The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 20, No. 1, pp.171-176, Feb. 29, 2020. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2020.20.1.171
JIIBC 2020-1-24

2D 레이싱 게임 학습 에이전트를 위한 강화 학습 알고리즘 비교 분석

Comparison of Reinforcement Learning Algorithms for a 2D Racing Game Learning Agent

이동철*

Dongcheul Lee*

요 약 강화 학습은 인공지능 에이전트가 비디오 게임을 학습할 때 가장 효과적으로 사용되는 방법이다. 강화 학습을 위해 여지껏 많은 알고리즘들이 제시되어 왔지만 알고리즘마다 적용되는 분야에 따라 다른 성능을 보여주었다. 본 논문은 최근 강화 학습에서 주로 사용되는 알고리즘들의 성능이 2D 레이싱 게임에서 어떻게 달라지는지 비교 평가한다. 이를 위해 평가에서 사용할 성능 메트릭을 정의하고 각 알고리즘에 따른 메트릭의 값을 그래프로 비교하였다. 그 결과 ACER (Actor Critic with Experience Replay)를 사용할 경우 게임의 보상이 다른 알고리즘보다 평균적으로 높은 것을 알 수 있었고, 보상 값이 가장 낮은 알고리즘과의 차이는 157%였다.

Abstract Reinforcement learning is a well-known method for training an artificial software agent for a video game. Even though many reinforcement learning algorithms have been proposed, their performance was varies depending on an application area. This paper compares the performance of the algorithms when we train our reinforcement learning agent for a 2D racing game. We defined performance metrics to analyze the results and plotted them into various graphs. As a result, we found ACER (Actor Critic with Experience Replay) achieved the best rewards than other algorithms. There was 157% gap between ACER and the worst algorithm.

Key Words : Game, Deep Learning, Reinforcement Learning

Ⅰ. 서 론

강화 학습(Reinforcement learning)은 딥러닝 에이전트(Deep Learning Agent)가 비디오 게임을 학습할 때 가장 효과적으로 사용되는 방법이다[1]. 강화 학습에서는 에이전트(Agent), 환경(Environment), 상태(State), 액션(Action), 보상(Reward) 등의 개념이 사용된다. 에이전트란 알고리즘에 따라 특정 액션을 취하는 주체를 뜻한다. 여기서 액션은 에이전트가 택할 수 있는 모든 움직임을 뜻한다. 환경은 에이전트가 움직일 수 있고 그 움직임에 반응하는 것을 말한다. 환경은 현재 상태와 에이전트의 액션을 입력받고 보상과 다음 상태를 반환한다. 여기서 상태란 환경에 의해 반환되는 특정 상황을 뜻하는 것으로 에이전트와 해당 시점에 연관되는 것을 뜻한

*종신회원, 한남대학교 멀티미디어공학과
접수일자: 2019년 12월 20일, 수정완료: 2020년 1월 20일
게재확정일자: 2020년 2월 7일

Received: 20 December, 2019 / Revised: 20 January, 2020 / Accepted: 7 February, 2020
Corresponding Author: jackdclee@hnu.kr
Dept. of Multimedia Engineering, Hannam University, Korea

다. 보상은 주어진 상태에서 에이전트의 액션에 대한 성공 또는 실패를 나타내는 값이다. 2D 레이싱 게임에서 에이전트는 자동차가 되고, 환경은 자동차가 달리는 도로와 다른 경쟁 자동차들이다. 액션은 왼쪽과 오른쪽으로 이동하는 것과 가속 또는 감속하는 것이다. 보상은 다른 자동차를 한 대씩 앞지를 때마다 주어진다.

강화 학습을 위해서 사용되는 상태 중에서 어떤 특징(Feature)을 사용할지 연구자가 직접 지정해 줄 수도 있으나 최근에는 주로 게임 화면의 픽셀 (Pixel) 정보를 신경망의 입력값으로 사용하는 추세이다. 신경망에서 이미지를 처리하기 위해서는 주로 CNN (Convolutional Neural Network)을 사용한다[2]. CNN은 이미지를 입력받아 이미지 내의 다양한 개체에 대하여 중요성을 부여하고 이를 통해 다른 개체와 구별해 낼 수 있는 능력을 제공한다.

또한, 게임의 종류에 따라 사용자가 액션을 취하자마자 보상이 주어지는 것이 아니라 일정한 시간이 흘러야 주어지는 경우가 있다. 예를 들어 레이싱 게임의 경우 가속을 하거나 방향을 변경할 경우 일정 시간이 흘러야 다른 자동차를 추월할 수 있고 이때 점수가 올라가는 것과 같다. 이러한 게임을 학습할 경우에는 이전 상태들을 일정 기간 기억해 두었다가 추후 어떤 액션을 하는 것이 좋을지 결정할 때 사용하는데 이런 방법을 구현한 알고리즘이 RNN (Recurrent Neural Network)이나 LSTM (Long Short Term Memory)이다[3, 4].

강화 학습에 최근 많이 사용되는 알고리즘으로 A3C (Asynchronous Advantage Actor-Critic), ACER (Actor Critic with Experience Replay), PPO (Proximal Policy Optimization) 등이 있다[5, 6, 7]. 이 중 어떤 알고리즘을 사용하느냐에 따라 에이전트의 학습 성능이 크게 좌우된다. 그러나 어떤 게임을 학습하는지에 따라 알고리즘의 우위가 달라지며 아직 모든 게임에서 우수하게 동작하는 알고리즘은 알려지지 않았다. 본 논문은 2D 레이싱 게임을 학습하기 위해 각 알고리즘 별 에이전트를 구현하고 어떤 알고리즘을 사용하는 것이 학습에 유리한지 성능을 비교 평가하고자 한다.

본 논문은 다음과 같은 구성이다. 2장에서는 본 논문에서 비교할 알고리즘에 대하여 각각의 특징을 알아본다. 3장에서는 에이전트의 성능을 비교할 방법에 대하여 정의하고 4장에서는 성능 평가 결과를 분석한다. 마지막으로 5장에서는 결과를 종합하고 결론을 제시한다.

II. 관련 연구

1. Asynchronous Advantage Actor-Critic (A3C)

A3C는 병렬적으로 모델을 학습시키는데 초점을 맞춘 Off-Policy Gradient 방법이다. Critic 모델은 여러 개의 Actor 모델이 병렬적으로 학습하면서 글로벌 신경망의 파라미터를 갱신하며 동기화하는 동안 가치 함수(Value Function)을 학습한다. 업데이트하는 과정에서는 다단계 손실 함수와 엔트로피 손실 함수를 사용한다. 각 에이전트는 시간 t에 정책 (Policy) π를 따라 행동 (Action) a_t를 선택한다. 정책 π에서 상태 (State) s_t의 가치 함수는 다음과 같이 정의되며 해당 상태에서 해당 정책을 따랐을 경우 받을 수 있는 기댓값에 해당한다.

$$V_\pi(s_t) = E[R_t|s_t] \quad (1)$$

상태 s_t에서 행동 a_t를 따르는 것이 다른 행동을 따르는 것보다 얼마나 더 좋은 결정인지 판단하기 위하여 보상 함수 (Advantage Function)를 다음과 같이 정의한다.

$$A_\pi(s_t, a_t) = Q_\pi(s_t, a_t) - V_\pi(s_t) \quad (2)$$

여기서 $Q_\pi(s_t, a_t)$는 정책 π를 따랐을 때 s_t와 a_t에 대한 기댓값 (Expected Return)을 의미하는 상태 행동 가치 함수 (State Action Value Function)로써 다음과 같이 정의된다.

$$Q_\pi(s_t, a_t) = E[r_{t+1} + \gamma V_\pi(s_{t+1})|s_t, a_t] \quad (3)$$

또한, 어떤 정책 π가 얼마나 좋은지 표현하기 위한 목적 함수 (Objective Function)를 다음과 같이 정의한다.

$$J(\theta) = E[V_\pi(s_0)] \quad (4)$$

목적 함수 $J(\theta)$을 최적화하기 위해 이 함수의 기울기를 구하는 방법은 다음과 같다.

$$\Delta_\theta J(\theta) = \Delta_\theta \log \pi_\theta(a_t|s_t) A_\pi(s_t, a_t) \quad (5)$$

2. Actor Critic with Experience Replay (ACER)

ACER은 On-Policy 모델인 A3C와는 달리 지난 에피소드에서 샘플을 뽑아 재사용하는 Experience Replay를 이용하는 Off-Policy Actor-Critic 모델이며 샘플링을 효율적으로 하면서 데이터 간에 상관관계 (Correlation)를 줄이는 방법이다. Off-Policy Estimator에 대한 안정성을 제어하기 위해 Retrace Q-value Estimation을 다음과 같이 Importance Weight가 상수 c보다 커질 수 없도록 ΔQ를 수정하도록 하였으며 다음과 같이 정의된다.

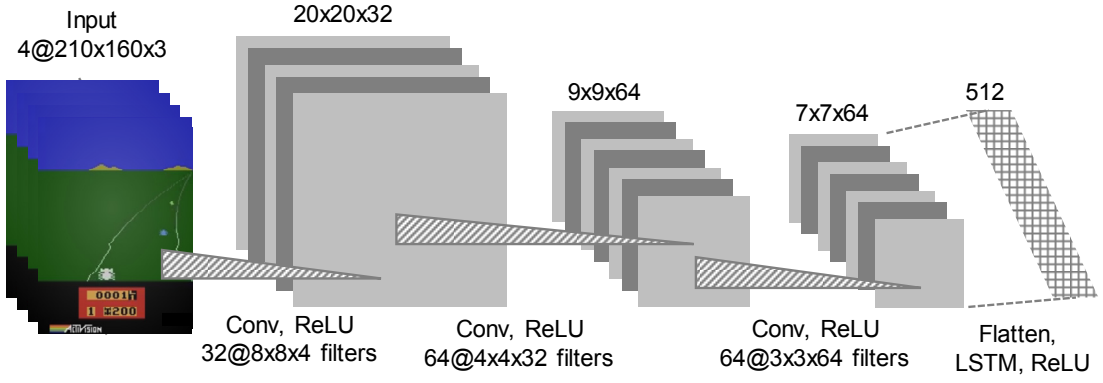

그림 1. 강화 학습 에이전트가 2D 레이싱 게임 방법을 학습하는데 사용한 딥러닝 모델
Fig. 1. Illustration of deep learning model used by the reinforcement learning agent to learn how to play a 2D racing game

$$\Delta Q^{ret}(S_t,A_t)=\gamma^t\prod_{1\le\tau\le t}\min(c,\frac{\pi(A_\tau|S_\tau)}{\beta(A_\tau|S_\tau)})\delta_t \quad (6)$$

여기서 Temporal Difference (TD) Error δ_t는 다음과 같이 정의된다.

$$\delta_t=R_t+\gamma E_{a\sim\pi}Q(S_{t+1},a)-Q(S_t,A_t) \quad (7)$$

여기서 $R_t+\gamma E_{a\sim\pi}Q(S_{t+1},a)$ 부분은 TD target 이라고 한다. 기댓값 $E_{a\sim\pi}$는 현재 정책을 따를 때 얻을 수 있는 리턴 값에 대한 가장 좋은 추측 값이기 때문에 사용되었다.

또한 Policy Gradient $\hat{g_t}$의 high variance 현상을 줄이기 위해 Importance Weight의 범위를 상수 c로 제한하고 여기에 correction term을 다음과 같이 더하였다.

$$\hat{g_t}=\min(c,w_t)(Q^{ret}(S_t,A_t)-V_w(S_t))\nabla_\theta\ln\pi_\theta(A_t|S_t) \quad (8)$$
$$+E_{a\sim\pi}[\max(0,\frac{\omega_t(a)-c}{\omega_t(a)})(Q_\omega(S_t,a)-V_\omega(S_t))$$
$$\nabla_\theta\ln\pi_\theta(a|S_t)]$$

3. Proximal Policy Optimization (PPO)

PPO는 Clipped Surrogate Objective를 사용하여서 한 스텝 내에서 정책을 변화시킬 수 있는 파라미터 갱신이 너무 자주 발생하는 것을 막아 안정성을 향상한다. 이를 위해 PPO의 목적 함수 $J(\theta)$는 Probability Ratio $r(\theta)$가 $[1-\epsilon,1+\epsilon]$ 내에서 유지될 수 있도록 Clip Function을 적용하였으며 다음과 같이 정의된다.

$$J(\theta)=E[\min(r(\theta)\widehat{A_{\theta_{old}}}(s,a), \quad (9)$$
$$clip(r(\theta),1-\epsilon,1+\epsilon)\widehat{A_{\theta_{old}}}(s,a))]$$

Probability Ratio $r(\theta)$는 이전 상태의 정책과 현재 상태의 정책의 비율을 의미하며 다음과 같이 정의된다.

$$r(\theta)=\frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)} \quad (10)$$

정책 함수와 가치 함수 간에 파라미터를 공유하는 신경망에 PPO를 사용할 때는 Exploration을 충분히 수행할 수 있도록 목적 함수에 Value Estimation에 대한 Error Term과 Entropy Term을 추가하여 다음과 같이 사용한다.

$$J'(\theta)=E[J(\theta)-c_1(V_\theta(s)-V_{target})^2 \quad (11)$$
$$+c_2H(s,\pi_\theta(\bullet))]$$

Ⅲ. 성능 평가 방법

본 논문은 2D 레이싱 게임을 플레이하기 위한 딥러닝 강화 학습 에이전트를 제시한다. 이 에이전트를 이용하면 다양한 강화 학습 알고리즘을 모듈 형태로 다양한 관점에서 평가할 수 있다. 이번 연구에서는 강화 학습에서 최근 많이 사용되는 알고리즘인 A3C, ACER, PPO를 비교한다. 신경망은 그림 1과 같이 게임 화면의 픽셀 정보를 해석하기 위한 CNN과 장기간의 정보를 기억하기 위한 LSTM을 사용하였다. CNN에 사용된 필터는 각각 32개, 64개, 64개이며, 필터 크기는 각각 8x8, 4x4, 3x3 이다. LSTM에 사용된 셀은 512개이다. 각각의 은닉층에서 사용된 활성화 함수는 ReLU이다.

게임은 OpenAI Gym에서 제공되는 Enduro v4를 사용하였다[8]. 이 게임의 특징은 여러 대의 자동차들이 경

주하는데 다른 자동차를 앞지를수록 점수가 올라가고 추월당할수록 점수가 줄어든다는 것이다. 또한, 게임 화면에 이동 거리도 표시되는데 제한된 시간 안에 이동 거리가 길어야 그만큼 많은 자동차들을 추월할 수 있으므로 둘 사이의 상관 관계가 있을 수 있다.

실험 환경은 Nvidia GeForce GTX 1080ti GPU가 설치된 Ubuntu 18.04 머신에서 구축되었다. 에이전트를 만들기 위해 OpenAI Gym 0.13, Tensorflow 1.10, Keras 2.2, Python 3.6을 활용한 프로그램을 작성하였다. 에이전트는 학습을 위해 알고리즘별로 $1x10^7$번의 타임 스텝을 처리하였고 게임 화면 프레임은 4개를 하나의 스택으로 처리하여 벡터 연산을 하였다. 성능 평가에서 사용된 알고리즘에서 사용된 하이퍼 파라미터는 표1과 같다.

표 1. 각 알고리즘에서 사용된 하이퍼 파라미터 값
Table 1. The values of hyperparameters for each algorithm

Algorithm	Hyperparameters	Value
A3C	Discount factor	0.99
	N_steps	5
	Value function coefficient	0.25
	Entropy coefficient	0.01
	Learning rate	0.0007
	RMSProp decay parameter	0.99
	RMSProp epsilon	0.00001
ACER	Discount factor	0.99
	N_steps	20
	Q value coefficient	0.5
	Entropy coefficient	0.01
	Learning rate	0.0007
	RMSProp decay parameter	0.99
	RMSProp epsilon	0.00001
	Buffer size	5000
	Replay ratio	4
PPO	Discount factor	0.99
	N_steps	128
	Entropy coefficient	0.01
	Learning rate	0.00025
	Value function coefficient	0.5
	Number of training minibatches	4
	Number of epoch	4
	Clipping parameter	0.2
	The maximum value for the gradient clipping	0.5

Ⅳ. 성능 평가

각 알고리즘이 에이전트가 2D 레이싱 게임을 플레이하는데 끼치는 영향을 알아보기 위하여 게임이 끝났을 때 점수인 총 보상 값과 게임을 얼마나 오래 플레이할 수 있었는지 나타내주는 게임 시간 (Length)을 알고리즘별로 비교하였다.

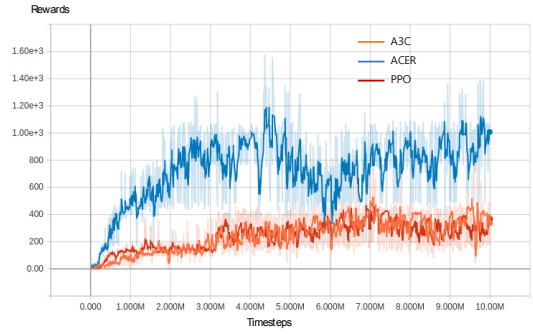

그림 2. 타임 스텝에 따른 알고리즘 별 보상 값
Fig. 2. Rewards for each algorithm along with the time steps

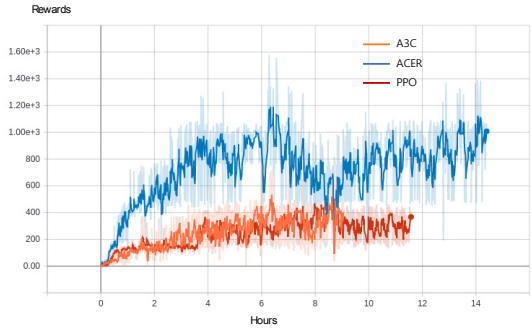

그림 3. 트레이닝 시간에 따른 알고리즘 별 보상 값
Fig. 3. Rewards for each algorithm along with the training time

그림 2는 트레이닝 (Training) 시 각 알고리즘을 10M타임 스텝 실행하였을 경우 보상 값을 나타낸 그래프이다. A3C와 PPO의 경우 비슷한 트레이닝 효과를 보이며 ACER의 경우 다른 두 알고리즘보다 높은 효과를 보이는 것을 알 수 있다. 그림 3은 같은 조건에서 각 알고리즘이 10M 타임 스텝 실행되는데 얼마나 걸리는지 나타낸 그래프이다. ACER가 14시간 정도로 가장 오래 걸리고 그다음으로 PPO가 11시간, A3C가 9시간 정도로 가장 빨랐다. 그림 2와 3에서 각 그래프는 smooth factor 0.6으로 설정하여 그려졌으며 원본 그래프는 옅은 색으로 칠해졌다.

그림 4, 그림 5, 그림 6은 각 알고리즘 별로 트레이닝 시 보상 값과 게임 시간 간의 상관관계를 보여주는 그래프이다. 세 알고리즘 모두 초기에는 기본적으로 주어지는 시간만 플레이하지만 트레이닝 시간이 지날수록 게임 시간이 증가하면서 보상 값과 같은 추이를 보이는 것을 알 수 있다.

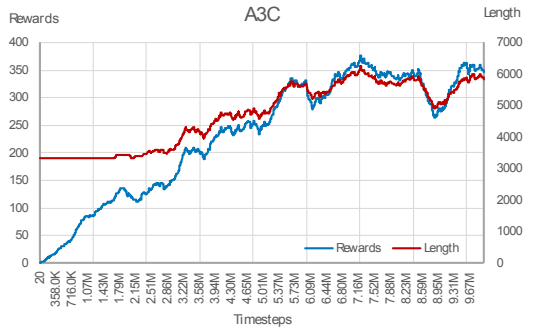

그림 4. A3C를 사용하였을 때 타임스텝에 따른 보상 값과 게임 시간

Fig. 4. Rewards and playing time for A3C along with the time steps

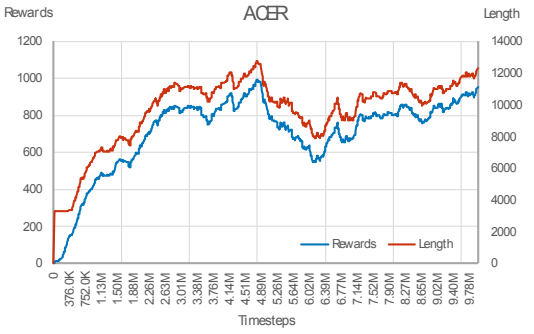

그림 5. ACER를 사용하였을 때 타임스텝에 따른 보상 값과 게임 시간

Fig. 5. Rewards and playing time for ACER along with the time steps

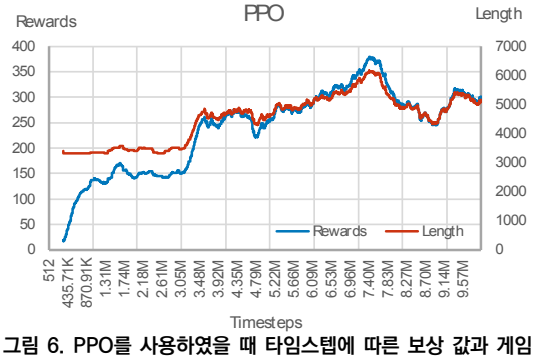

그림 6. PPO를 사용하였을 때 타임스텝에 따른 보상 값과 게임 시간

Fig. 6. Rewards and playing time for PPO along with the time steps

표2는 각 알고리즘별로 10M 타임 스텝 동안 트레이닝 후 만들어진 모델로 실제 게임을 1000회 플레이했을 경우 얻은 보상 값의 평균 및 최댓값이다. ACER를 사용했을 경우 보상 값의 평균이 297.4, 최댓값이 461로 세 알고리즘 중 가장 높은 성능을 보여주었다.

표 2. 각 알고리즘 별 평균 및 최대 보상 값

Table 2. The average and maximum rewards for each algorithm

Algorithm	Average	Max
A3C	167.7	441
ACER	297.4	461
PPO	115.7	142

V. 결 론

본 논문은 딥러닝 에이전트가 2D 레이싱 게임을 학습할 때 어떤 알고리즘을 사용하는지에 따른 성능을 평가하였다. 성능 평가 시 비교했던 알고리즘은 강화 학습에서 가장 많이 사용되는 알고리즘인 A3C, ACER, PPO를 대상으로 하였다. 성능 평가 결과 ACER를 사용할 경우 게임의 보상이 다른 알고리즘보다 평균적으로 높은 것을 알 수 있었고, 보상 값이 가장 낮았던 PPO보다 157% 높았다. 그러나 10M 타임스텝이 수행되는데 걸리는 시간은 ACER가 가장 오래 걸렸고 A3C가 가장 짧았다.

향후 연구로는 딥러닝 에이전트가 게임을 학습하는데 필요한 다른 여러 요인에 대하여 성능 평가를 할 것이다. 이를 통해 게임 스타일 별로, 또는 학습 알고리즘 별로 어떤 요소를 사용하는 것이 효과적인지 알아볼 수 있을 것이다.

References

[1] S. Mukhopadhyay, O. Tilak, S. Chakrabarti, “Reinforcement Learning Algorithms for Uncertain, Dynamic, Zero-Sum Games”, IEEE International Conference on Machine Learning and Applications, 2018.
DOI: https://doi.org/10.1109/ICMLA.2018.00015

[2] Y. Wei, W. Xia, M. Lin, J. Huang, B. Ni, J. Dong, Y. Zhao, S. Yan, “HCP: A Flexible CNN Framework for Multi-Label Image Classification”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 38, Iss. 9, pp. 1901-1907, 2016.
DOI: https://doi.org/10.1109/TPAMI.2015.2491929

[3] S.-G. Choi, W. Xu, “A Study on Person Re-Identification System using Enhanced RNN”, The Journal of The Institute of Internet, Broadcasting and Communication, Vol. 17, No. 2, 2017.
DOI: https://doi.org/10.7236/JIIBC.2017.17.2.15

[4] I.-T. Joo, S.-H. Choi, "Stock Prediction Model based on Bidirectional LSTM Recurrent Neural Network",

The Journal of KIIECT, Vol. 11, No. 2, 2018.
DOI: https://doi.org/10.17661/jkiiect.2018.11.2.204

[5] V. Mnih, A.P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning", Proceedings of the International Conference on Machine Learning (ICML), pp. 1928-1937, 2016.

[6] Z. Wang, V. Bapst, N. Heess, V. Mnih, R. Munos, K. Kavukcuoglu, N. de Freitas, "Sample efficient actor-critic with experience replay", Proceedings of the International Conference on Learning Representations, 2017.

[7] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, "Proximal policy optimization algorithms," arXiv preprintarXiv:1707.06347, 2017.

[8] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, W. Zaremba, "OpenAI Gym", arXiv:1606.01540, 2016.

저 자 소 개

이 동 철(종신회원)

- 2002년 : POSTECH 컴퓨터공학 학사
- 2004년 : POSTECH 전자컴퓨터공학 석사
- 2004년~2012년 : KT 무선연구소 책임연구원
- 2012년 : 한양대학교 전자컴퓨터 통신공학 박사
- 2012년~현재 : 한남대학교 멀티미디어공학과 교수
- 관심분야 : 딥러닝, 게임, 신경망, 알고리즘

※ 이 논문은 2019년도 한남대학교 교비학술연구비지원으로 작성되었습니다.