DMQA Seminar 20221230

Introduction to Policy Gradient

From Policy Gradient Theorem to Actor-Critic Methods

일반대학원 산업경영공학과
김재훈

고려대학교
KOREA UNIVERSITY

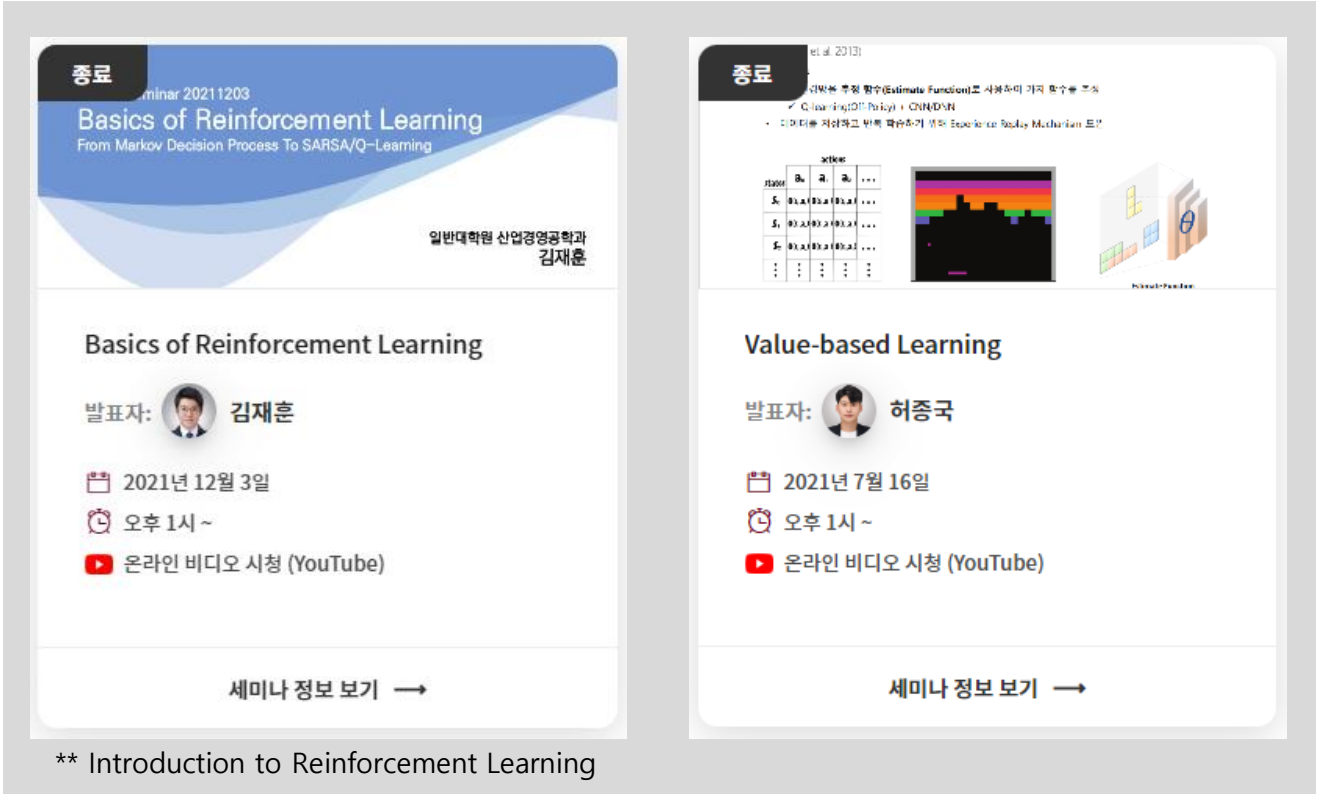
Introduction

- 발표자 소개

- 이름: 김재훈
- 학력
  - ✓ 2020.03 – 현재 | 석박사통합과정 | 고려대학교 산업경영공학과 (지도교수: 김성범)
- 연구분야
  - ✓ Self-supervised learning
  - ✓ Reinforcement learning
- e-mail : jhoon0418@korea.ac.kr

CONTENTs

- Deep Reinforcement Learning
- Policy-gradient
- Actor-Critic



** Introduction to Reinforcement Learning

참고할 수 있는 강화학습 관련 세미나

Deep Reinforcement Learning

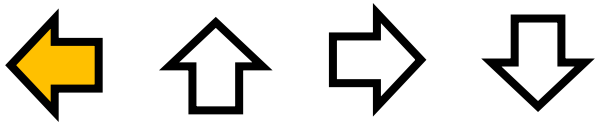
What is Reinforcement Learning?

❖ 강화학습의 목적

- 강화학습은 **순차적인 의사결정 문제**에서 **누적 보상을 최대화**하기 위해 시행착오를 거쳐서 상황에 따른 행동 정책을 학습
- 강화학습은 에이전트가 속한 상태(state), 선택한 행동(action), 행동에 따른 보상(reward)으로 구성됨

State

Action



Reward

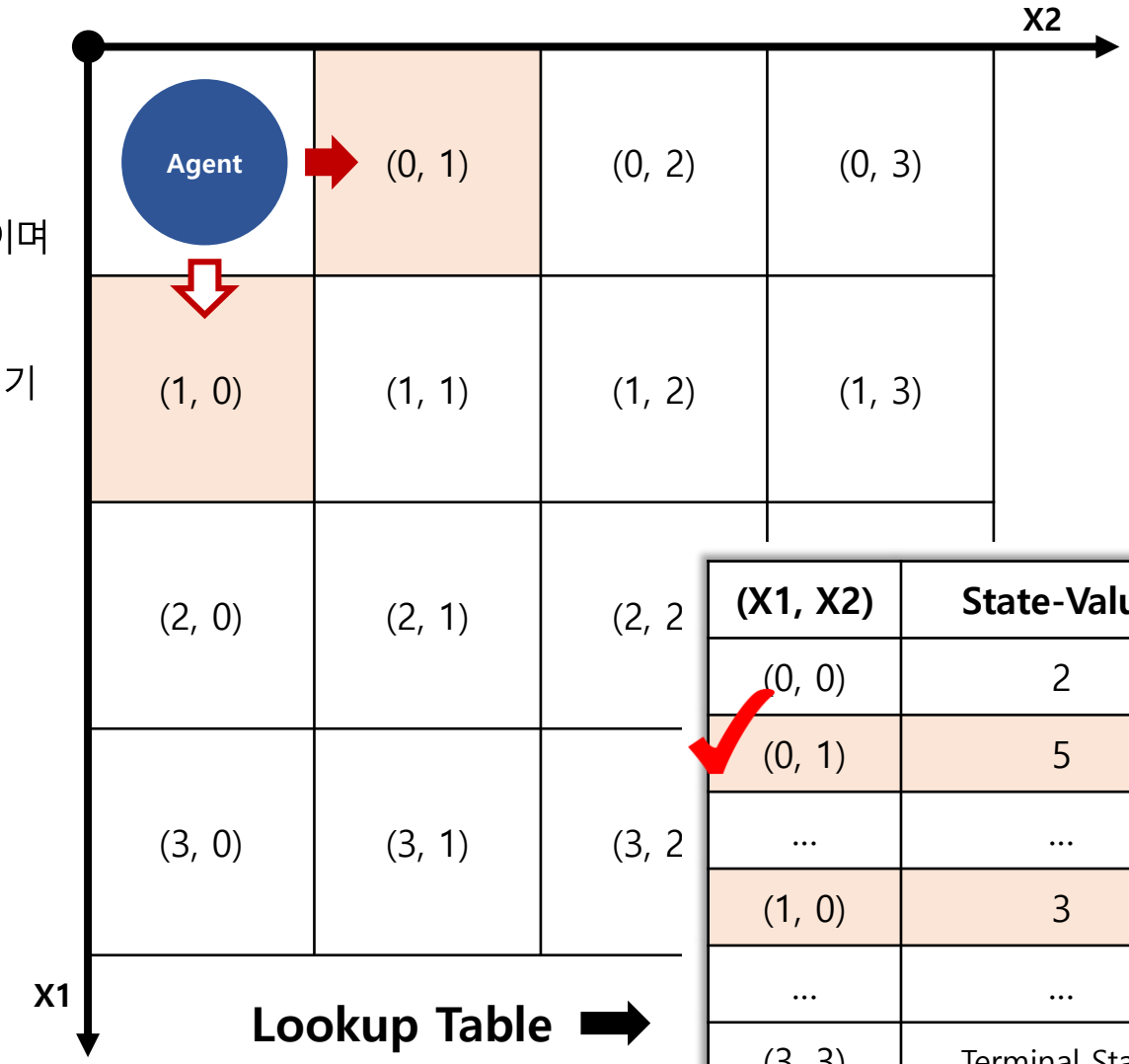
-10 / 0 / 1

Deep Reinforcement Learning

Reinforcement Learning with Tabular Method

❖ 테이블 기반 강화학습이란?

- 각 상태가 가지는 가치(value)를 테이블에 기록
- 우측의 예시는 상태 공간의 크기가 4x4 (총 16개의 고유 상태)이며 행동 공간의 크기는 4 (상, 하, 좌, 우로 이동 가능함)
- 에이전트가 획득한 경험에 따라서 테이블에 각 상태의 가치를 기록 및 정책 학습
- 가치를 기록할 공간이 고유 상태의 개수만큼 필요함
- 올바른 가치를 구하기 위해서 모든 상태를 방문해야 함



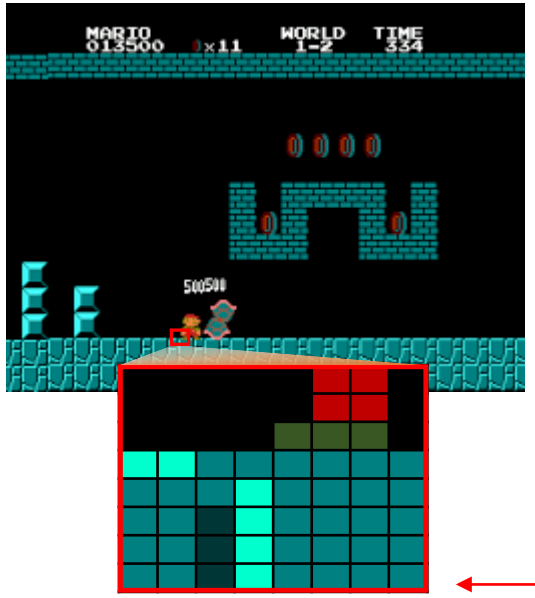
Deep Reinforcement Learning

Reinforcement Learning with Tabular Method

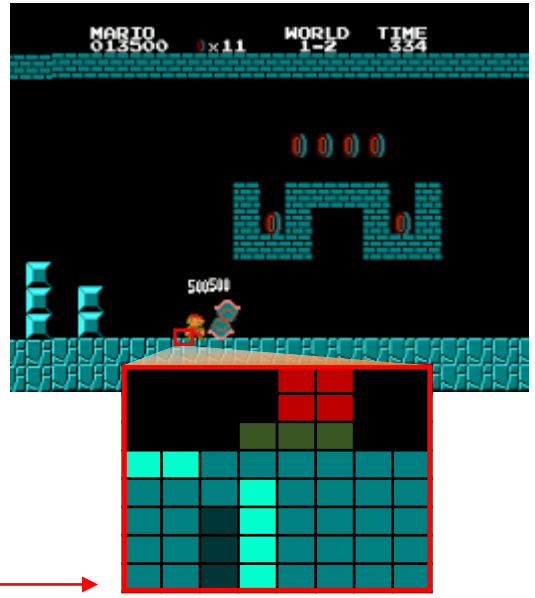
❖ 테이블 기반 강화학습의 단점

- 고유 상태의 수가 매우 크거나 무한에 가까울 경우 현실적으로 모든 가치를 저장하기 어려움
- 고유 상태의 수가 매우 크거나 무한에 가까울 경우 현실적으로 모든 상태를 방문하기 어려움

State 1



State 1758135



Pixel 몇 개의 차이더라도 서로 다른 상태로 기록

State	State-Value
0	2
1	30
…	…
1758135	30
1758136	29
…	…

**Game: Atari Super Mario Bros

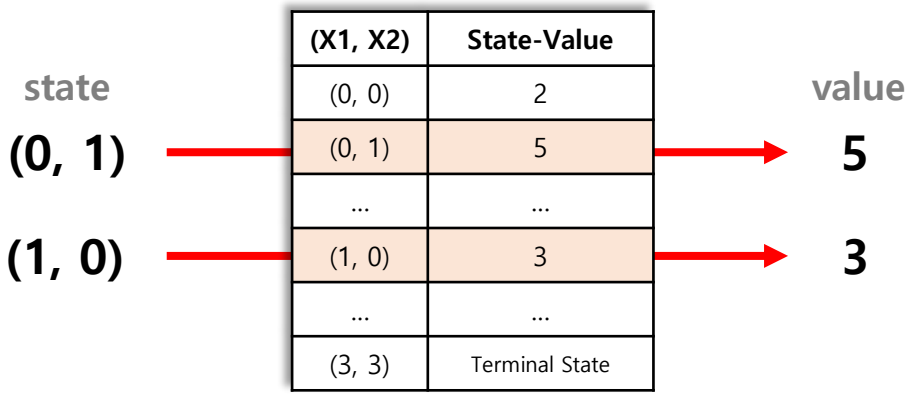
Deep Reinforcement Learning

Reinforcement Learning with Function Approximation Method

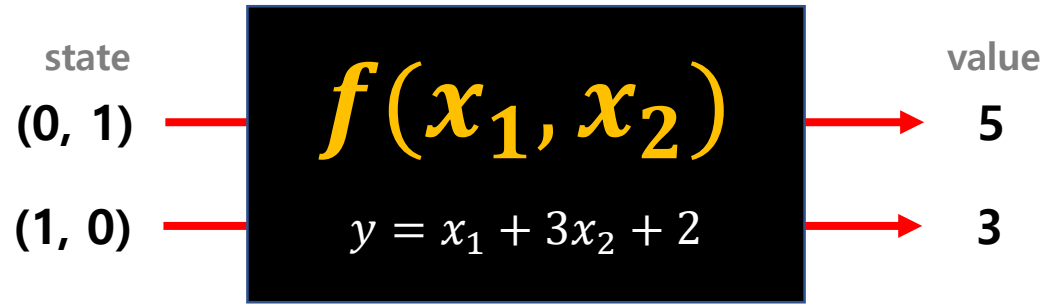
❖ 함수 기반 강화학습이란?

- 상태 정보와 보상 값을 활용하여 실제 가치를 근사하는 함수를 학습
- 테이블은 각 고유 상태에 대한 가치를 기록하는 반면, 함수는 **실제 가치를 근사할 수 있는 파라미터를 저장**

테이블 기반 방법론



함수 기반 방법론

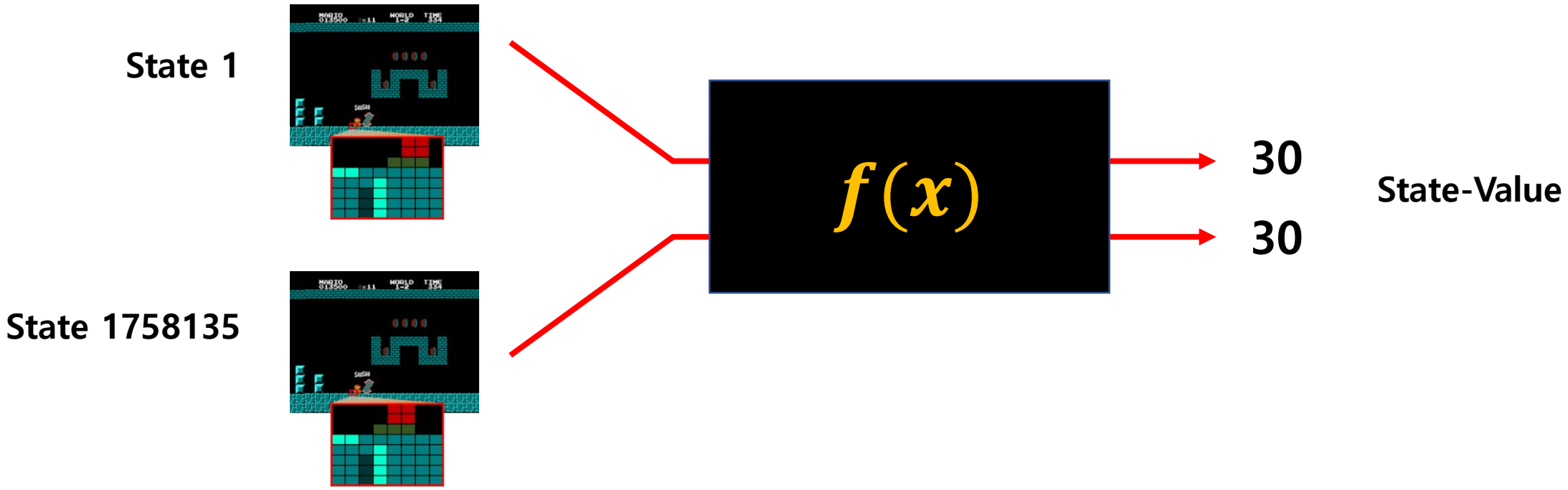


Deep Reinforcement Learning

Reinforcement Learning with Function Approximation Method

❖ 함수 기반 강화학습의 장점

- 함수의 일반화(generalization) 성질은 경험하지 못한 상태도 유사한 경험으로 추정할 수 있도록 함
- 각각의 상태를 모두 기억할 필요가 없으며, 근사한 함수의 파라미터만 기억하면 되므로 저장 공간 어려움이 없음



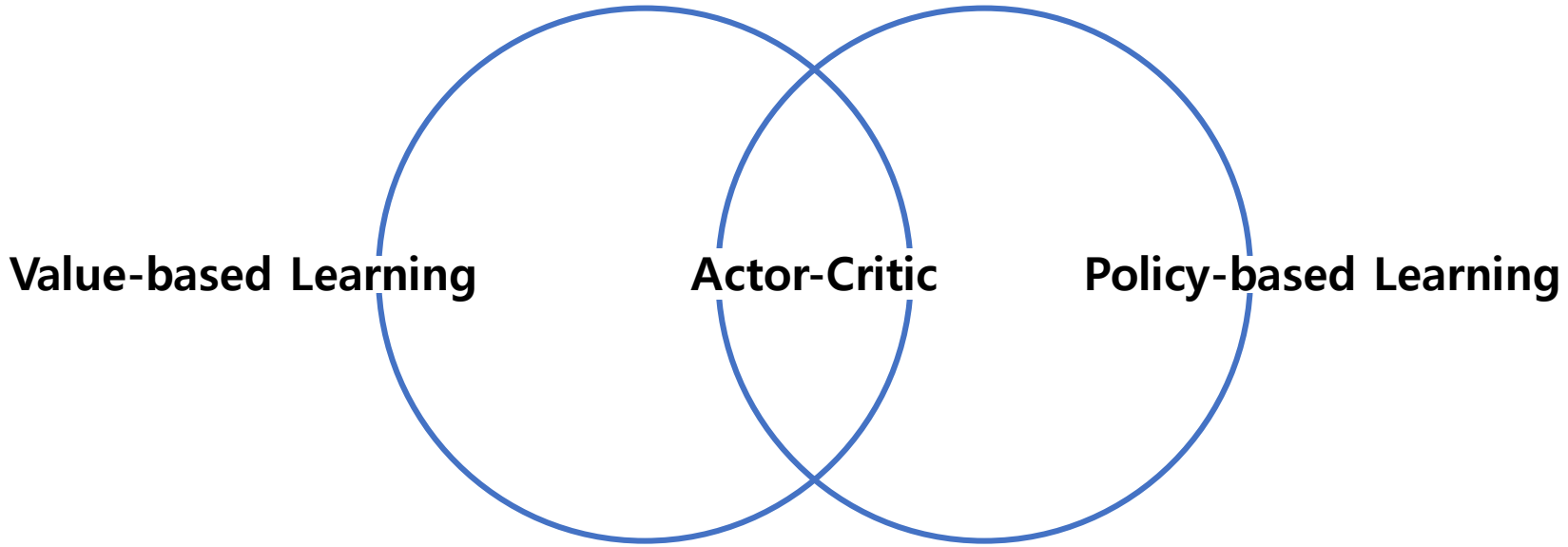
**Game: Atari Super Mario Bros

Deep Reinforcement Learning

What is Deep Reinforcement Learning?

❖ 심층 강화학습이란?

- 함수 기반 강화학습으로서 인공신경망을 접목한 방법론
- 인공신경망으로 가치함수(q_π, v_π) 혹은 정책함수(π)를 표현함에 따라서 방법론의 종류가 나뉨
- 액터-크리틱은 가치함수와 정책함수를 모두 사용하는 방법론



Deep Reinforcement Learning

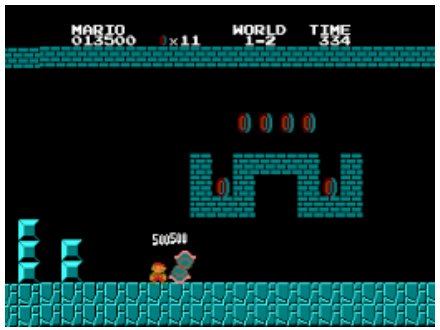
Recap. Value-based Learning

❖ 가치기반 강화학습

- 행동-상태 가치(q)에 기반하여 행동을 선택하는 방법론으로 실제 가치를 추정할 가치함수(Q_θ)를 학습
- 별도의 명시된 정책함수가 없으며 탐험에 필요한 소수의 확률을 제외하고 가장 높은 가치를 갖는 행동을 선택함 (가치함수에 의존)
- 가치함수의 업데이트는 정답 가치와 추정 가치의 차이를 줄이는 방향으로 이루어 짐(gradient descent : $\theta' \leftarrow \theta - \alpha \nabla_\theta \big(L(\theta)\big)$)

정답 가치 추정 가치

$$L(\theta) = \mathbb{E}\left[\left(r + \gamma \max_{a'} Q_\theta(s', a') - Q_\theta(s, a)\right)^2\right]$$



State
(timestep=334)

Q_θ

Action-State	Value
$(S_{334}, \leftarrow)$	8
$(S_{334}, \downarrow)$	0
$(S_{334}, \uparrow)$	0
$(S_{334}, \rightarrow)$	12

State
(timestep=335)

Q_θ

Action-State	Value
$(S_{335}, \leftarrow)$	3
$(S_{335}, \downarrow)$	0
$(S_{335}, \uparrow)$	0
$(S_{335}, \rightarrow)$	15

**Game: Atari Super Mario Bros

Deep Reinforcement Learning

Policy-based Learning

❖ 정책기반 강화학습

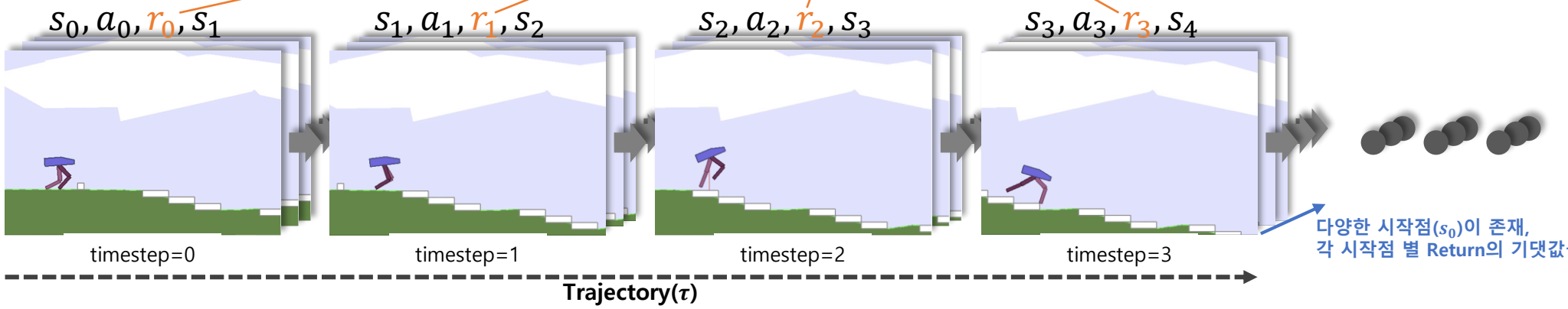
- 주어진 상태에 대하여 정책함수(π_θ)가 행동을 직접 선택하는 방법론 (가치함수를 쓰지 않음)
- 정책함수는 확률분포에 기반하여 행동을 선택(stochastic policy)하므로 행동 공간이 연속적인 경우에도 사용 가능함
- 명시된 정책함수가 주어지며 누적 보상이 최대화 되는 방향으로 학습 (gradient ascent : $\theta' \leftarrow \theta + \alpha \nabla_\theta \big(J(\theta)\big)$)

Return의 기댓값

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[R(\tau)] = \sum_{s \in S} d(s) * v_{\pi_\theta}(s)$$

시작점의 상태 가치(=Return)

시작점이 특정 상태(s)일 확률분포



**Game: Bipedal Walker

Policy Gradient

How to update parameters in policy function?

정책 기반 학습의 목적 함수

$$J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)] = \sum_{s\in S} d(s) * v_{\pi_\theta}(s) = \sum_{s\in S} d(s) * \sum_{a\in A} Q_\pi(s,a) * \pi_\theta(s,a)$$

❖ Policy Gradient Theorem

- 정책 기반 학습의 목적함수는 평균 Return을 최대화하는 것이므로 gradient ascent로 신경망의 파라미터를 업데이트함
- 기울기(gradient)를 구하기 위해서는 환경에 대한 정보가 없는 것을 고려하여 제시된 식에서 변형이 필요함
  - ✓ Return 계산 시 필요한 상태 별 행동에 따른 보상 정보($R_{s,a}$)와 상태-행동 전이확률($P^a_{ss'}$)을 알 수 없음
  - ✓ 해당 식은 모든 상태 별 행동에 따른 보상의 합을 필요로 함
- 변형된 식은 샘플 기반의 방법론으로 충분한 수의 샘플로 구한 평균 값을 사용하여 목적함수의 기울기를 계산할 수 있음

$$\nabla_\theta J(\theta) = \nabla_\theta \sum_{s\in S} d(s) * v_{\pi_\theta}(s)$$
$$= \sum_{s\in S} d(s) * \nabla_\theta v_{\pi_\theta}(s)$$
$$= \sum_{s\in S} d(s) * \nabla_\theta \left(\sum_{a\in A} Q_\pi(s,a) * \pi_\theta(s,a)\right)$$
$$= \sum_{s\in S} d(s) * \sum_{a\in A} Q_\pi(s,a) * \nabla_\theta \pi_\theta(s,a)$$
$$= \sum_{s\in S} d(s) * \sum_{a\in A} Q_\pi(s,a) * \pi_\theta(s,a) * \frac{\nabla_\theta \pi_\theta(s,a)}{\pi_\theta(s,a)}$$
$$= \sum_{s\in S} d(s) * \sum_{a\in A} \pi_\theta(s,a) * \nabla_\theta \log \pi_\theta(s,a) * Q_\pi(s,a)$$
$$= \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) * Q_\pi(s,a)]$$

← 특정 상태에서 시작할 확률 $d(s)$와 특정 상태에서 특정 행동을 고를 확률 $\pi_\theta(s,a)$를 고려한 $\nabla_\theta \log \pi_\theta(s,a) * Q_\pi(s,a)$의 가중합

Policy Gradient

How to update parameters in policy function?

정책 기반 학습의 목적 함수

$$J(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[R(\tau)] = \sum_{s\in S} d(s) * v_{\pi_\theta}(s) = \sum_{s\in S} d(s) * \sum_{a\in A} Q_\pi(s,a) * \pi_\theta(s,a)$$

❖ Policy Gradient Theorem (Proof)

$$\nabla_\theta J(\theta) = \nabla_\theta \sum_{s\in S} d(s) * V_{\pi_\theta}(s)$$

$$= \sum_{s\in S} d(s) * \nabla_\theta V_{\pi_\theta}(s)$$

$$= \sum_{s\in S} d(s) * \nabla_\theta \left(\sum_{a\in A} Q_\pi(s,a) * \pi_\theta(s,a)\right)$$

→ $Q_\pi(s,a)$는 미분이 어렵기 때문에 이를 대체할 수 있는 식으로 변형하는 것이 목표

$$= \sum_{s\in S} d(s) * \sum_{a\in A} \left(\nabla_\theta Q_\pi(s,a) * \pi_\theta(s,a) + Q_\pi(s,a) * \nabla_\theta \pi_\theta(s,a)\right)$$

← 미분의 곱셈 법칙에 의해 다음과 같이 표현

$$= \sum_{s\in S} d(s) * \sum_{a\in A} \left(\nabla_\theta \sum_{s',r} P(s',r|s,a)\left(r + V_{\pi_\theta}(s')\right) * \pi_\theta(s,a) + Q_\pi(s,a) * \nabla_\theta \pi_\theta(s,a)\right)$$

← 미래 상태 가치 함수를 활용하여 표현

$$= \sum_{s\in S} d(s) * \sum_{a\in A} \left(\sum_{s',r} P(s',r|s,a)\nabla_\theta V_{\pi_\theta}(s') * \pi_\theta(s,a) + Q_\pi(s,a) * \nabla_\theta \pi_\theta(s,a)\right)$$

← 상수항 제거

$$= \sum_{s\in S} d(s) * \sum_{a\in A} \left(\sum_{s'} P(s'|s,a)\nabla_\theta V_{\pi_\theta}(s') * \pi_\theta(s,a) + Q_\pi(s,a) * \nabla_\theta \pi_\theta(s,a)\right)$$

← $\sum_{s',r} P(s',r|s,a) = \sum_{s'} P(s'|s,a)$

$$\therefore \nabla_\theta V_{\pi_\theta}(s) = \sum_{a\in A} \left(\sum_{s'} P(s'|s,a)\nabla_\theta V_{\pi_\theta}(s') * \pi_\theta(s,a) + Q_\pi(s,a) * \nabla_\theta \pi_\theta(s,a)\right)$$

해당 값이 재귀적으로 반복됨

Policy Gradient

How to update parameters in policy function?

정책 기반 학습의 목적 함수 변형(일부)

$$\nabla_\theta V_{\pi_\theta}(s) = \sum_{a\in A}\left(\sum_{s'} P(s'|s,a)\nabla_\theta V_{\pi_\theta}(s') * \pi_\theta(s,a) + Q_\pi(s,a) * \nabla_\theta \pi_\theta(s,a)\right)$$

❖ Policy Gradient Theorem (Proof)

- 변형한 식에 전이확률(transition probability, ρ)을 활용하여 전개를 진행
  - ✓ 특정 상태에서 또 다른 특정 상태로 이동할 확률을 전이확률이라 함,
  - ✓ ex) $\rho_\pi(s \to s', k)$, 정책 π하에서 상태s에서 상태s'로 k 스텝 후에 도착할 확률
- 아래 증명에서 식을 간단히 하기 위하여 변형식(일부)의 $\sum_{a\in A} Q_\pi(s,a) * \nabla_\theta \pi_\theta(s,a)$ 부분은 $\phi(s)$으로 축약함

$$\nabla_\theta V_{\pi_\theta}(s)$$

$$= \phi(s) + \sum_a \pi_\theta(a|s) \sum_{s'} P(s'|s,a)\nabla_\theta V_{\pi_\theta}(s')$$

$$= \phi(s) + \sum_{s'}\sum_a \pi_\theta(a|s)P(s'|s,a)\,\nabla_\theta V_{\pi_\theta}(s')$$

현재 상태(s)에서 다음 상태(s')까지 1 timestep으로 전이할 확률으로 표현

$$= \phi(s) + \sum_{s'} \rho_\pi(s \to s', 1)\,\nabla_\theta V_{\pi_\theta}(s')$$

$$= \phi(s) + \sum_{s'} \rho_\pi(s \to s', 1)\left[\phi(s') + \sum_{s''} \rho_\pi(s' \to s'', 1)\nabla_\theta V_{\pi_\theta}(s'')\right]$$

현재 상태(s)에서 다음 상태(s')를 거쳐 그 다음 상태(s'')까지 2 timestep으로 전이할 확률으로 표현

$$= \phi(s) + \sum_{s'} \rho_\pi(s \to s', 1)\phi(s') + \sum_{s''} \rho_\pi(s \to s'', 2)\nabla_\theta V_{\pi_\theta}(s'')$$

$$= \phi(s) + \sum_{s'} \rho_\pi(s \to s', 1)\phi(s') + \sum_{s''} \rho_\pi(s \to s'', 2)\phi(s'') + \sum_{s'''} \rho_\pi(s \to s''', 3)\nabla_\theta V_{\pi_\theta}(s''')$$

전개

... 이하 반복 ...

$$= \sum_{x\in S}\sum_{k=0}^{\infty} \rho_\pi(s \to x, k)\phi(x)$$

$\nabla_\theta Q_\pi(s,a)$를 제거함으로써 목표 달성 (Slide 13 참고)

Policy Gradient

How to update parameters in policy function?

정책 기반 학습의 목적 함수 변형(일부)

$$\nabla_\theta V_{\pi_\theta}(s) = \sum_{a\in A}\left(\sum_{s'} P(s'|s,a)\nabla_\theta V_{\pi_\theta}(s') * \pi_\theta(s,a) + Q_\pi(s,a) * \nabla_\theta \pi_\theta(s,a)\right)$$

❖ Policy Gradient Theorem (Proof)

- 앞서 변형한 재귀식을 본래 목적함수 미분 식에 적용

$$\nabla_\theta J(\theta) = \nabla_\theta V_{\pi_\theta}(s_0)$$

← 처음 시작하는 상태를 s_0이라 정의함

$$= \sum_s \sum_{k=0}^{\infty} \rho_\pi(s_0 \to s, k)\,\phi(s)$$

$\eta(s) = \sum_{k=0}^{\infty} \rho_\pi(s_0 \to s, k)$이라 축약
(Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press., p.199)

$$= \sum_s \eta(s)\,\phi(s)$$

$$= \left(\sum_{s'} \eta(s')\right) \sum_s \frac{\eta(s)}{\sum_{s'} \eta(s')}\phi(s)$$

← $\eta(s)$을 일반화하여 확률의 형태로 만들어줌
(Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press., p.199)

$$= \left(\sum_{s'} \eta(s')\right) \sum_s d^\pi(s)\,\phi(s)$$

← $\sum_s d^\pi(s) = \frac{\eta(s)}{\sum_{s'} \eta(s')}$는 정책함수 하의 상태분포이며 stationary distribution임

$$\propto \sum_s d^\pi(s) \sum_a \nabla_\theta \pi_\theta(a|s) Q_\pi(s,a)$$

← $\sum_{s'} \eta(s')$는 종료 시점이 없는 경우 에피소드의 평균 길이가 되며 1이 됨
(Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press., p.326)

$$= \sum_s d^\pi(s) \sum_a \pi_\theta(a|s) Q_\pi(s,a) \frac{\nabla_\theta \pi_\theta(a|s)}{\pi_\theta(a|s)}$$

샘플링 기반의 방법론으로 변경됨

$$= \mathbb{E}_\pi[\nabla_\theta \log \pi_\theta(a|s) * Q_\pi(s,a)]$$

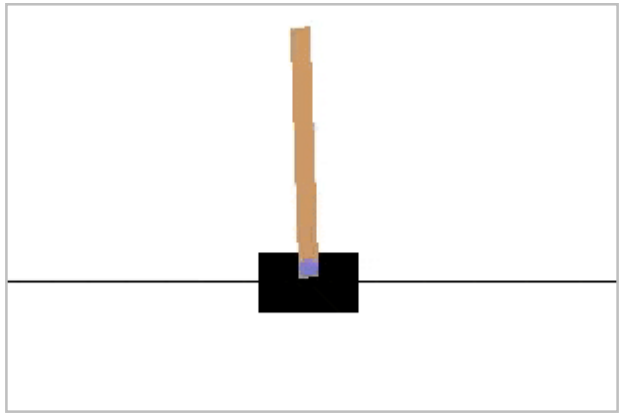
** 상태전이를 충분히 진행하여 전이확률이 더 이상 변하지 않을 때 stationary distribution이라 함

Policy Gradient

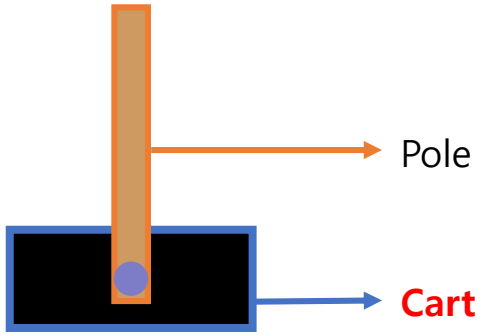
Softmax Policy & Gaussian Policy

❖ Softmax Policy (discrete action)

- 이산적인 값을 행동으로 취할 때 사용하는 정책
- 주어진 상태에 따른 행동 별 로짓(logit) 값에 소프트맥스(softmax)를 취한 뒤 해당 확률에 기반하여 선택할 행동을 추출함
- 행동의 개수가 2개인 경우 시그모이드 함수(sigmoid function)를 쓰기도 하며 3개 이상인 경우 소프트맥스 함수를 씀



< Cartpole >



< Agent >

Action	Category(index)
Cart	Left(0), Right(1)

< Action Space >

Policy Gradient

Softmax Policy & Gaussian Policy

❖ Softmax Policy (discrete action)

$$\pi(a|s,\theta) \doteq \frac{e^{h(a|s,\theta)}}{\sum_b e^{h(b|s,\theta)}}$$

Action	Category(index)
Cart	Left(0), Right(1)

각 행동에 대한 로짓값

$h_\theta(a_0|s)$ $h_\theta(a_1|s)$

로짓값에 대한 소프트맥스

Softmax

$$\frac{e^{h_\theta(a_0|s)}}{e^{h_\theta(a_0|s)} + e^{h_\theta(a_1|s)}} \qquad \frac{e^{h_\theta(a_1|s)}}{e^{h_\theta(a_0|s)} + e^{h_\theta(a_1|s)}}$$

Sampling

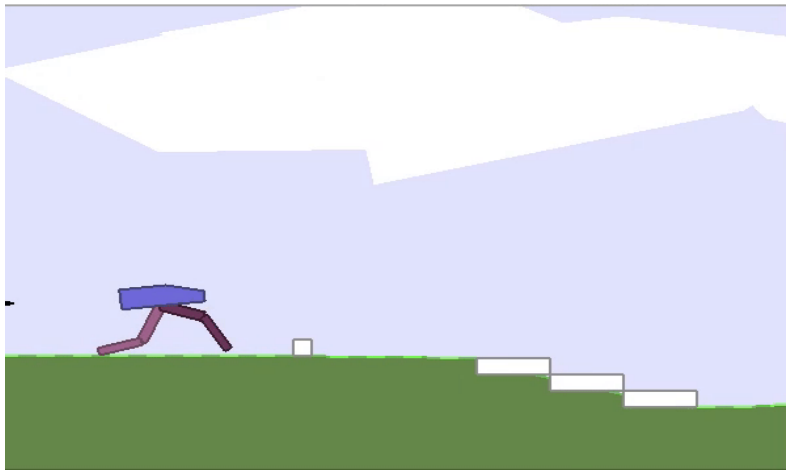
Left or Right

Policy Gradient

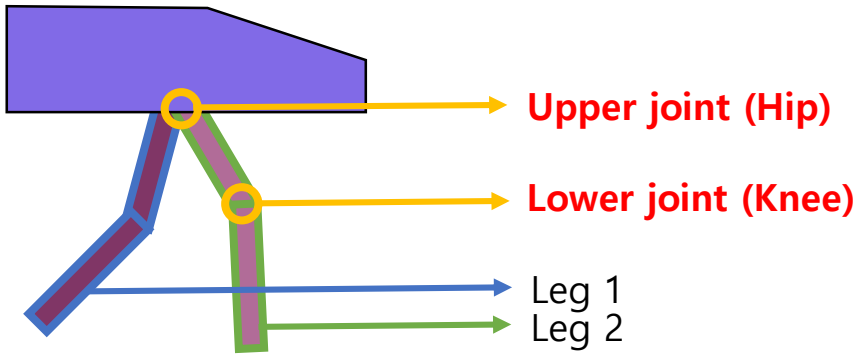
Softmax Policy & Gaussian Policy

❖ Gaussian Policy (continuous action)

- 연속적인 값을 행동으로 취할 때 사용하는 정책
- 일반적으로 주어진 상태에 적합한 정규분포(Gaussian distribution)를 추론하여 이에 추출된 값을 행동으로 사용함
  - ✓ 평균(μ), 분산(σ) 각각을 추론하는 신경망($\mu_\theta, \sigma_\theta$)이 존재
  - ✓ 분산의 경우 상수로 고정하거나 주어지는 상태 값에 독립적으로 동작하도록 학습하기도 함
- 분산을 추론할 때 값이 음수가 되지 않도록 일반적으로 지수함수(exponential function)를 취함



< Bipedal Walker >



< Agent >

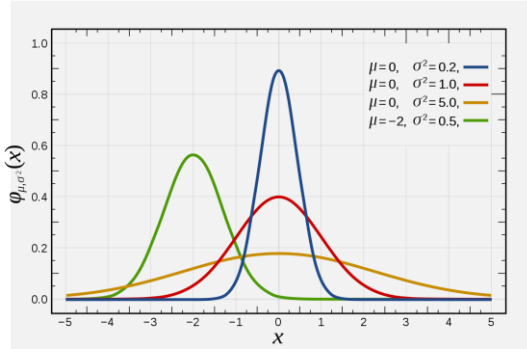
Action	Range
Hip 1 (Torque/Velocity)	-1 ~ 1
Hip 2 (Torque/Velocity)	-1 ~ 1
Knee 1 (Torque/Velocity)	-1 ~ 1
Knee 2 (Torque/Velocity)	-1 ~ 1

< Action Space >

Policy Gradient

Softmax Policy & Gaussian Policy

❖ **Gaussian Policy (continuous action)**



Gaussian Distribution

$$\pi(a|s,\theta) \doteq \frac{1}{\sigma(s,\theta)\sqrt{2\pi}} \exp\left(-\frac{\left(a-\mu(s,\theta)\right)^2}{2\sigma(s,\theta)^2}\right)$$

독립적인 신경망으로 동작함

Hip 1 (Torque/Velocity)	Knee 1 (Torque/Velocity)	Hip 2 (Torque/Velocity)	Knee 2 (Torque/Velocity)

$\mu_\theta^{a_1}$ $\sigma_\theta^{a_1}$ $\mu_\theta^{a_2}$ $\sigma_\theta^{a_2}$ $\mu_\theta^{a_3}$ $\sigma_\theta^{a_3}$ $\mu_\theta^{a_4}$ $\sigma_\theta^{a_4}$

$\mathcal{N}\left(\mu_\theta^{a_1},\left(\sigma_\theta^{a_1}\right)^2\right)$ $\mathcal{N}\left(\mu_\theta^{a_2},\left(\sigma_\theta^{a_2}\right)^2\right)$ $\mathcal{N}\left(\mu_\theta^{a_3},\left(\sigma_\theta^{a_3}\right)^2\right)$ $\mathcal{N}\left(\mu_\theta^{a_4},\left(\sigma_\theta^{a_4}\right)^2\right)$

a_1 a_2 a_3 a_4

추론된 평균 및 분산의
정규분포에서 행동 값을 추출함

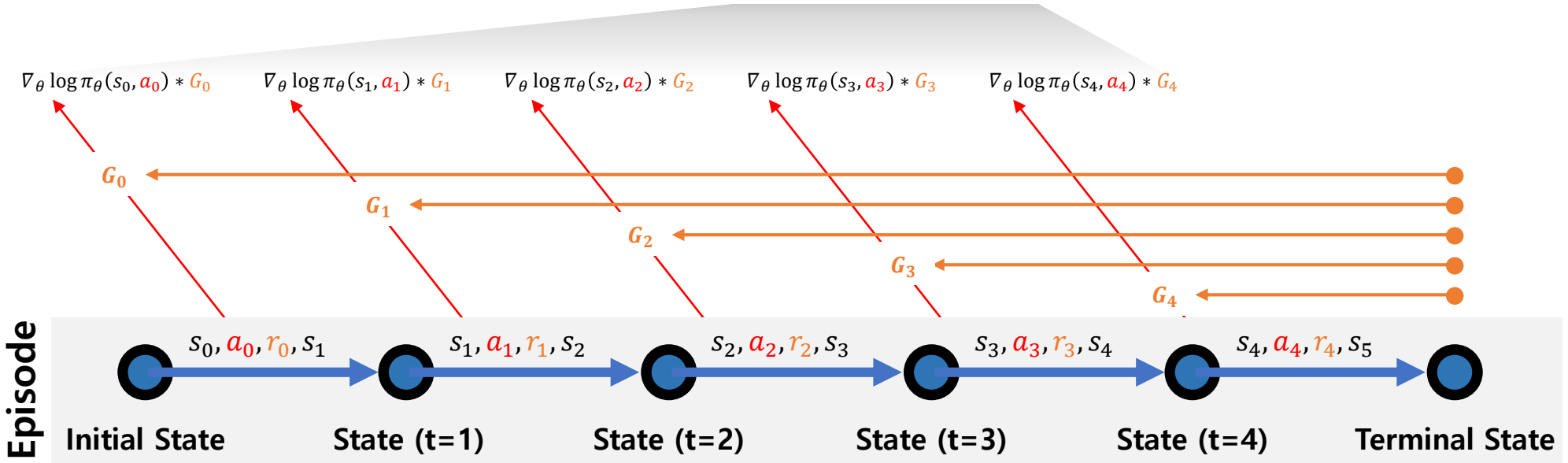
Policy Gradient

Policy-based Reinforcement Learning with MC method

❖ REINFORCE

- 에피소드 단위의 경험을 사용하여 정책을 학습 (Return(G)을 사용함)
  - ✓ $Q_\pi(s,a) = \mathbb{E}[G_t | s_t = s, a_t = a]$
  - ✓ 에피소드 단위의 경험을 사용하므로 반드시 종료 상태(terminal state)가 있어야함
- Monte-carlo 방식을 사용하기 때문에 샘플의 평균은 불편추정량이지만 높은 분산을 가짐
  - ✓ 불편추정량임을 유지하고 동시에 높은 분산을 낮추기 위해서 리턴에서 baseline을 빼는 방식을 사용함 ($\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) * (G - b)]$)

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) * G\ ]$$

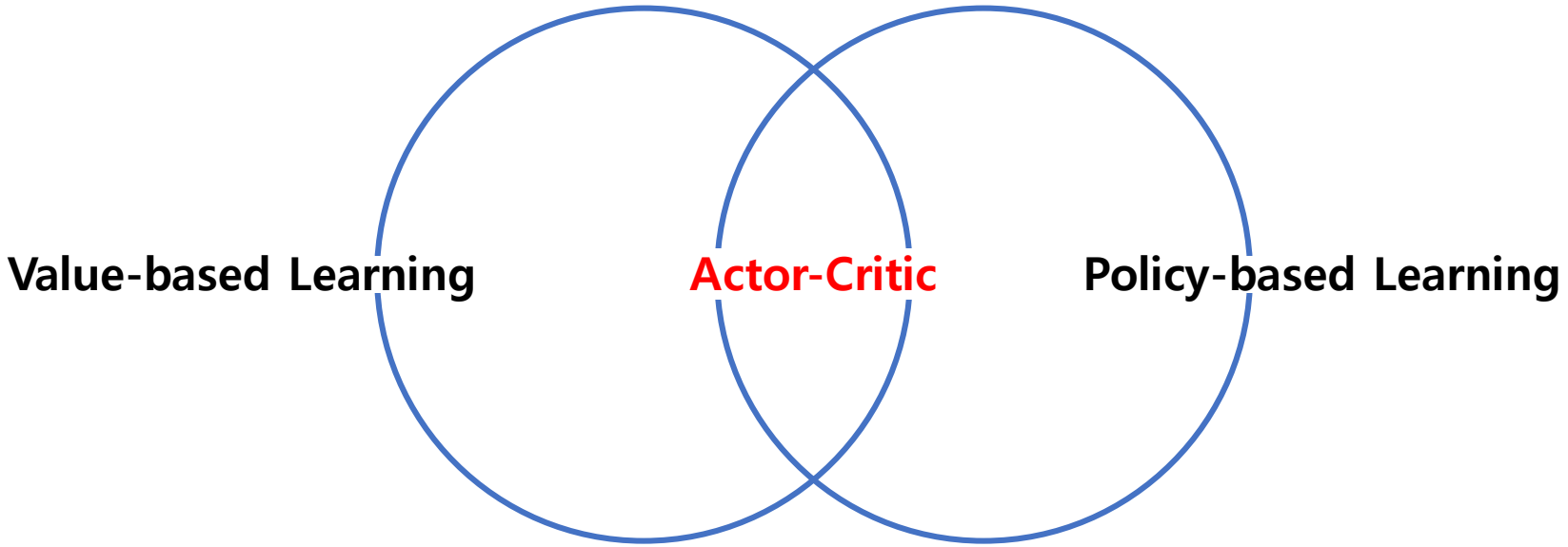


Actor-Critic

Actor-Critic

❖ 액터-크리틱이란?

- 정책함수(액터)와 가치함수(크리틱)를 함께 학습하는 방법론
- 액터는 주어진 상태에서 행동을 선택하며, 크리틱은 선택한 행동의 가치를 평가함
- Monte-Carlo를 사용하는 REINFORCE와 달리 Temporal-Difference를 사용할 수 있음 (non-terminal 상황에서도 적용 가능)



Actor-Critic

Actor-Critic using State-Action Value

❖ Q Actor-Critic

- REINFORCE에서 사용한 실제 리턴(G) 대신 정책함수에서 선택한 행동의 가치($Q_{\pi_\theta}(s,a)$)를 추정하는 가치함수의 값($Q_w(s,a)$)을 사용함
- 정책함수(π_θ)와 가치함수(Q_w)는 서로 다른 파라미터(θ, w)로 구성된 신경망이며 각각 학습이 필요함
- 가치함수의 평가에 따라서 정책함수가 상태에 따라 행동을 선택하는 학습의 방향이 결정됨
- Q Actor-Critic은 Monte-Carlo, Temporal-Difference 업데이트 방식 모두 사용 가능함

액터 업데이트 수식 $\theta \leftarrow \theta + \alpha_1 \nabla_\theta \log \pi_\theta(s,a) * Q_w(s,a)$

크리틱 업데이트 수식 $w \leftarrow w + \alpha_2 \big(\underbrace{r + \gamma Q_w(s',a')}_{\text{TD Target}} - Q_w(s,a)\big) \nabla_w Q_w(s,a)$

TD Error

Actor-Critic

Actor-Critic using baseline

❖ Advantage Actor-Critic

- 추정한 행동 가치(Q_w)에서 추정한 상태 가치(V_ϕ)를 제거함으로써 상대적인 가치를 판단
- 추정한 상태 가치라는 baseline을 도입함으로써 추정한 기울기의 분산이 감소하는 효과
- Baseline 자체는 정책 함수의 업데이트에 영향을 미치지 않음
  - ✓ 상태 별 추정된 행동 가치 간의 변동을 줄여주는 역할을 하며 전체 샘플의 평균 기울기 관점에서는 값이 0이 됨

액터 업데이트 수식

$$\theta \leftarrow \theta + \alpha_1 \nabla_\theta \log \pi_\theta(s,a) * A(s,a)$$

Advantage Function

$$A(s,a) = Q_w(s,a) - V_\phi(s)$$

Baseline

크리틱 업데이트 수식

$$w \leftarrow w + \alpha_2 \left(r + \gamma Q_w(s',a') - Q_w(s,a)\right) \nabla_w Q_w(s,a)$$

$$\phi \leftarrow \phi + \alpha_3 \left(r + \gamma V_\phi(s') - V_\phi(s)\right) \nabla_\phi V_\phi(s)$$

Actor-Critic

Actor-Critic using baseline

❖ Baseline does not introduce the bias

$$\mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a) * \{Q_{\pi_\theta}(s,a) - B(s)\}\right]$$
$$= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a) * Q_{\pi_\theta}(s,a)\right] - \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) * B(s)]$$

$$\mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta(s,a) * B(s)]$$
$$= \sum_{s\in S} d_{\pi_\theta}(s) \sum_{a\in A} \pi_\theta(s,a) \nabla_\theta \log \pi_\theta(s,a) * B(s)$$
$$= \sum_{s\in S} d_{\pi_\theta}(s) \sum_{a\in A} \pi_\theta(s,a) * \frac{\nabla_\theta \pi_\theta(s,a)}{\pi_\theta(s,a)} * B(s)$$
$$= \sum_{s\in S} d_{\pi_\theta}(s) \sum_{a\in A} \nabla_\theta \pi_\theta(s,a) * B(s)$$
$$= \sum_{s\in S} d_{\pi_\theta}(s) * B(s) * \sum_{a\in A} \nabla_\theta \pi_\theta(s,a)$$
$$= \sum_{s\in S} d_{\pi_\theta}(s) * B(s) * \nabla_\theta \sum_{a\in A} \pi_\theta(s,a)$$
$$= \sum_{s\in S} d_{\pi_\theta}(s) * B(s) * \nabla_\theta 1$$
$$= 0$$

따라서 baseline은 기댓값 연산에 영향을 미치지 않음

Actor-Critic

Actor-Critic using Temporal-Difference

❖ TD Actor-Critic

- Advantage Actor-Critic은 파라미터 세 가지(θ, w, ϕ)를 업데이트 해야하므로 학습 비용이 많이 든다는 단점이 있음
- TD Actor-Critic은 TD error(δ)의 기댓값이 Advantage function의 불편추정량이라는 점을 활용함
- TD error는 행동 가치가 필요 없으며 상태 가치만을 사용함
- Advantage function과 동일한 효과를 누리되 보다 적은 파라미터($\theta, w, \phi \rightarrow \theta, \phi$)로 사용할 수 있음

액터 업데이트 수식 $\theta \leftarrow \theta + \alpha_1 \nabla_\theta \log \pi_\theta(s,a) * \delta$

TD Error

$\delta = r + \gamma V(s') - V(s)$

크리틱 업데이트 수식 $\phi \leftarrow \phi + \alpha_2 \delta \nabla_\phi V_\phi(s)$

Summary

Conclusion

❖ Conclusion

- 본 세미나에서는 policy gradient에 대한 기초 및 관련 알고리즘을 정리함
- 정책 기반 강화학습은 기존 가치 기반 강화학습에서 어려웠던 연속적인 행동 공간에서의 학습 그리고 보상을 통한 직접적인 정책 학습을 가능하게 함
- 액터-크리틱은 정책 기반 및 가치 기반 강화학습 모두를 응용한 방법론으로 순수 정책 기반의 강화학습 대비 효율적인 학습이 가능하도록 함

Citation

- Deep Reinforcement Learning

노승은, 『바닥부터 배우는 강화학습』, 영진닷컴(2020)

Sutton, Richard S., and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

https://www.youtube.com/playlist?list=PLqYmG7hTraZBiG_XpjnPrSNw-1XQaM_gB
(Introduction to Reinforcement learning with David Silver, DeepMind)

https://lilianweng.github.io/posts/2018-04-08-policy-gradient/
(Policy Gradient Algorithms)

https://www.youtube.com/watch?v=BvZvx7ENZBw
(Proximal Policy Optimization Implementation: 8 Details for Continuous Actions (3/3))

https://en.wikipedia.org/wiki/Normal_distribution