OpenAI Gym 환경에서 A3C 와 PPO 의 실험적 분석

황규영*, 임현교**, 허주성**, 한연희*
*한국기술교육대학교 컴퓨터공학과
**한국기술교육대학교 창의융합공학협동과정
e-mail : {to6289, glenn89, chil1207, yhhan}@koreatech.ac.kr

Experimental Analysis of A3C and PPO in the OpenAI Gym Environment

Gyu-Young Hwang*, Hyun-Kyo Lim**, Joo-Seong Heo**, Youn-Hee Han*
*Dept. of Computer Science and Engineering, KoreaTech University
**Interdisciplinary Program in Creative Engineering, KoreaTech University

요 약

Policy Gradient 방식의 학습은 최근 강화학습 분야에서 많이 연구되고 있는 주제로, 본 논문에서는 강화학습을 적용시킬 수 있는 OpenAi Gym 의 'CartPole-v0' 와 'Pendulum-v0' 환경에서 Policy Gradient 방식의 Asynchronous Advantage Actor-Critic (A3C) 알고리즘과 Proximal Policy Optimization (PPO) 알고리즘의 학습 성능을 비교 분석한 결과를 제시한다. 딥러닝 모델 등 두 알고리즘이 동일하게 지닐 수 있는 조건들은 가능한 동일하게 맞추면서 Episode 진행에 따른 Score 변화 과정을 실험하였다. 본 실험을 통해서 두 가지 서로 다른 환경에서 PPO 가 A3C 보다 더 나은 성능을 보임을 확인하였다.

1. 서론

강화학습은 머신러닝의 한 범주로서 지도학습 및 비지도학습과 다르게 정답 데이터가 주어지지 않으며, 주어진 데이터만을 이용하여 학습하는 것은 아니다. 강화학습은 어떤 환경 안에서 정의된 에이전트가 현재의 상태를 토대로 보상을 최대화하는 행동을 선택할 수 있도록 결정하는 정책을 학습하는 것이다.

강화학습 전략으로는 크게 value 기반 강화학습과 policy 기반 강화학습이 있다. value 기반 강화학습은 에이전트가 value 함수를 기반으로 정책을 생성해내고 이를 통해 행동하며 value 함수를 업데이트함으로써 학습을 한다. 반면 policy 기반 강화학습은 정책을 직접 근사하고 정책을 근사한 정책신경망을 업데이트하며 학습한다.

정책신경망은 목적함수를 기준으로 업데이트가 이루어진다. 목적함수는 에이전트가 임의의 상태에서 시작하여 앞으로 받을 것으로 기대되는 보상의 합이다. 이 목적함수가 최대화되도록 정책신경망을 업데이트하는 방식을 Policy Gradient 라고 한다. Q-learning 은 Value 기반의 강화학습 알고리즘이지만 Q 값의 작은 변화에도 policy 가 크게 변할 수 있는 단점이 존재하기 때문에 policy 의 점진적인 업데이트를 통해 더 나은 policy 를 찾아가는 Policy Gradient 방식이 개발되었다.

본 논문에서는 Policy Gradient 방식의 Asynchronous Advantage Actor-Critic (A3C) [1] 알고리즘과 Proximal Policy Optimization (PPO) [2] 알고리즘의 성능을 비교하는 실험을 OpenAi Gym 환경에서 진행한다. OpenAi Gym 의 'CartPole-v0' 환경은 이산적인 행동을 학습하는 실험을 진행하고 'Pendulum-v0' 환경에서는 연속적인 행동을 학습하는 실험을 진행한다.

본 논문의 2 장에서는 강화학습 알고리즘인 A3C 와 PPO 에 대해서 간략히 설명하며, 3 장에서는 A3C 와 PPO 를 비교하는 실험을 진행한다. 마지막 4 장에서 결론을 서술한다.

2. 강화학습 알고리즘

2.1 A3C

Actor-Critic 은 value 기반 강화학습과 policy 기반 강화학습의 두가지 이점을 결합한 것이다. Actor 가 정책을 근사하여 정책신경망을 업데이트하며 행동을 결정한다면 Critic 은 value 함수를 업데이트하며 선택된 행동을 평가하는 역할을 한다. 이렇게 policy 와 value 를 나누어 학습함으로써 매 타임스텝마다 업데이트를 할 수 있게 된다.

* 교신 저자: 한연희(한국기술교육대학교)
이 논문은 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. NRF-2016R1D1A3B03933355, No. 2018R1A6A1A03025526).

A3C 알고리즘은 샘플 사이의 상관관계를 줄이기 위해 Actor-Critic 구조를 가진 여러 개의 에이전트를 사용한다. 각 에이전트는 일정 타임스텝 동안 모은 샘플을 통해 글로벌신경망을 업데이트한다. 이 과정은 여러 개의 에이전트가 비동기적으로 진행을 한다.

2.2 PPO

PPO 알고리즘은 Trust Region Policy Optimization (TRPO) [3] 알고리즘의 장점을 취하면서도 구현하기에 훨씬 더 간단하며, 조금 더 일반적인 곳에 적용할 수 있다. PPO 알고리즘의 핵심은 policy 업데이트에 제한을 두는 것이다. policy 를 업데이트할 때 새로 업데이트 된 policy 가 기존의 policy 와 너무 많이 달라지면 문제가 생기기 때문에, Clipping 기법을 적용하여 필요 이상의 policy 업데이트를 방지한다. 이렇게 함으로써 PPO 알고리즘은 다른 Policy Gradient 방식 알고리즘들의 성능을 능가한다.

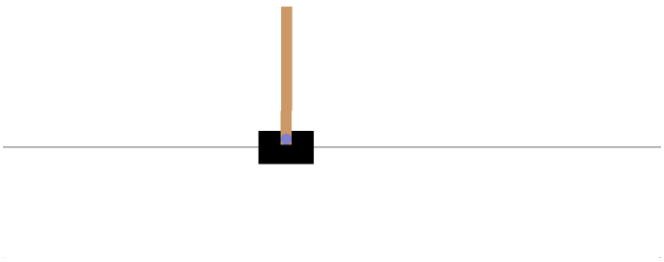
3. 실험 및 비교 분석

3.1 실험 환경

본 논문에서 A3C 와 PPO 알고리즘의 비교 실험을 진행하기위해 사용된 환경은 OpenAi Gym 이다. 이산적인 행동을 하는 환경을 실험하기 위해서 'CartPole-v0'를 연속적인 행동을 하는 환경을 실험하기 위해서는 'Pendulum-v0'를 선택하였다.

3.1.1 CartPole-v0

(그림 1)에서 볼 수 있듯이 이 환경은 pole, cart, joint, 그리고 track 으로 구성되어있다. pole 은 cart 에 joint 로 연결되어 있어서 joint 를 중심으로 자유롭게 회전할 수 있다. cart 는 track 위를 마찰없이 움직일 수 있다. 'CartPole-v0' 게임은 pole 이 위로 올라간 상태에서 시작하고 cart 를 좌우로 움직이며 pole 이 아래로 떨어지지 않도록 유지하는 것이 목표이다. 이 환경은 Barto 에 의해 기술된 cart-pole 문제와 일치한다 [4].



(그림 1) CartPole-v0

이 환경에서 agent 는 <표 1>과 같이 4 가지의 관측값을 얻어 state 로 사용한다. Cart Position 과 Cart Velocity 는 카트의 수평선 상의 위치와 속도를, Pole Angle 과 Pole Velocity At Tip 은 폴과 수직선 상의 각도와 폴 끝의 각속도를 나타낸다. Agent 는 cart 를 왼쪽이나 오른쪽으로 미는 2 가지의 이산적인 행동을 수행한다. 보상은 pole 이 위로 올라간 상태에서 매 timestep 마다 +1 이 주어진다.

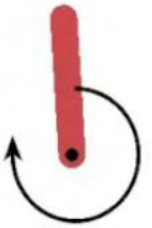
에피소드의 종료 조건으로는 첫째, Pole Angle 이 ±12°를 넘거나 둘째, Cart Position 이 ±2.4 를 넘거나 (cart 의 중심이 화면의 끝에 도달) 셋째, timestep 의 길이가 200 이 넘을 때이다. 만약 195 이상의 보상을 받은 에피소드가 100 회 연속으로 나온다면 'CartPole-v0 문제를 해결했다'라고 간주한다 [5].

<표 1> CartPole-v0 환경에서의 Observation

num	Observation	Min	Max
0	Cart Position	-2.4	2.4
1	Cart Velocity	-Inf	Inf
2	Pole Angle	~ -41.8°	~ 41.8°
3	Pole Velocity At Tip	-Inf	Inf

3.1.2 Pendulum-v0

'Pendulum-v0' 환경(그림 2)에서는 펜듈럼이 joint 를 중심으로 마찰없이 자유롭게 회전할 수 있다. 이 게임은 펜듈럼이 -π ~ π 사이의 임의의 각도와 -1 ~ 1 사이의 임의의 속도로 시작하고 이를 좌우로 움직이며 세운 뒤 최소한의 힘을 가하며 떨어뜨리지 않는 것이 목표이다 [6].



(그림 2) Pendulum-v0

이 환경에서 agent 의 state 로는 <표 2>와 같이 3 가지의 관측 값으로 정의 되어있다. 여기서 theta 는 펜듈럼과 수직축 사이의 각도를 지칭한다. 또한 Agent 의 action 은 펜듈럼을 회전시키기 위한 힘을 가하는 것인데, -2.0 ~ 2.0 사이의 연속적인 값을 가진다. 보상은 theta 에 의해 계산이 되어 -16.2736044 ~ 0 사이의 값이 매 timestep 마다 주어진다.

'Pendulum-v0' 환경에서는 에피소드의 종료 조건이 명시되어있지 않아 'CartPole-v0' 환경과 마찬가지로 timestep 의 길이가 200 이 넘으면 종료하도록 설정하였다. 그리고 -400 이상의 보상을 받은 에피소드가 10 회 연속으로 나오면 학습을 종료하였다.

<표 2> Pendulum-v0 환경에서의 Observation

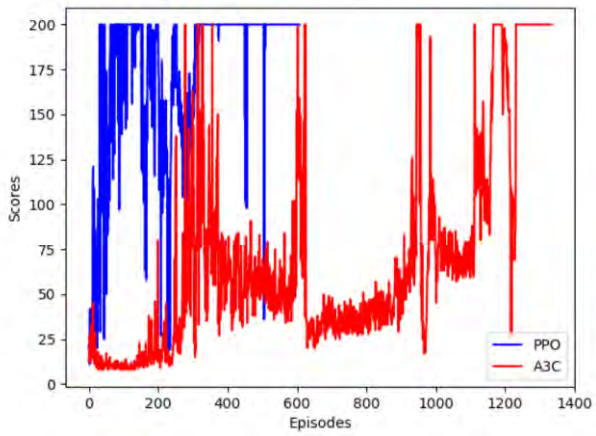
num	Observation	Min	Max
0	cos(theta)	-1.0	1.0

1	sin(theta)	-1.0	1.0
2	theta dot	-8.0	8.0

3.2 실험 조건

A3C 알고리즘과 PPO 알고리즘을 비교하는 실험을 진행하기 위해 동일하게 맞춘 요건들은 다음과 같다. 쓰레드는 4 개로 맞추었고, 신경망 모델은 Multilayer Perceptron 를 사용, 은닉층은 3 개, 뉴런수는 각각 32, 16, 8 개로 신경망을 구성하였다. 은닉층의 활성화 함수로는 ELU 를 사용하였고 정책신경망의 출력층 활성화 함수로는 Softmax 를 사용하였다. discount factor 는 0.99, 옵티마이저는 AdamOptimizer(learning_rate=1e-3, epsilon=1e-8)를 사용하였다.
PPO 알고리즘의 Clipping 기법을 위해 사용되는 하이퍼파라미터 ε 은 [2]에서 제시한 최적의 값인 0.2 로 설정하여 실험을 진행하였다.

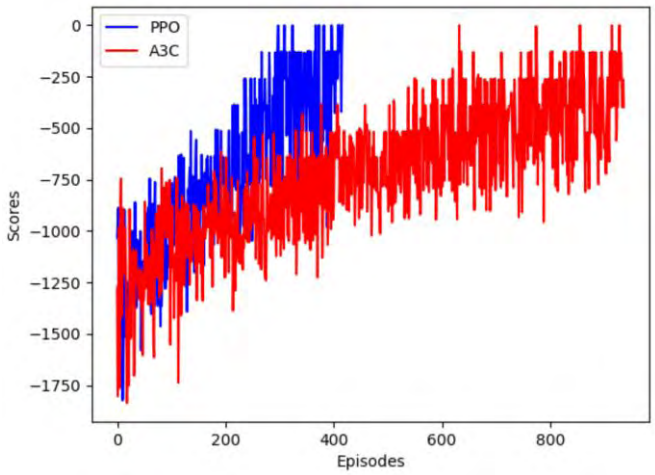
3.3 실험 결과



(그림 3) 'CartPole-v0' 환경에서 A3C 알고리즘과 PPO 알고리즘 학습 결과

(그림 3)에서 PPO 알고리즘의 그래프가 600 Episode 이후로 보이지 않는데, 그 이유는 600 Episode 가 문제해결 조건인 195 이상의 보상이 100 회의 에피소드동안 연속으로 나온 Episode 이기 때문이다. y 축은 Scores 로 각 에피소드마다 받은 보상을 나타내므로 이 학습 완료 조건을 한눈에 알아볼 수 있다. 반면, A3C 알고리즘은 1300 Episode 가 넘게 진행된 것을 볼 수 있다. 또한, 이 실험에서 PPO 는 57 초가 A3C 는 105 초가 소요되었다.

(그림 4)는 'Pendulum-v0'환경에서 A3C 알고리즘과 PPO 알고리즘 학습 비교 결과를 보여준다. 본 그림에서 알 수 있듯이 PPO 알고리즘은 400 Episode 일 때에 학습이 완료되는 반면에, A3C 알고리즘은 800 Episode 일 때 학습이 완료되는 것을 알 수 있다. 이 그래프를 통해 PPO 가 A3C 보다 빠른 시간 안에 더 많은 보상을 받고있다는 것을 알 수 있다. 마지막으로 이 실험에서의 소요시간은 PPO 가 54 초, A3C 는 111 초가 걸렸다.



(그림 4) 'Pendulum-v0' 환경에서 A3C 알고리즘과 PPO 알고리즘 학습 결과

4. 결론

본 논문에서는 최근 강화학습 분야에서 많이 사용되고 있는 Policy Gradient 방식으로 구현된 A3C 알고리즘과 PPO 알고리즘을 OpenAi Gym 의 'CartPole-v0' 과 'Pendulum-v0' 환경에서 비교하는 실험을 진행하였다. 비록 여러 환경에서 다양한 실험을 진행하진 않았지만 본 논문에서 실험을 진행한 이산적인 행동이나 연속적인 행동을 하는 환경 모두에서 PPO 알고리즘이 A3C 알고리즘에 비해 좋은 성능을 낸다는 것을 알 수 있었다.

참고문헌

[1] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Tim Harley, Timothy P. Lillicrap, David Silver, Koray Kavukcuoglu, "Asynchronous Methods for Deep Reinforcement Learning," arXiv:1602.01783v2, 2016.

[2] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov, "Proximal Policy Optimization Algorithms," arXiv:1707.06347v2, 2017.

[3] John Schulman, Sergey Levine, Philipp Moritz, Michael Jordan, Pieter Abbeel, "Trust Region Policy Optimization," arXiv:1502.05477v5, 2017.

[4] A. G. Barto, R. S. Sutton, C. W. Anderson, "Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problem," IEEE Transactions on Systems, Man, and Cybernetics, 1983.

[5] BigBadBurrow, "CartPole v0", accessed Mar 27, 2019. https://github.com/openai/gym/wiki/CartPole-v0.

[6] Chao Wen, "Pendulum v0", accessed April 5, 2019. https://github.com/openai/gym/wiki/Pendulum-v0.