 참고문헌 그리고 더 다양한 적용 예를 다루고 있다.

VII. 결론

본 고에서 살펴본 바, 심층 강화학습 기술은 알고리즘의 성능, 학습 속도, 학습 안정성, 샘플 효율성 등을 향상시키기 위한 기술 발전뿐만 아니라 멀티 태스크, 멀티 에이전트, 에이전트 간 통신, 에이전트 간 협업과 경쟁이 공존하는 시나리오 등 실제 우리 주변에서 접할 수 있는 문제들을 해결하기 위한 방향으로도 기술 확장이 이루어지고 있다.

게다가 지도 학습이 적용되기 어려운 분야에 적용되어 더 좋은 성능의 문제 해결 방법을 찾기도 하며, 아예 새로운 문제에 맞닥뜨려도 스스로 인식한 상황 속에서 일련의 행동들을 수행하며 지속적인 학습을 통해 최적 정책을 찾아 나가는 에이전트 개발에도 적용될 수 있다. 이렇듯 심층 강화학습 기술은 광범위한 분야에서 성공적이며 창의적인 해결책들을 제시할 수 있는 기술이기에 향후 AGI의 주요 핵심 기술로써 연구의 질적 양적 성장을 넘어 실제 비즈니스 적용 사례 또한 폭발적으로 증가할 것으로 기대된다.

약어 정리

AGI	Artificial General Intelligence
DQN	Deep Q Networks
ES	Evolution Strategies
IMPALA	Importance Weighted Actor–Learner Architecture
IQL	Independent Q–Learning
KL	Kullback–Leibler
LSTM	Long Short Term Memory
MARA	Modular Articulated Robotic Arm
MARL	Multi–Agent Reinforcement Learning
PG	Policy Gradient
PPO	Proximal Policy Optimization
PPO	Proximal Policy Optimization
R2D2	Recurrent Replay Distributed DQN
RND	Random Network Distillation
RNN	Recurrent Neural Network
ROS	Robot Operating System
TRPO	Trust Region Policy Optimization

참고문헌

[1] V. Mnih et al., “Playing Atari with Deep Reinforcement Learning,” arxiv:1 312.5602, 2013.

[2] M. Hessel et al., “Rainbow: Combining Improvements in Deep Reinforcement Learning,” in *AAAI Conf. Crtif. Intell.*, New Orleans LA, USA, Feb. 2018, pp. 3215-3222.

[3] R.S. Sutton et al., “Policy Gradient Methods for Reinforcement Learning with Function Approximation,” in *Proc. Int. Conf. Neural Inf. Process. Syst.*, Denver, CO, USA, 2000, pp. 1057-1063.

[4] J. Schulman et al., “Proximal Policy Optimization Algorithms,” arxiv:1707.06347, 2017.

[5] J. Schulman et al., “Trust Region Policy Optimization,” in *Int. Conf. Mach. Learning(ICML)*, Lille, France, July 2015.

[6] Y. Burda et al., “Exploration by Random Network Distillation,” in *Int. Conf. Learning Representations*, New Orleans, LA, USA, 2019.

[7] T. Rashid et al., “QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning,” in *Int. Conf. Mach. Learning(ICML)*, Stockholm, Sweden, 2018.

[8] R. Lowe et al., “Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments,” in *Conf. Neural Inf. Process. Syst.*, Long Beach, CA, USA, 2017.

[9] M. Tan, “Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents,” in *Int. Conf. Mach. Learning(ICML)*, Amherst, MA, USA, 1993.

[10] A. Tampuu et al., “Multiagent Cooperation and Competition with Deep Reinforcement Learning,” *PLOS One*, vol. 12, no. 4, Apr. 2017, pp. 1-15.

[11] S. Li et al., “Robust Multi-Agent Reinforcement Learning via Minimax Deep Deterministic Policy Gradient,” in *AAAI Conf. Crtif. Intell.*, Honolulu, HI, USA, 2019.

[12] J. Dean et al., “Large Scale Distributed Deep Networks,” in *Int. Conf. Neural Inf. Process. Syst.*, Lake Tahoe, NV, USA, Dec. 2012, pp. 1223-1231.

[13] V. Mnih et al., “Asynchronous Methods for Deep Reinforcement Learning,” in *Proc. Int. Conf. Mach. Learning*, New York, USA, 2016, pp. 1928-1937.

[14] A. Nair et al., “Massively Parallel Methods for Deep Reinforcement Learning,” in *Int. Conf. Mach. Learning(ICML)*, Lille, France, July 2015.

[15] V. Mnih et al., “Human-Level Control Through Deep Reinforcement Learning,” *Nature*, vol. 518, no. 7540, 2015, pp. 529-533.

[16] T. Salimans et al., “Evolution Strategies as a Scalable Alternative to Reinforcement Learning,” CoRR, arXiv: 1703.03864, 2017.

[17] I. Adamski et al., “Distributed Deep Reinforcement Learning: Learn How to Play Atari Games in 21 Minutes,” CoRR, arXiv: 1801.02852, 2018.

[18] D. Horgan, et al., “Distributed Prioritized Experience Replay,” in *Int. Conf. Learning Representations*, Vancouver, Canada,

May 2018.

[19] E. Liang et al., "RLlib: Abstractions for Distributed Reinforcement Learning," in *Int. Conf. Learning Representations*, Vancouver, Canada, May 2018.

[20] P. Buchlovsky et al., "TF-Replicator: Distributed Machine Learning for Researchers," arxiv: 1902.00465, 2019.

[21] L. Espeholt et al., "IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures," Proc. Mach. Learning Research, vol. 80, 2018, pp. 1407-1416.

[22] S. Kapturowski et al., "Recurrent Experience Replay in Distributed Reinforcement Learning," in Int. Conf, Learning Representations, New Orleans, LA, USA, May 2019.

[23] H. Matthew and S. Peter, "Deep Recurrent Q-Learning for Partially Observable MDPs," in *AAAI Fall Symposia*, Arlington, VA, USA, Nov. 2015, pp. 29-37.

[24] N. G. Lopez et al., "Gym-Gazebo2, a Toolkit for Reinforcement Learning Using ROS 2 and Gazebo," arxiv: 1903.06278, 2019.

[25] J. Arthur et al., "Unity: A General Platform for Intelligent Agents," arxiv: 1809.02627, 2018.

[26] G. Brockman et al., "OpenAI Gym," arxiv:1 606.01540, 2016.

[27] C. Beattie et al., "DeepMind Lab," arxiv: 1612.03801, 2016.

[28] J. Tan et al., "Sim-to-Real: Learning Agile Locomotion for Quadruped Robots," in *Proc. Robotics: Sci. Syst.*, Pittsburgh, PA, USA, 2018.

[29] Y. Li, "Deep Reinforcement Learning," arxiv: 1810.06339, 2018.