시뮬레이션기반 자율주행 환경을 위한 정책 최적화 강화학습 비교

심우일, 박태화 °, 김경중*
세종대학교 컴퓨터공학부
piranas2067@gmail.com, dolavvv@naver.com, kimkj@sejong.ac.kr

Comparison of Policy Optimization Reinforcement Learning for Simulated Autonomous Car Environment

Wooil Shim, Tae-Hwa Park°, Kyung-Joong Kim*
Department of Computer Science and Engineering, Sejong University

요 약

자율 주행분야에서 심층 강화학습을 사용하여 문제를 해결하려는 시도가 늘어나고 있다. 하지만 심층 강화학습 알고리즘은 그 특성에 따라 학습되는 방향과 성능이 크게 차이가 난다. 이 때문에 자율주행에 심층 강화학습을 적용하기 전, 각 알고리즘을 검증해 접근방법에 따라 더 좋은 성능을 내는 알고리즘을 찾아야 한다. 본 논문에서는 가치기반 강화학습이 아닌 정책기반 방법인 정책 최적화 기법인 DDPG(Deep Deterministic Policy Gradient) 와 PPO(Proximal Policy Gradient)를 적용하였다. 비교를 위해 TORCS(The Open Racing Car Simulator)시뮬레이터를 사용하였고, 실험 결과 제시한 주행 환경에서 PPO 가 DDPG 보다 향상된 성능을 보여주었다.

1. 서 론

최근 자율 주행에 대한 연구가 활발하게 진행되고 있다. 자율 주행은 실제 자동차를 이용해서 바로 학습하기가 힘들기 때문에 시뮬레이터 환경에서 많이 이루어지고 있다. 그 이유는 자율주행 기술을 접목 시키기 위한 자동차를 구하기 힘들 뿐 아니라 그에 맞는 환경을 설정하는 것이 힘들기 때문이다. 하지만 시뮬레이터 환경에서는 이러한 제약 없이 다양한 실험을 수행 할 수 있기 때문이다. 다양한 자율주행 시뮬레이터 중에서 TORCS(The Open Racing Car Simulator) 시뮬레이터를 이용한 강화학습 연구가 활발한데 그 이유는 환경이 간단하고, 환경에서 제공하는 센서들의 상세한 정보를 제공받을 수 있기 때문이다.

또한 심층 강화학습 알고리즘은 가지고 있는 특징에 따라서 학습되는 방향과 성능의 차이가 있다. 자율주행 시뮬레이터에서 안정된 성능을 내는 에이전트를 만들기 위해서는 다양한 알고리즘 실험을 통해서 알고리즘의 성능을 비교하는 작업이 선행되어야 한다. 알고리즘을 비교하기 위해서는 자율 주행에서 정의한 문제를 해결하는 알고리즘을 찾아야 한다. 자율 주행은 연속적인 행동을 제어하는 문제를 해결하는 것이 목표이다. 이를 위해서 심층 강화학습 알고리즘 중 연속적인 행동을 제어하는 알고리즘의 성능 비교가 필요하다. 대표적인 연속적인 행동 제어 알고리즘은 DDPG(Deep Deterministic Policy Gradient) 알고리즘[2]과 PPO(Proximal Policy Optimization) 알고리즘[3]이 있다.

본 논문에서는 TORCS 라는 자율주행 시뮬레이터에서 심층 강화학습 알고리즘 중에서 대표적인 연속적인 행동 제어 알고리즘인 정책 기반 방법의 DDPG 알고리즘과 PPO 알고리즘의 성능을 비교 분석하였다. 시뮬레이터와 실험에 사용한 알고리즘에 대한 자세한 설명은 2,3 장에서 기술한다.

2. 배 경

2.1 DDPG 알고리즘

2016 년 심층 강화학습 알고리즘에서 의미 있는 결과를 보여준 것은 DQN(Deep Q Network)[4,5]밖에 없었고, DQN 은 연속행동을 다루기 어려웠다. 이를 해결하기 위해 만든 알고리즘이 DDPG(Deep Deterministic Policy Gradient) 이다[2].

DDPG 는 actor 와 critic 이라는 2 개의 신경망으로 구성한다. Actor 는 최적의 행동을 학습하고, critic 은 앞으로 얻을 보상의 기댓값을 최대화하는 방향으로 학습을 한다. 환경과 상호작용을 하며 그 과정에서 얻는 (s_i, a_i, r_i, s_{i+1}) 쌍을 이용해 학습을 한다.

$$y_i = r_i + \gamma Q\,(S_{i+1}, \mu\,(S_{i+1}|\theta^{\mu}\,) \vee \theta^{Q}$$

수식 1. DDPG 의 target 정의

$$L = \frac{1}{N}\sum_{i}(y_i - Q(S_i, a_i|\theta^{Q}))^2$$

수식 2. DDPG 에서 critic loss 의 정의

critic 의 학습은 수식 2 에서와 같이 replay memory 에 저장된 값들을 이용해 바로 계산, 학습이 가능하다.

$$\nabla_{\theta^{\mu}} J \approx \frac{1}{N}\sum_{i} \nabla_a Q(s, a|\theta^{Q})|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^{\mu}} \mu(s|\theta^{\mu}) \vee s_i$$

수식 3. DDPG 에서 actor loss 의 정의

하지만, actor 는 target 값이 없기 때문에 수식 3 에서와 같이 Q 값을 미분하여 target 을 구해 학습을 한다.

2.2 PPO 알고리즘

2017 년, 발표된 강화학습 알고리즘 중 TRPO(Trust Region Policy Optimization)는 성능은 좋았지만 계산량이 많아 학습이 느린 단점이 있었다. 이 문제를 해결하고 성능이 TRPO 와 비슷한 알고리즘이 PPO 이다[3].

PPO 는 DDPG 와 같이 Actor-critic 구조를 가진다. 구조가 완전히 동일하지는 않으며, DDPG 는 critic 의 출력이 Q 값이지만, PPO 는 V 값이라는 차이가 있다.

PPO 의 학습과정은 TRPO 이전 알고리즘과 많이 다르다. 이전 알고리즘들은 한 스텝 또는 한 에피소드 단위로 얻는 보상을 기준으로 신경망을 업데이트 했다. 이전 정책의 성능에 관계없이, 지금 얻은 보상 만으로 업데이트를 한 것이다. PPO 는 새로 계산된 정책이 이전 정책보다 좋지 않으면, 업데이트를 하지 않는다. 이에 따라 기존의 알고리즘보다 안정적으로 학습을 하게 된다.

3. TORCS 시뮬레이터

그림 1 TORCS 시뮬레이터 화면

TORCS(The Open Racing Car Simulator)는 open source 기반 자동차 경주 시뮬레이터[6]로 자율주행 연구 및 게임 등에 사용하고 있다. 다양한 트랙과 경주용 자동차를 제공하고 있으며, 사용자 목적에 맞게 수정이 가능하다. TORCS 의 가장 큰 장점은 주행 차량의 상태 정보와 환경에 대한 구체적인 정보들을 가지고 학습에 사용할 수 있다는 것이다. 예를 들어 현재 차량이 주행하는 속도, 도로의 꺾임 정도 등 다른 시뮬레이터들에 비해 사용 가능한 정보들이 다양하다. 이를 통해 현재 제어하는 자동차의 속도나 회전을 정확하게 조절할 수 있다.

실험을 진행하기 위해서 TORCS 시뮬레이터에서 제공하는 정보들 가운데 총 35 개의 정보를 사용하였으며, 자동차의 회전, 브레이크 페달, 가속 페달을 행동으로 정의하였다.

4. 실험

본 논문에서는 DDPG 알고리즘과 PPO 알고리즘에 대해 각각 3 번의 학습을 진행하였다. 각 학습은 총 1000 에피소드로 이루어 졌으며, 한 에피소드는 30 ~1024 스텝 사이로 이루어져 있다.

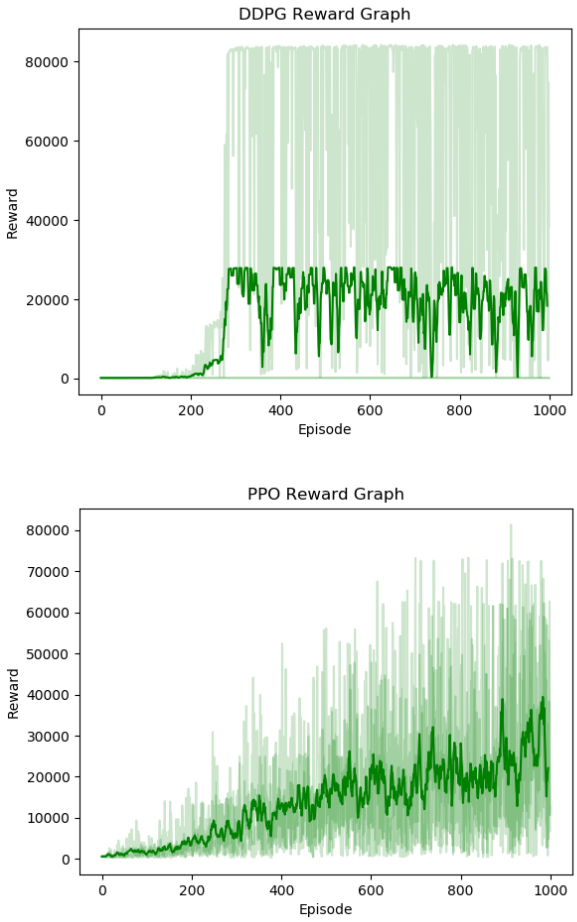

그림 2 DDPG 알고리즘(상단), PPO 알고리즘(하단) 실험 결과 그래프

그림 2 는 DDPG 알고리즘과 PPO 알고리즘의 실험 결과이다. 그림 2 에서 DDPG 알고리즘은 2 번의 실험에서 1000 번의 에피소드를 학습하는 동안, 학습이 이루어지지 않았다. 그러나 다른 한 번은 학습이 매우 빠르게 수렴해서 1024 스텝을 기준으로 최고 점수를 얻었다. 이것은 DDPG 의 학습 과정에서의 특징 때문이다. DDPG 는 탐험을 위해 신경망에서 출력된 행동에 무작위 값인 노이즈를 더한다. 이 값이 더해진 행동 값이 리플레이 메모리에 저장되어 학습에 이용한다. 학습의 방향이 노이즈에 영향을 크게 받기 때문이다.

반면 PPO 는 DDPG 와 달리 점진적으로 성능이 증가하는 모습을 보인다. 이는 PPO 의 업데이트 방식이 기존의 모델과 현재 모델을 비교해, 더 좋은 모델을 남겨두는 형태로 동작하기 때문이다. 그림 3 에서 나타나듯이 실험 중간에는 DDPG 와 PPO 의 성능이 비슷해 보이지만, 이후 PPO 의 성능이 월등하게 높아졌다.

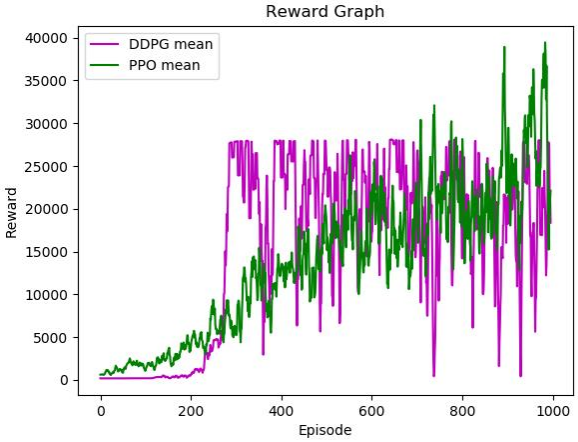

그림 3 알고리즘으로 얻은 평균값 비교 그래프

5. 결론

본 논문에서는 자율주행 시뮬레이터 인 TORCS 에서 자율 주행을 위한 연속적인 제어 문제를 해결하는 DDPG 알고리즘과, PPO 알고리즘을 비교하였다. 학습 과정이 무작위 값에 크게 영향을 받는 DDPG 에 비해 신경망의 성능을 비교해 성능이 더 좋은 쪽을 선택하는 PPO 가 더 안정적으로 학습을 수행하기 때문에 안정적인 학습이 필요한 자율주행에서는 DDPG 보다 PPO 가 더 좋은 학습 알고리즘이라 할 수 있다.

6. 감사의 글

이 논문은 2017 년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (2017R1A2B4002164). *교신저자

참고 문헌

[1] 최두섭, 안택현, 안경환, 최정단. "자율주행을 위한 TORCS 기반 End-to-End 학습." 대한전자공학회 학술대회, (2017.11): 740-743.

[2] Lillicrap, Timothy Paul, et al. "Continuous control with deep reinforcement learning." U.S. Patent Application No. 15/217,758.

[3] Schulman, John, et al. "Proximal policy optimization algorithms." *arXiv preprint arXiv:1707.06347* (2017).

[4] Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." *arXiv preprint arXiv:1312.5602* (2013).

[5] Mnih, Volodymyr, et al. "Human-level control through deep reinforcement learning." *Nature* 518.7540 (2015): 529.

[6] Wymann, Bernhard, et al. "Torcs, the open racing car simulator." *Software available at http://torcs. sourceforge. net* 4 (2000):