RLCode와 A3C 쉽고 깊게 이해하기

RLCode 리더 이웅원

파이썬과 케라스로 배우는 강화학습

http://wikibook.co.kr/reinforcement-learning/

아기울음소리 감지기의 예시

인공지능 기반 시계열 데이터 분석

-비음성 소리인식

청각장애인: 아기울음, 경적, 초인종 등

보안/안전: 비명, 충돌음, 폭발음 등

-고장 모니터링

자동차, 산업기계 진동 등

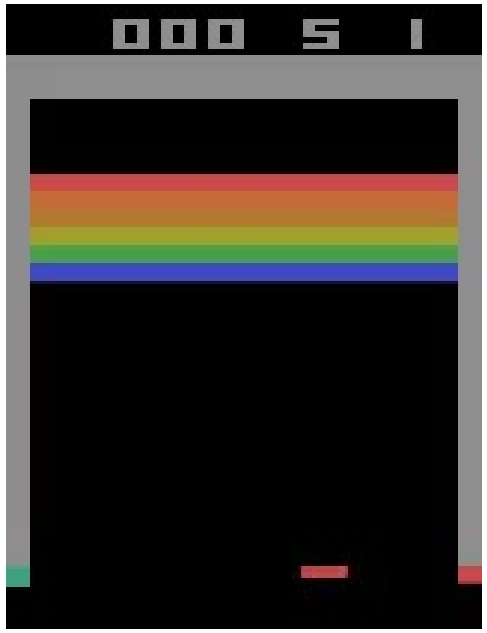
목차

New 베이스라인 A3C

기본적인 강화학습 내용은 안다고 가정

가장 유명한 알고리즘 : DQN

DeepMind → Atari → DQN



게임 화면으로 게임 플레이하는 법을 학습

DQN이 이룬 점과 부족한 점

1. DQN이 이룬 점

1) 게임 화면을 상태 입력으로 받아서 학습(CNN)

2) 사람보다 게임 플레이를 더 잘하는 에이전트

3) 샘플들 사이의 강한 상관관계를 리플레이 메모리로 해결

2. DQN이 부족한 점

1) 많은 메모리의 사용

2) 느린 학습 속도

3) 불안정한 학습 과정

(가치함수에 대한 greedy policy이므로)

A3C = Asynchronous Advantage Actor-Critic

1. 샘플 사이의 상관관계를 비동기 업데이트로 해결
2. 리플레이 메모리를 사용하지 않음
3. policy gradient 알고리즘 사용가능(Actor-Critic)
4. 상대적으로 빠른 학습 속도(여러 에이전트가 환경과 상호작용)

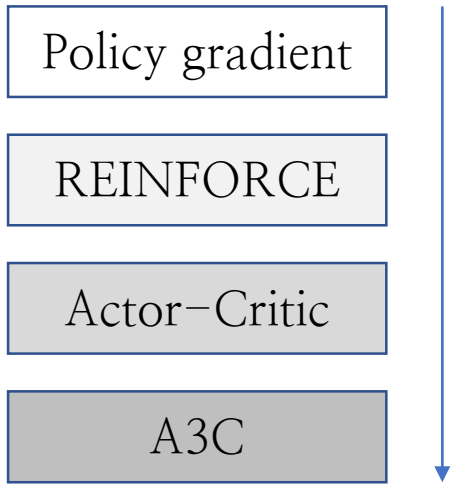
A3C = 비동기 + Actor-Critic

Actor-Critic = REINFORCE + 실시간 학습

REINFORCE = 몬테카를로 Policy gradient

A3C를 파헤치는 과정

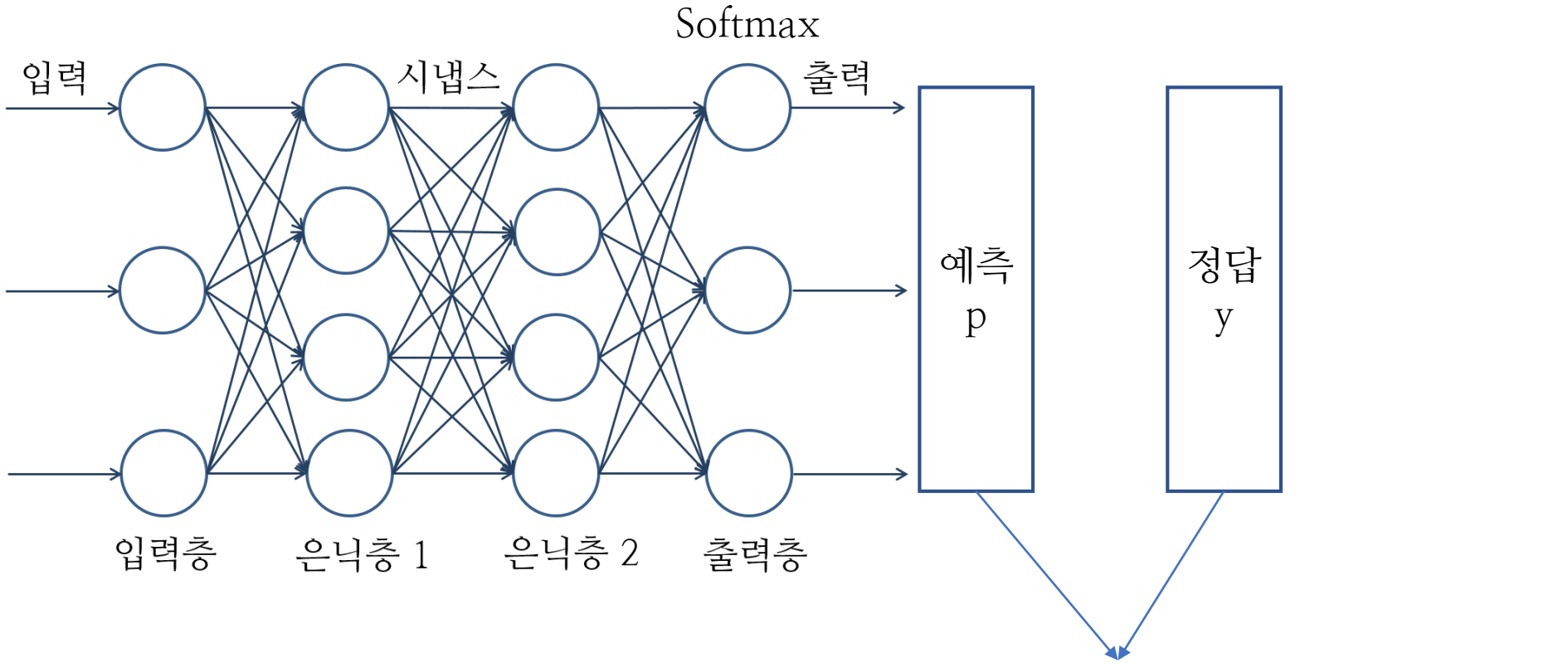
1. Policy gradient의 수식 유도 과정에 대한 이해
2. REINFORCE 알고리즘 이해
3. Actor-Critic 알고리즘 이해
4. Actor-Critic 을 비동기식으로 업데이트하는 방법 이해



A3C의 직관적 이해

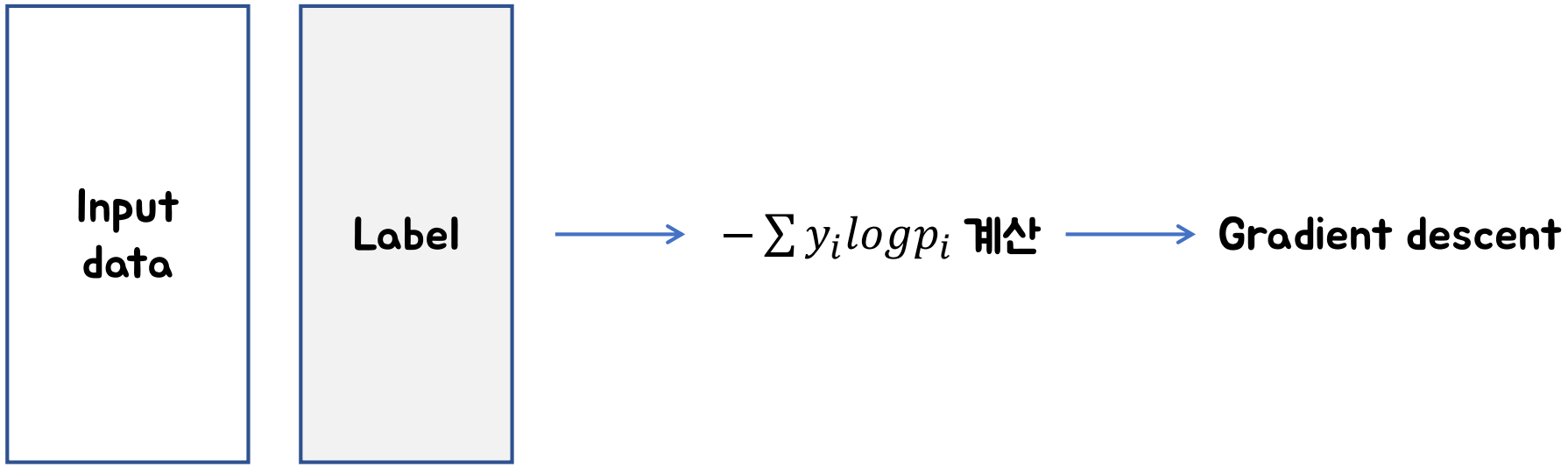
크로스 엔트로피

지도학습의 크로스 엔트로피(Cross entropy)

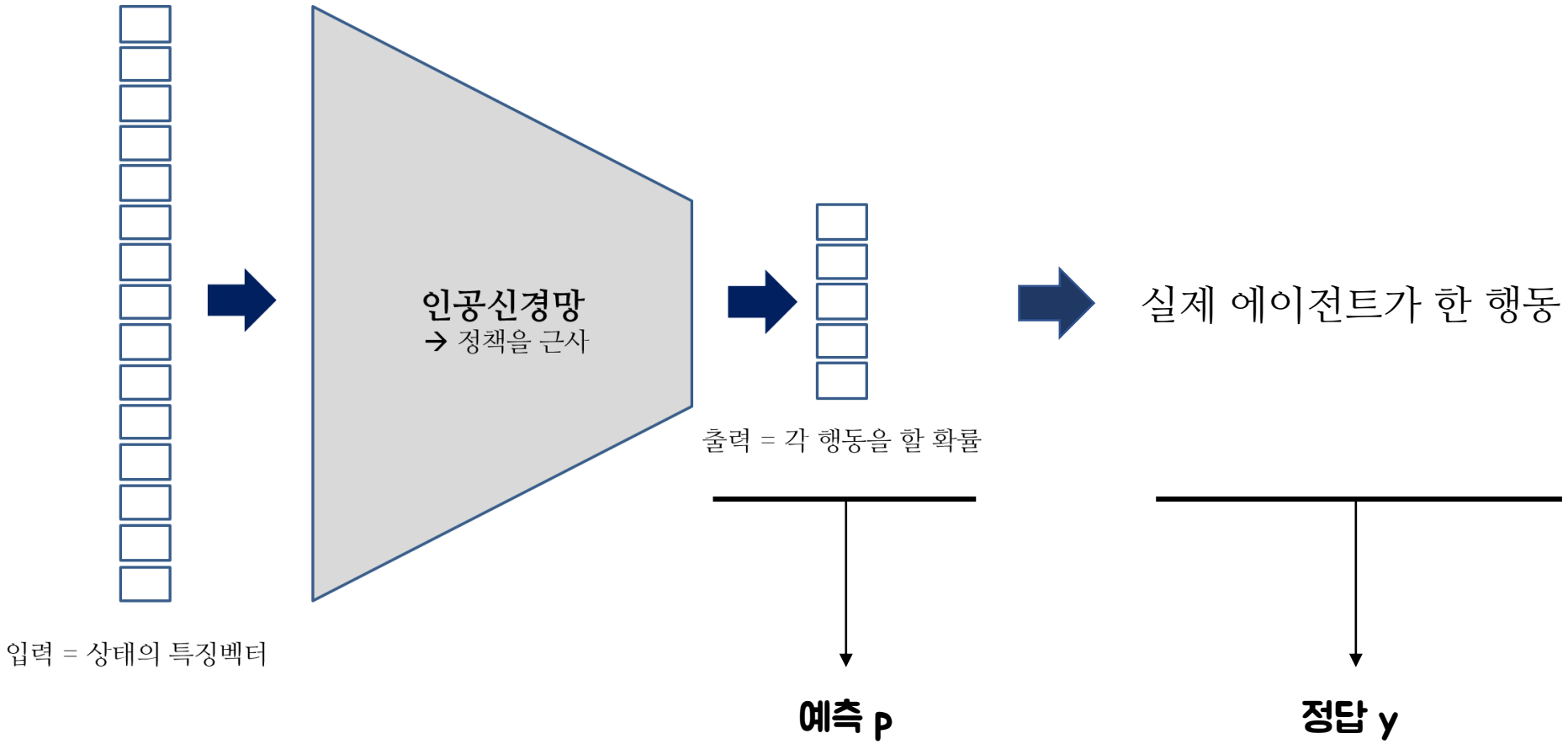


크로스 엔트로피

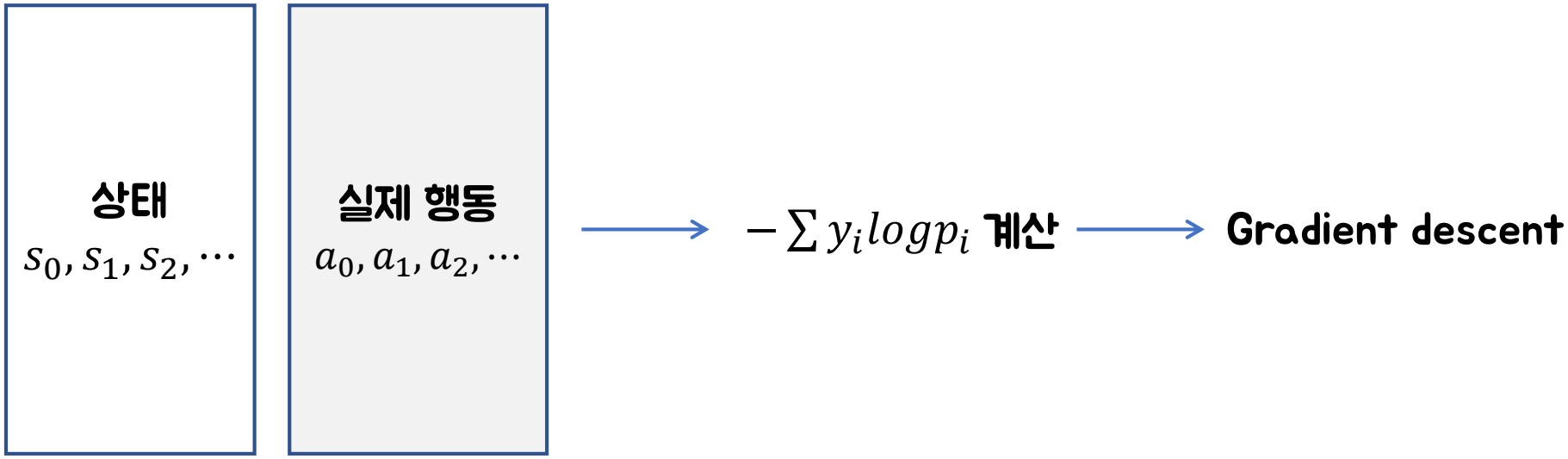
지도학습의 크로스 엔트로피(Cross entropy)



강화학습의 크로스 엔트로피



강화학습의 크로스 엔트로피



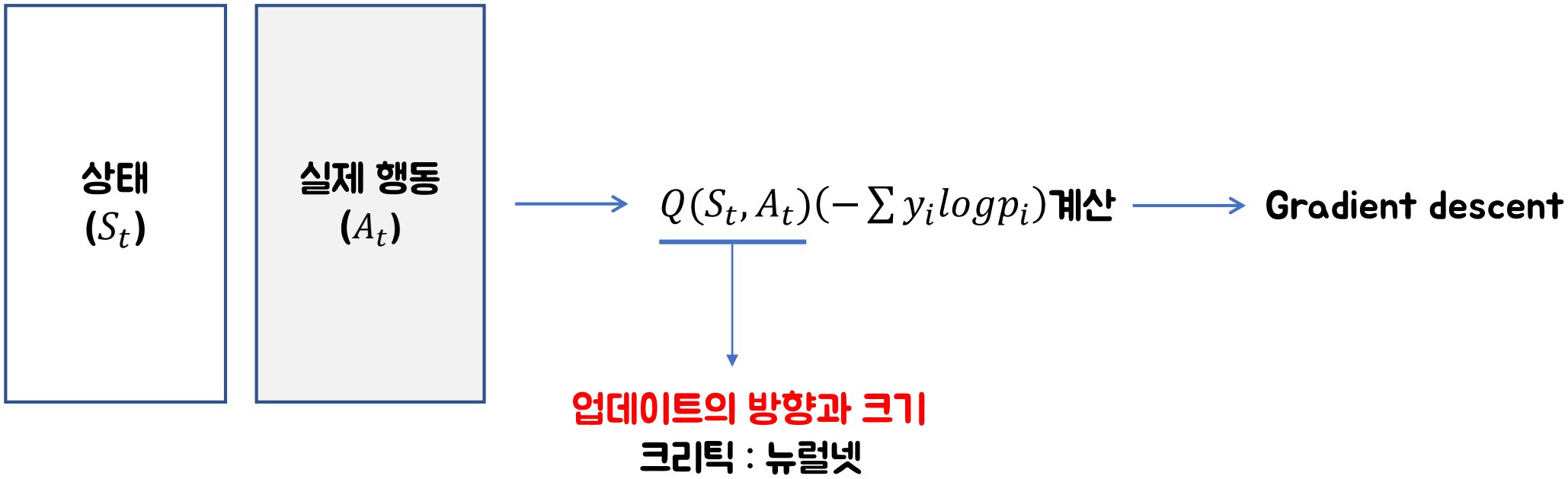
나의 예측을 실제 행동에 가깝게 만든다 → 어떤 의미??

업데이트의 방향성이 필요!

Actor-Critic

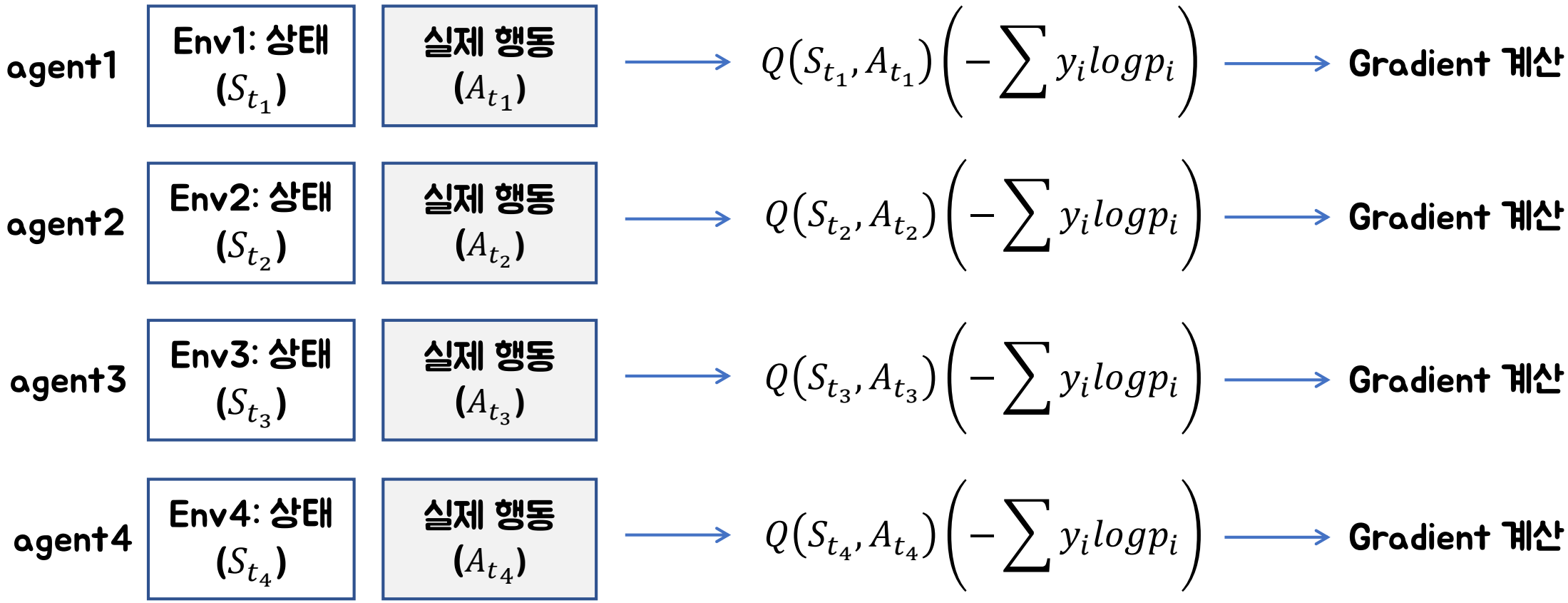
정답이 되는 각 행동이 실제로 좋은 지, 좋다면 얼마나 좋은 지를 알 수 있을까?

→ 큐함수(Q-function) : $Q_\pi(s,a) = \boldsymbol{E}_\pi[R_{t+1} + \gamma R_{t+2} + \cdots | S_t = s, A_t = a]$



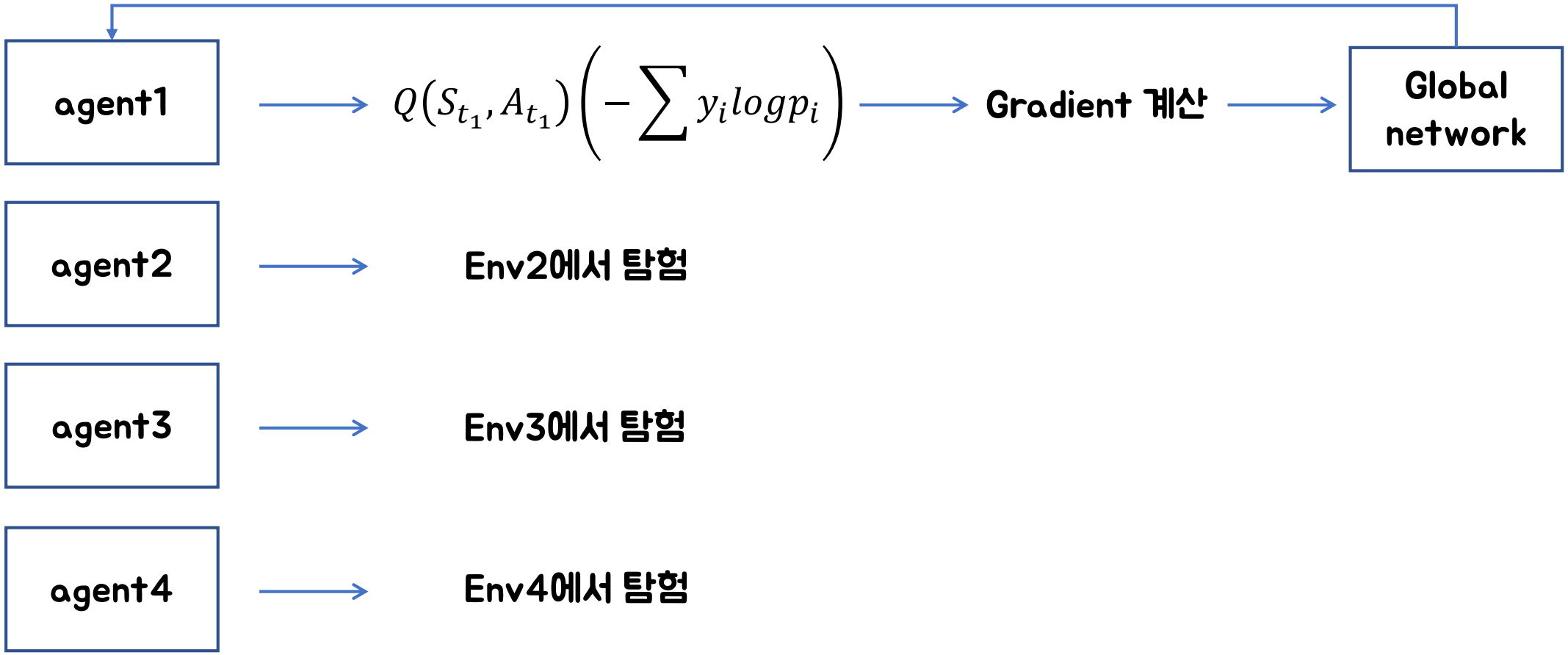
A3C

여러 개의 에이전트를 만들어서 각각 gradient를 계산하면?



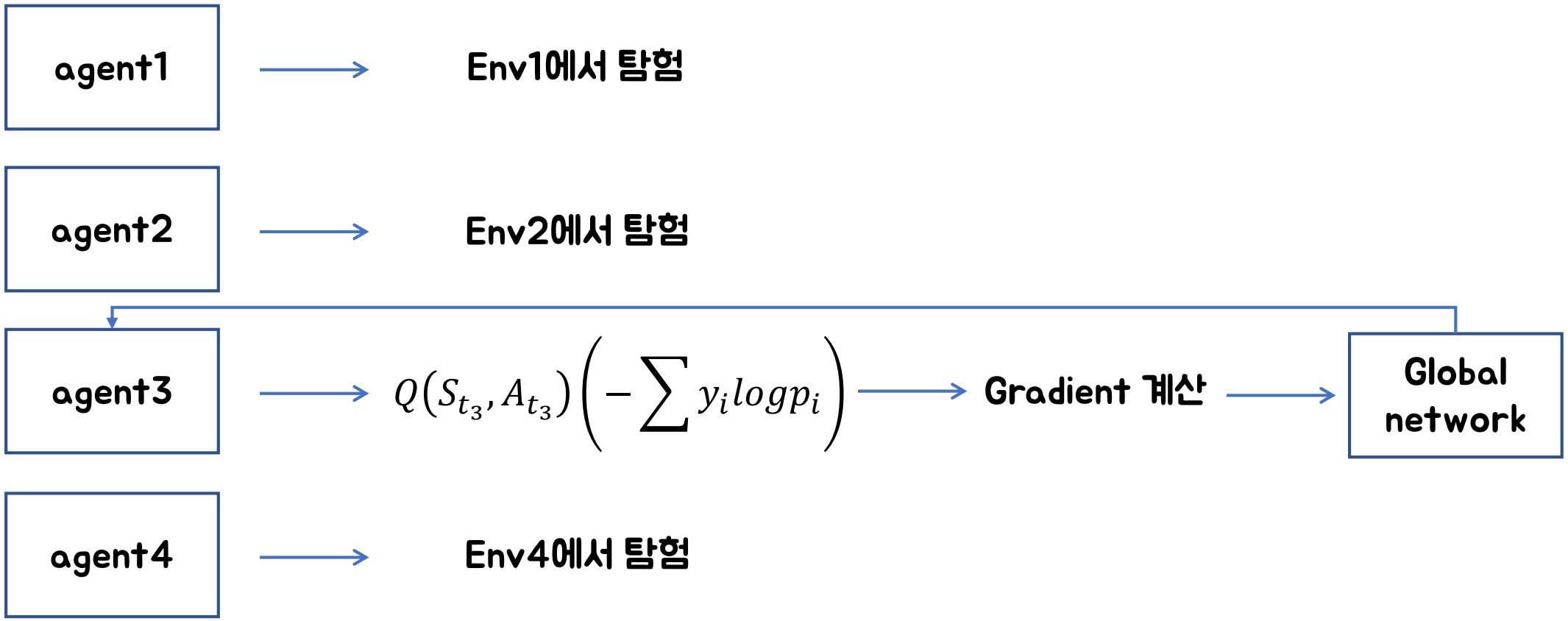
A3C

비동기적으로 global network를 업데이트

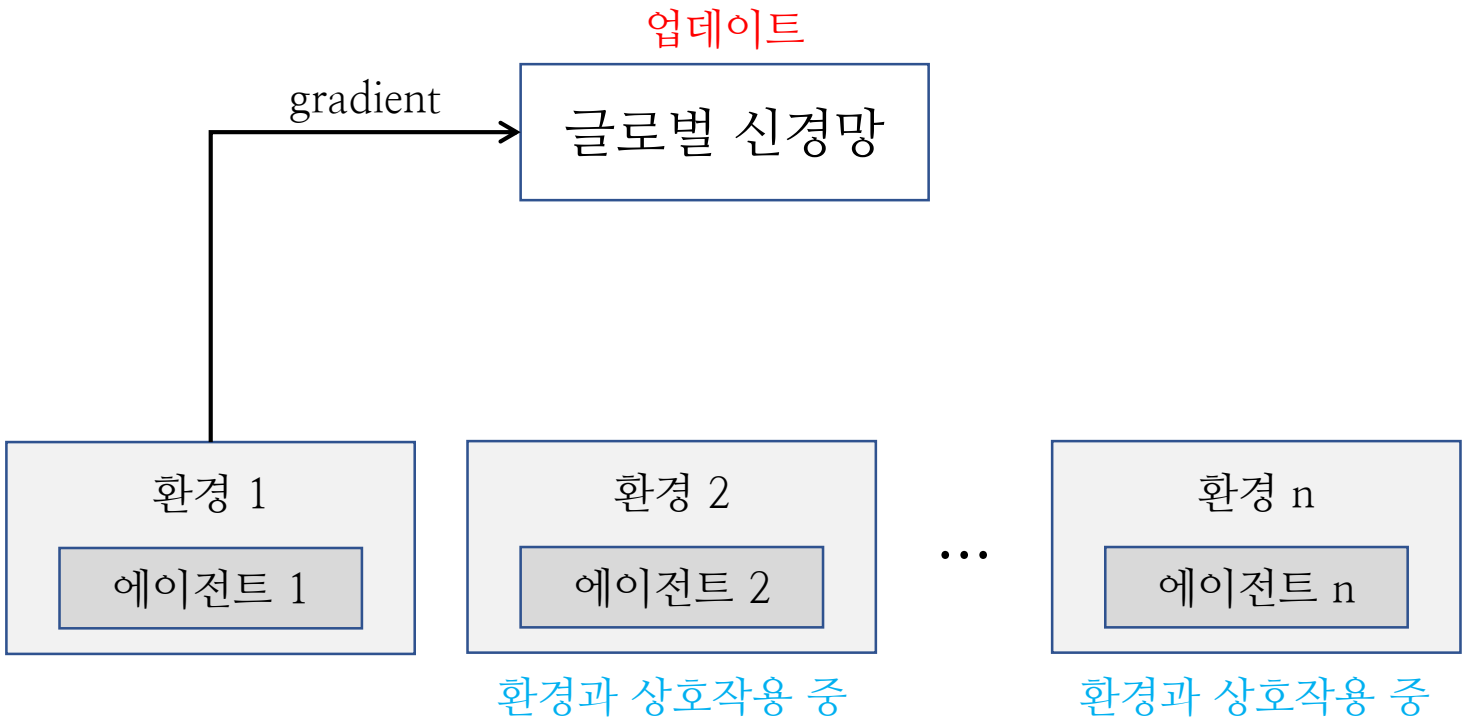


A3C

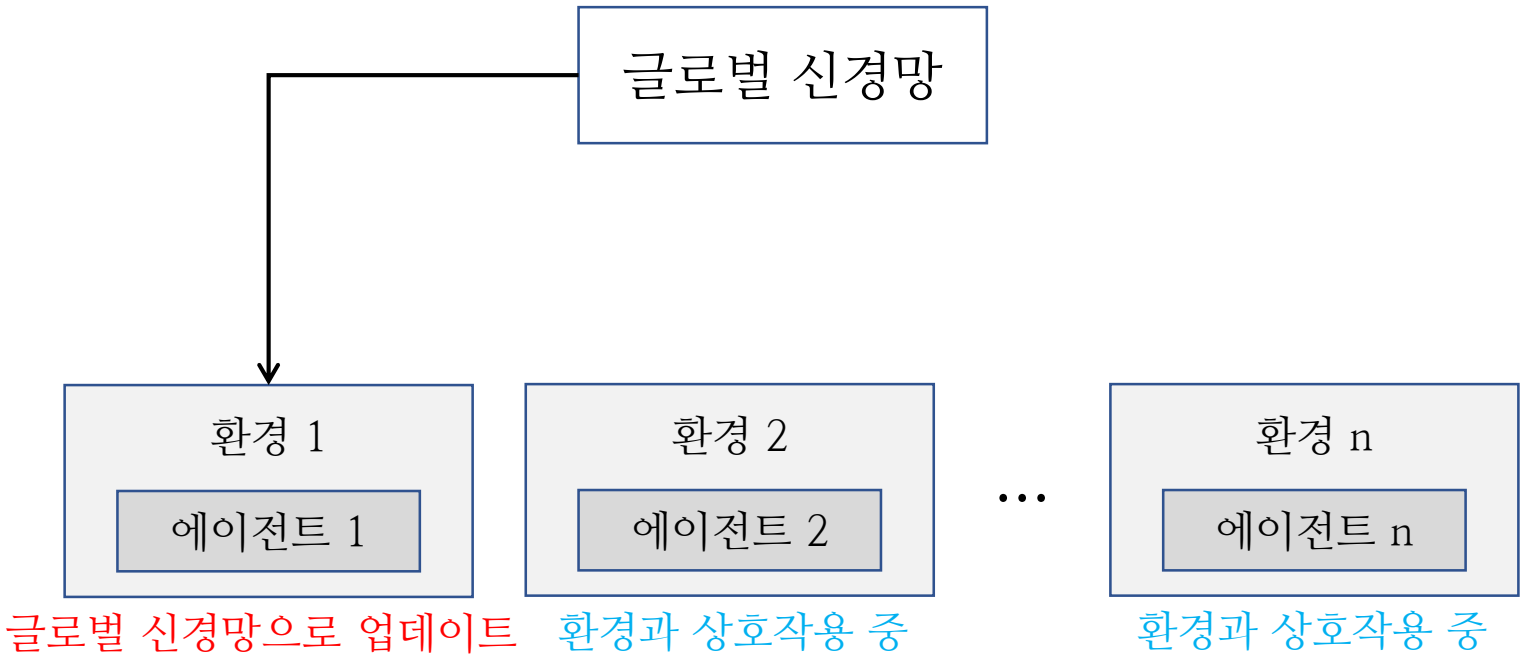
비동기적으로 global network를 업데이트



비동기적으로 global network를 업데이트: 에이전트 1이 글로벌 신경망을 업데이트

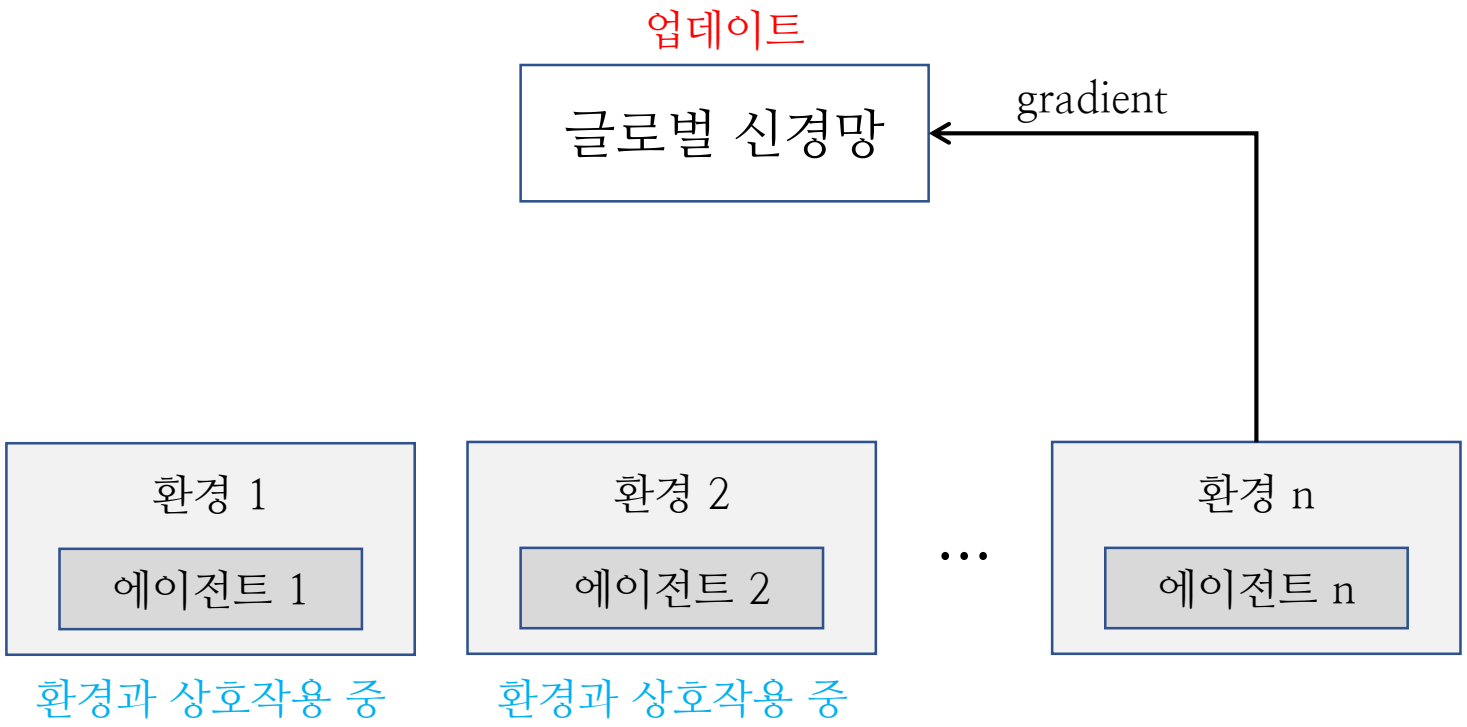


비동기적으로 global network를 업데이트 : 글로벌 신경망으로 에이전트 1을 업데이트



A3C

비동기적으로 global network를 업데이트 : agent n이 글로벌 신경망을 업데이트



이제 수학적으로 접근해봅시다!

Policy Gradient

정책과 강화학습

- 강화학습 → 보상을 받게 하는 행동의 확률 증가
- 즉, 현재 정책을 환경으로부터 받은 보상을 기준으로 업데이트를 해야 "강화학습"
- 2가지를 알아야 함

    1. 정책을 업데이트하는 기준
    2. 그 기준에 따라 정책의 업데이트 방법

Policy gradient

1. 정책을 업데이트 하는 기준: 목표함수

2. 목표함수에 따라 정책을 업데이트하는 방법:
Gradient ascent

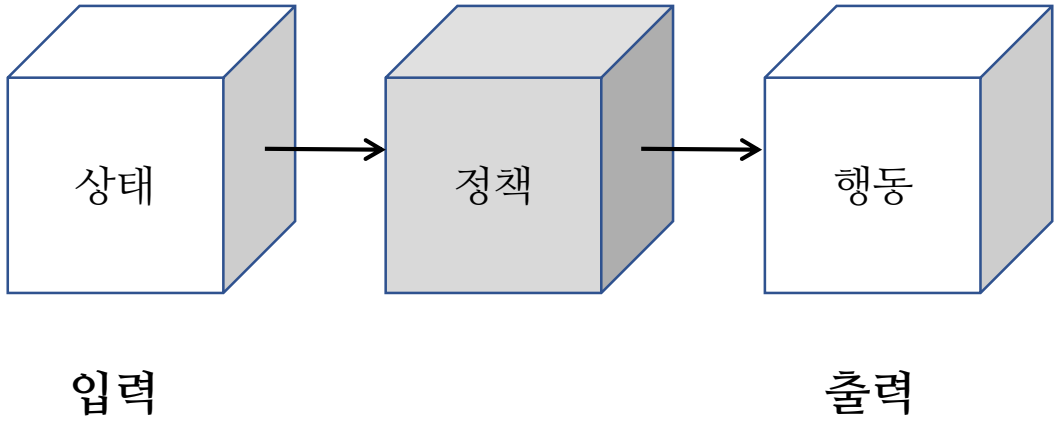
Policy gradient

1. 정책(Policy)의 근사화

2. 목표함수에 대한 Gradient ascent

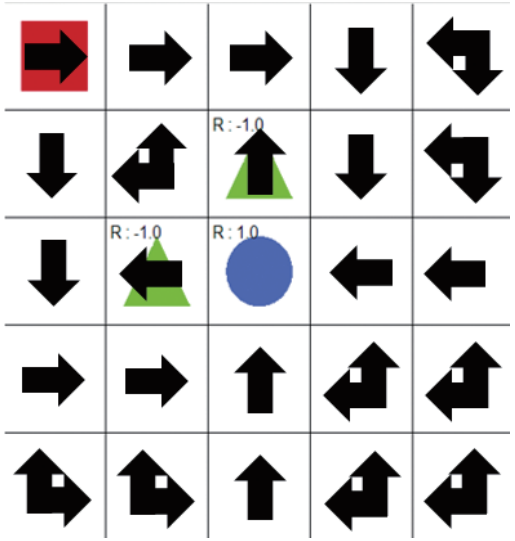
정책의 근사화

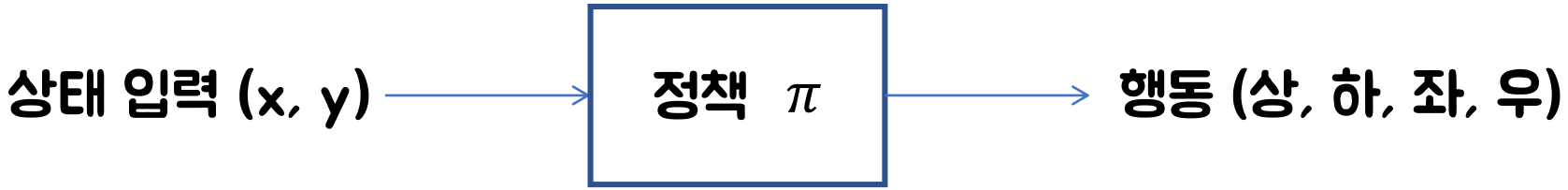
- Huge + High Dimension 상태 공간 → 각 상태에 대한 정책을 가지는 것은 힘들다
- 행동 $= f$(상태) → f가 정책



정책의 근사화

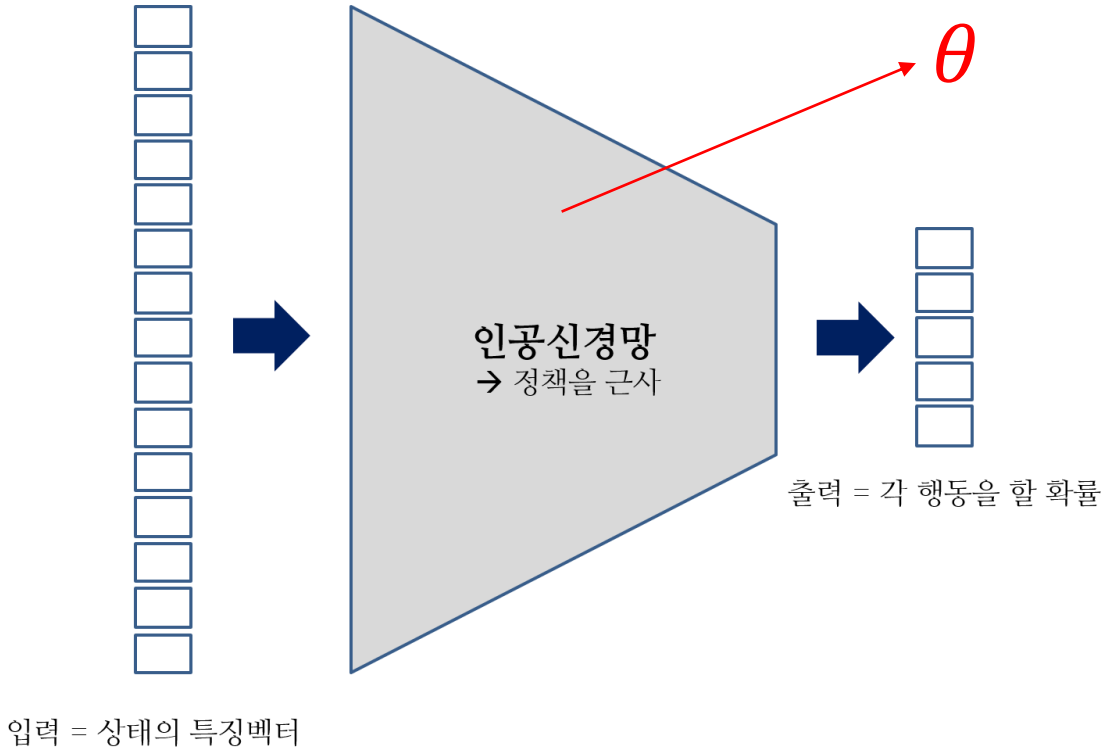
- 그리드월드의 경우
- 상태 : (x, y) 좌표
- 행동 : (상, 하, 좌, 우)





정책의 근사화

- 인공신경망으로 정책을 근사 → 정책 신경망



정책 $\pi_\theta(a|s) = \boldsymbol{P}[A_t = a|S_t = s, \theta]$

Policy gradient

1. 정책(Policy)의 근사화

2. 목표함수에 대한 Gradient ascent

목표함수

- 가치 기반 강화학습

  매 스텝마다 에이전트가 행동을 선택하는 기준 → 가치함수(Value function)

- 정책 기반 강화학습

  정책을 업데이트할 때마다 어떤 방향으로 업데이트할 지에 대한 기준

  → 목표함수 (Objective function) or $J(\theta)$

목표함수

- 에이전트가 정책 π_θ에 따라서 가게 되는 "경로"를 생각해보자!
- 경로(trajectory) = 에이전트와 환경이 상호작용한 흔적

$$\tau = s_0, a_0, r_1, s_1, a_1, r_2, \cdots, s_T$$

- $J(\theta)$ = 경로 동안 받을 것이라고 기대하는 보상의 합(경로가 매번 달라지므로)

$$J(\theta) = \boldsymbol{E}\left[\sum_{t=0}^{T-1} r_{t+1} \mid \pi_\theta\right] = \boldsymbol{E}[r_1 + r_2 + r_3 + \cdots + r_T \mid \pi_\theta]$$

Gradient ascent

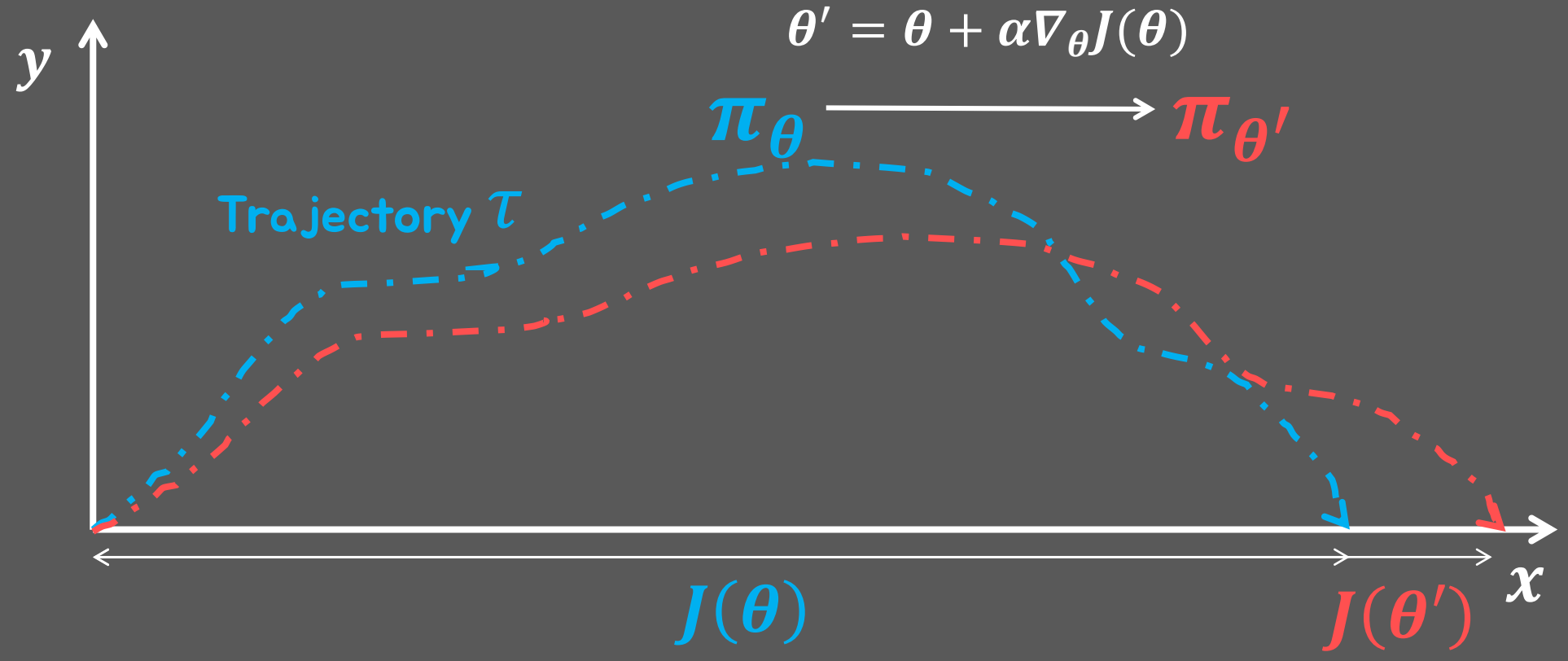
- $J(\theta)$ 를 기준으로 어떻게 θ(정책신경망)을 업데이트할 것인가?

  → θ에 대한 $J(\theta)$의 경사를 따라 올라가다(Gradient ascent)

$$\theta' = \theta + \alpha \nabla_\theta J(\theta)$$

$\nabla_\theta J(\theta)$ = **Policy gradient**

Policy gradient



REINFORCE

수식 주의!!

수식이 많이 나옵니다

Policy gradient 수식 유도

- Policy gradient의 정책 업데이트식 → 핵심은 Policy gradient

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$$

- Policy gradient의 기본 알고리즘 REINFORCE 식 → 이제부터 유도

$$\nabla_\theta J(\theta) \sim \sum_{t=0}^{T-1} \nabla_\theta log \pi_\theta(a_t|s_t)\, G_t$$

Policy gradient 수식 유도

목표함수의 정의

$$J(\theta) = \boldsymbol{E}[\sum_{t=0}^{T-1} r_{t+1} \mid \pi_\theta]$$

기대값 = (x일 확률 × x일 때의 값)의 x에 대한 합

ex) 주사위 기대값

$$\boldsymbol{E}[f(x)] = \sum_{x} p(x) f(x)$$

Policy gradient 수식 유도

$$\tau = s_0, a_0, r_1, s_1, a_1, r_2, \cdots, s_T$$

목표함수의 정의

$$J(\theta) = \boldsymbol{E}\left[\sum_{t=0}^{T-1} r_{t+1} \mid \pi_\theta\right] = \boldsymbol{E}_\tau[r_1 \mid \pi_\theta] + \boldsymbol{E}_{\boldsymbol{\tau}}[r_2 \mid \pi_\theta] + \boldsymbol{E}_{\boldsymbol{\tau}}[r_3 \mid \pi_\theta] + \cdots$$

Trajectory τ에서 r_1에 대한 기대값

$$= \sum_{t=0}^{T-1} \boldsymbol{P}(s_t, a_t|\tau) R(s_t, a_t) = \sum_{t=0}^{T-1} \boldsymbol{P}(s_t, a_t|\tau) r_{t+1}$$

(s_t, a_t)일 확률 (s_t, a_t)일 때의 값

Policy gradient 수식 유도

양변에 ∇_θ 를 취하기(미분하기)

$$\nabla_\theta J(\theta) = \nabla_\theta \, \boldsymbol{E}[\sum_{t=0}^{T-1} r_{t+1} \mid \pi_\theta]$$

$$= \nabla_\theta \sum_{t=0}^{T-1} \boldsymbol{P}(s_t, a_t|\tau) r_{t+1}$$

$$= \sum_{t=0}^{T-1} \nabla_\theta \boldsymbol{P}(s_t, a_t|\tau) r_{t+1}$$

?

$$= \sum_{t=0}^{T-1} \boldsymbol{P}(s_t, a_t|\tau) \nabla_\theta log \boldsymbol{P}(s_t, a_t|\tau) r_{t+1}$$

Policy gradient 수식 유도

$$\nabla_\theta J(\theta) = \sum_{t=0}^{T-1} \nabla_\theta \boldsymbol{P}(s_t, a_t|\tau) r_{t+1}$$

$$= \sum_{t=0}^{T-1} \boldsymbol{P}(s_t, a_t|\tau) \frac{\nabla_\theta \boldsymbol{P}(s_t, a_t|\tau)}{\boldsymbol{P}(s_t, a_t|\tau)} r_{t+1}$$

$$= \sum_{t=0}^{T-1} \boldsymbol{P}(s_t, a_t|\tau) \nabla_\theta log \boldsymbol{P}(s_t, a_t|\tau) r_{t+1}$$

$$\frac{dlogx}{dx} = \frac{1}{x}$$

$$\frac{dlogf(x)}{dx} = \frac{df(x)/dx}{f(x)}$$

Policy gradient 수식 유도

$$\nabla_\theta J(\theta) = \sum_{t=0}^{T-1} \boldsymbol{P}(s_t, a_t|\tau)\nabla_\theta log\boldsymbol{P}(s_t, a_t|\tau)r_{t+1}$$

$$= \boldsymbol{E_\tau}[\textstyle\sum_{t=0}^{T-1} \nabla_\theta log\boldsymbol{P}(s_t, a_t|\tau)r_{t+1}]$$

강화학습에서는 E[]를 계산 하지 않고 Sampling!!

$$\rightarrow \sum_{t=0}^{T-1} \nabla_\theta log\boldsymbol{P}(s_t, a_t|\tau)r_{t+1}$$

알아야함

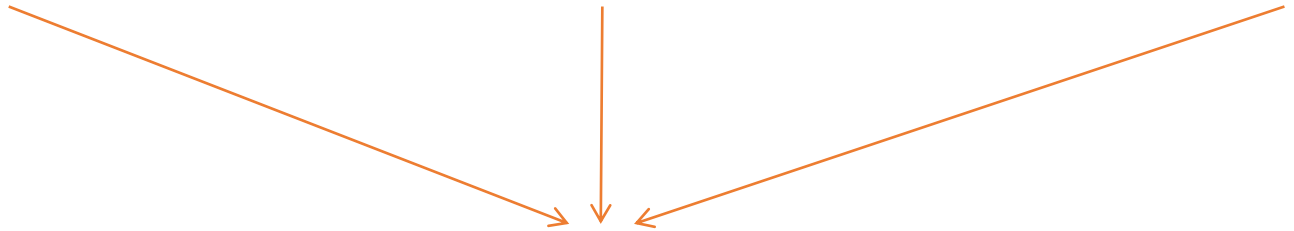
Policy gradient 수식 유도

여기서 더 나아가기 위해서는 $P(s_t, a_t|\tau)$를 파헤쳐야 한다!

$$\boldsymbol{P}(s_t, a_t|\tau) = \boldsymbol{P}(s_0, a_0, r_1, s_1, a_1, r_2, \cdots, s_t, a_t|\theta)$$

에이전트(agent)와 환경(env)의 상호작용 흔적

$$= \underbrace{\boldsymbol{P}(s_0)}_{\text{env}} \underbrace{\pi_\theta(a_0|s_0)}_{\text{agent}} \underbrace{\boldsymbol{P}(s_1|s_0, a_0)}_{\text{env}} \underbrace{\pi_\theta(a_1|s_1)}_{\text{agent}} \underbrace{\boldsymbol{P}(s_2|s_1, a_1)}_{\text{env}} \cdots$$



에이전트가 알 수 없는 정보

Policy gradient 수식 유도

목표 $\sum_{t=0}^{T-1} r_{t+1} \nabla_\theta log\boldsymbol{P}(s_t, a_t|\tau)$

기대하시라 …

$$log\boldsymbol{P}(s_t, a_t|\tau)$$

$$= log[\boldsymbol{P}(s_0)\pi_\theta(a_0|s_0)\boldsymbol{P}(s_1|s_0, a_0)\pi_\theta(a_1|s_1) \cdots \boldsymbol{P}(s_t|s_{t-1}, a_{t-1})\pi_\theta(a_t|s_t)]$$

$$logAB = logA + logB$$

$$= log\boldsymbol{P}(s_0) + log\pi_\theta(a_0|s_0) + log\boldsymbol{P}(s_1|s_0, a_0) + log\pi_\theta(a_1|s_1) \cdots$$

$$+ log\boldsymbol{P}(s_t|s_{t-1}, a_{t-1}) + log\pi_\theta(a_t|s_t)$$

Policy gradient 수식 유도

목표 $\sum_{t=0}^{T-1} r_{t+1} \nabla_\theta log\boldsymbol{P}(s_t, a_t|\tau)$

기대하시라 …

$$\nabla_\theta \, log\boldsymbol{P}(s_t, a_t|\tau)$$

$$= \nabla_\theta [log\boldsymbol{P}(s_0) + log\pi_\theta(a_0|s_0) + log\boldsymbol{P}(s_1|s_0, a_0) + log\pi_\theta(a_1|s_1) \cdots$$

$$+ log\boldsymbol{P}(s_t|s_{t-1}, a_{t-1}) + log\pi_\theta(a_t|s_t)]$$

$\nabla_\theta log\boldsymbol{P}(s_0), \nabla_\theta log\boldsymbol{P}(s_1|s_0, a_0), \cdots = 0$ 몰라도 돼!!

$$= \nabla_\theta \, log\pi_\theta(a_0|s_0) + \nabla_\theta log\pi_\theta(a_1|s_1) + \cdots + \nabla_\theta log\pi_\theta(a_t|s_t)$$

Policy gradient 수식 유도

목표 $\sum_{t=0}^{T-1} r_{t+1} \nabla_\theta log \boldsymbol{P}(s_t, a_t|\tau)$

$$\sum_{t=0}^{T-1} r_{t+1} \nabla_\theta log \boldsymbol{P}(s_t, a_t|\tau)$$

$$= \sum_{t=0}^{T-1} r_{t+1} [\nabla_\theta log \pi_\theta(a_0|s_0) + \nabla_\theta log \pi_\theta(a_1|s_1) + \cdots + \nabla_\theta log \pi_\theta(a_t|s_t)]$$

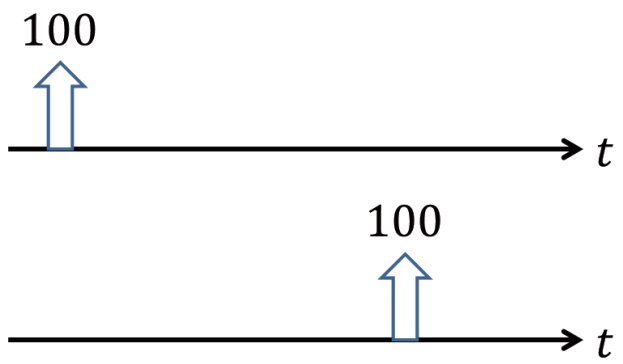
$$= \sum_{t=0}^{T-1} r_{t+1} \left(\sum_{t'=0}^{t} \nabla_\theta log \pi_\theta(a_{t'}|s_{t'}) \right)$$

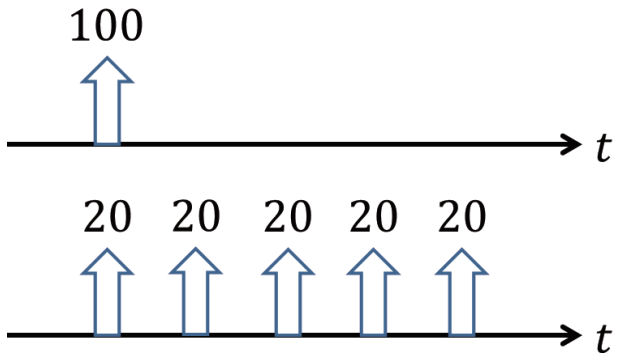
$$= \sum_{t=0}^{T-1} \nabla_\theta log \pi_\theta(a_t|s_t) \sum_{t'=t+1}^{T} r_{t'}$$

REINFORCE

$$\nabla_\theta J(\theta) = \boldsymbol{E}_{\boldsymbol{\tau}}\left[\sum_{t=0}^{T-1} \nabla_\theta log\pi_\theta(a_t|s_t) \sum_{t'=t+1}^{T} r_{t'}\right]$$

$\sum_{t'=t}^{T-1} r_{t'}$ **와 같은 단순 보상의 합 → 3가지 문제**





$$0.1 + 0.1 + \cdots = \infty$$
$$1 + 1 + \cdots = \infty$$

감가율(discount factor) $0 \leq \gamma \leq 1$의 도입

REINFORCE

$$\nabla_\theta J(\theta) = \boldsymbol{E}_{\boldsymbol{\tau}}\left[\sum_{t=0}^{T-1} \nabla_\theta log\pi_\theta(a_t|s_t) \sum_{t'=t+1}^{T} r_{t'}\right]$$

$$\sim \boldsymbol{E}_{\boldsymbol{\tau}}\left[\sum_{t=0}^{T-1} \nabla_\theta log\pi_\theta(a_t|s_t) \sum_{t'=t+1}^{T} \gamma^{t'-t-1} r_{t'}\right]$$

$$\sum_{t'=t+1}^{T} \gamma^{t'-t-1} r_{t'} = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots + \gamma^{T-t-1} r_T \rightarrow G_t$$

Discounted future reward = Return G

REINFORCE

$$\nabla_\theta J(\theta) = \boldsymbol{E}_{\boldsymbol{\tau}}\left[\sum_{t=0}^{T-1} \nabla_\theta log\pi_\theta(a_t|s_t)\, G_t\right]$$

Sampling으로 E[]을 대체 → 강화학습

한 에피소드 : 하나의 샘플

$$\nabla_\theta J(\theta) \sim \sum_{t=0}^{T-1} \nabla_\theta log\pi_\theta(a_t|s_t)\, G_t$$

$$\theta \leftarrow \theta + \alpha\nabla_\theta J(\theta) = \boxed{\theta + \alpha \sum_{t=0}^{T-1} \nabla_\theta log\pi_\theta(a_t|s_t)\, G_t}$$

REINFORCE 알고리즘 업데이트 식

REINFORCE

1. 한 에피소드를 현재 정책에 따라 실행
2. Trajectory를 기록
3. 에피소드가 끝난 뒤 G_t 계산
4. Policy gradient를 계산해서 정책 업데이트 Policy gradient $= \sum_{t=0}^{T-1} \nabla_\theta log\pi_\theta(a_t|s_t)\, G_t$
5. (1~4) 반복

에피소드 마다 업데이트 → 몬테카를로 Policy gradient = REINFORCE

REINFORCE

REINFORCE의 문제

1. Variance가 높다

2. 에피소드마다 업데이트가 가능하다(on-line X)

몬테카를로 → TD(Temporal-Difference)

REINFORCE → Actor-Critic

Actor-Critic

Actor-Critic

$$\nabla_\theta J(\theta) = \boldsymbol{E}_{\tau}\left[\sum_{t=0}^{T-1} \nabla_\theta log\pi_\theta(a_t|s_t)\, G_t\right]$$

Expectation을 쪼개보자 → $(s_0 \sim a_t) + (r_{t+1} \sim r_T)$

$$\sum_{t=0}^{T-1} \boldsymbol{E}_{\boldsymbol{s_0},\boldsymbol{a_0},\cdots,\boldsymbol{s_t},\boldsymbol{a_t}}[\nabla_\theta log\pi_\theta(a_t|s_t)]\, \boldsymbol{E}_{\boldsymbol{r_{t+1}},\boldsymbol{s_{t+1}},\cdots,\boldsymbol{s_T},\boldsymbol{r_T}}[G_t]$$

어디선가 봤던 이 수식은 바로 Q-function

$$\boldsymbol{E}_{\boldsymbol{r_{t+1}},\boldsymbol{s_{t+1}},\cdots,\boldsymbol{s_T},\boldsymbol{r_T}}[G_t] = Q(s_t, a_t)$$

Actor-Critic

$$\nabla_\theta J(\theta) = \boldsymbol{E}_{\boldsymbol{\tau}} \left[\sum_{t=0}^{T-1} \nabla_\theta log \pi_\theta(a_t|s_t)\, Q_{\pi_\theta}(s_t, a_t) \right]$$

$$\sim \sum_{t=0}^{T-1} \nabla_\theta log \pi_\theta(a_t|s_t)\, Q_{\pi_\theta}(s_t, a_t)$$

만약 $Q_{\pi_\theta}(s_t, a_t)$를 알 수 있다면 매 time-step마다 업데이트하는 것이 가능!!

$Q_{\pi_\theta}(s_t, a_t) \sim Q_w(s_t, a_t)$ Let's make Critic!!!!

Actor-Critic

$$\nabla_\theta J(\theta) \sim \nabla_\theta log \pi_\theta(a_t|s_t) Q_w(s_t, a_t)$$

Advantage 함수 = 큐함수 − 베이스라인 → Variance를 낮춰준다

큐함수 : 특정 상태, 특정 행동에 따른 값

가치함수 : 특정 상태, 전반적 행동에 따른 값 → 베이스라인

$$\nabla_\theta J(\theta) \sim \nabla_\theta log \pi_\theta(a_t|s_t)(Q_w(s_t, a_t) - V_v(s_t))$$

Actor-Critic

$$\nabla_\theta J(\theta) \sim \nabla_\theta log\pi_\theta(a_t|s_t)(Q_w(s_t,a_t) - V_v(s_t))$$

Q와 V를 둘 다 근사하는 건(ex 뉴럴넷) 비효율적

$$Q(s_t,a_t) = \boldsymbol{E}[r_{t+1} + \gamma V(s_{t+1})|s_t,a_t]$$**을 이용!**

$$\nabla_\theta J(\theta) \sim \nabla_\theta log\pi_\theta(a_t|s_t)(\underbrace{r_{t+1} + \gamma V_v(s_{t+1})}_{\text{큐함수}} - \underbrace{V_v(s_t)}_{\text{베이스라인}})$$

Actor-Critic

1. Actor

1) 정책을 근사: θ

2) $\nabla_\theta log\pi_\theta(a_t|s_t)(r_{t+1} + \gamma V_v(s_{t+1}) - V_v(s_t))$로 업데이트

2. Critic

1) 가치함수(Value function)을 근사: v

2) $\left(r_{t+1} + \gamma V_v(s_{t+1}) - V_v(s_t)\right)^2$의 오차함수로 업데이트

Actor-Critic

1. Actor의 loss function

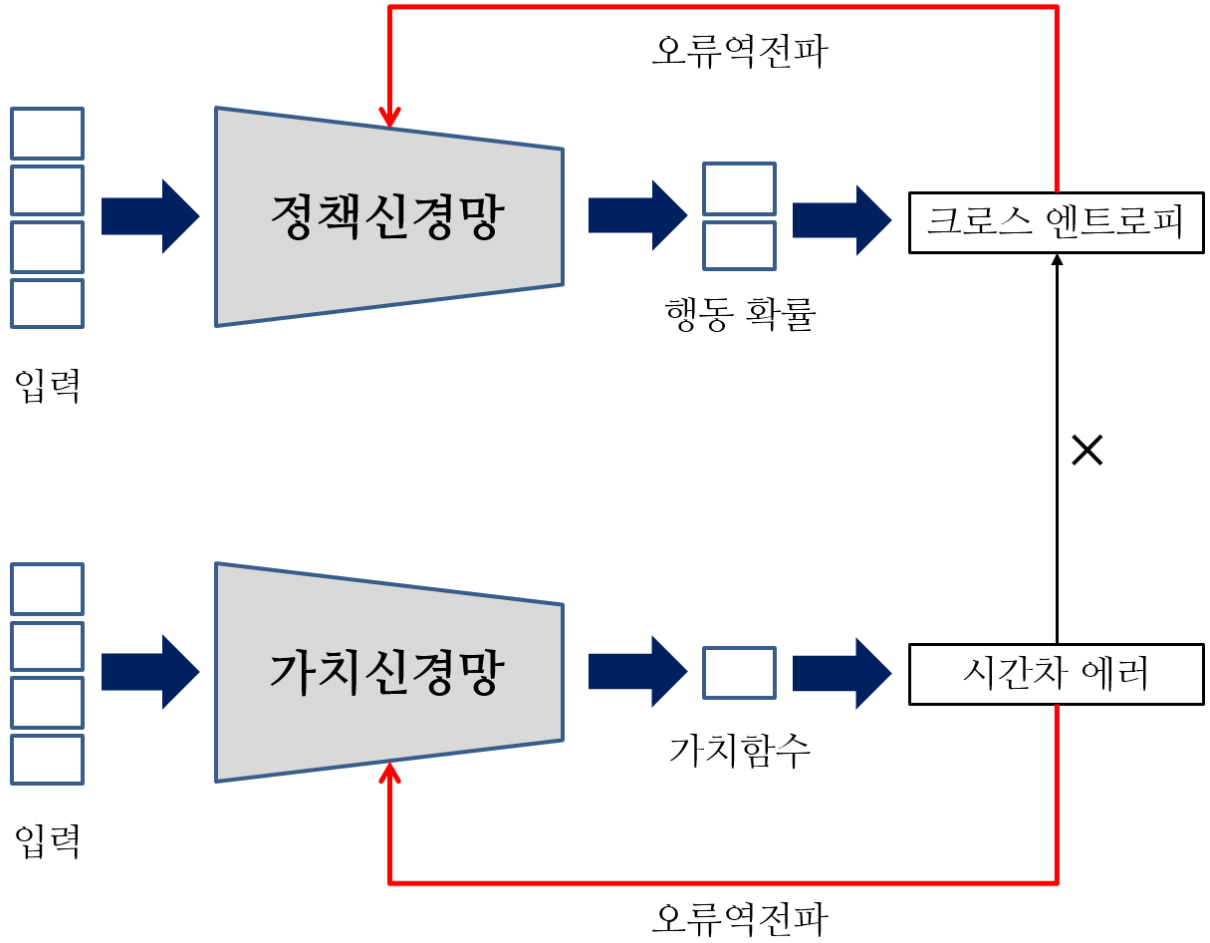
$$-\nabla_\theta log\pi_\theta(a_t|s_t)(r_{t+1} + \gamma V_v(s_{t+1}) - V_v(s_t))$$

크로스 엔트로피 — 시간차 에러

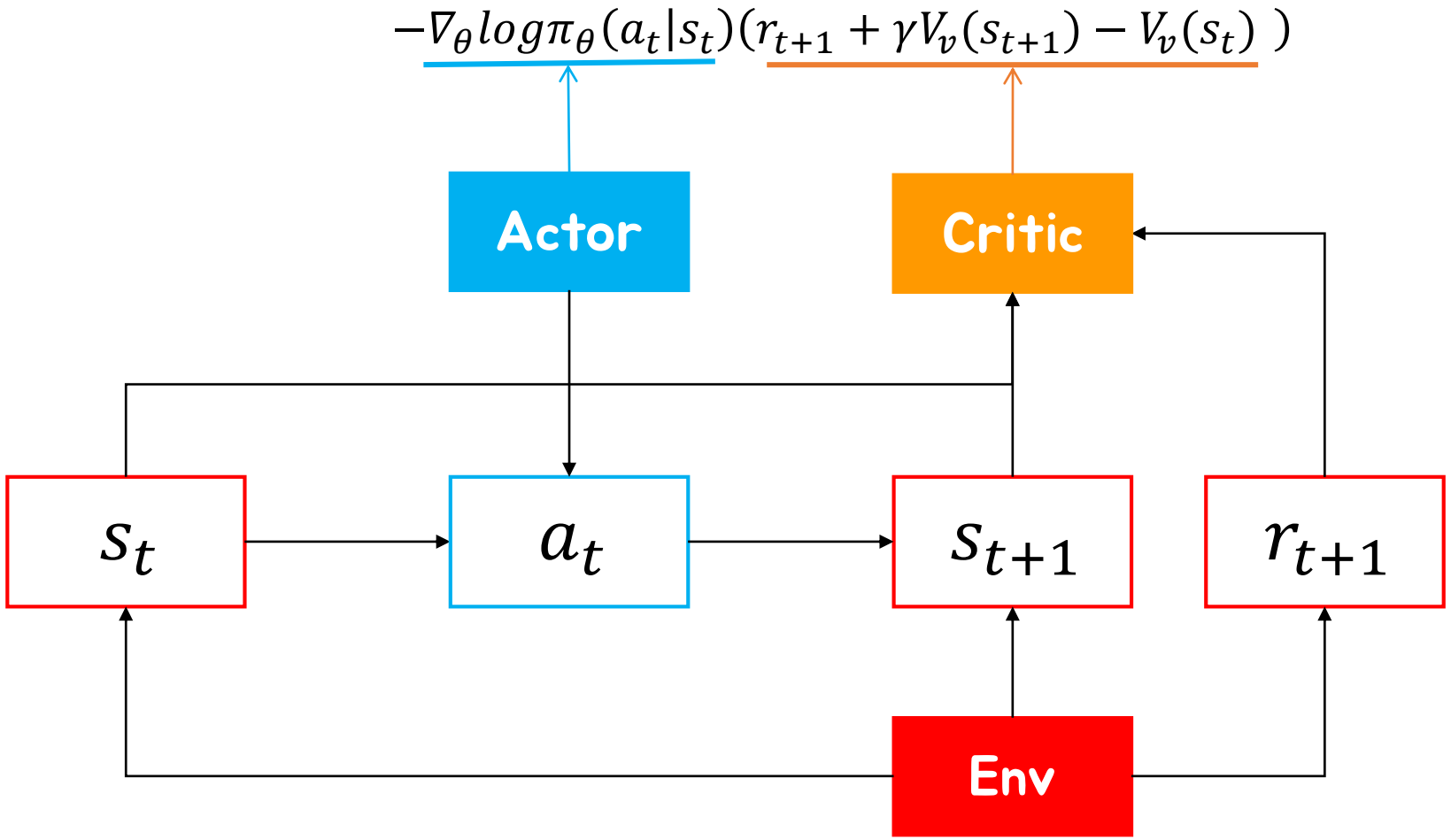
2. Critic의 loss function

$$\left(r_{t+1} + \gamma V_v(s_{t+1}) - V_v(s_t)\right)^2$$

시간차 에러



Actor-Critic



A3C

Actor-Critic과 다른 점

Actor를 업데이트하는 과정에서

1. Multi-step loss function

2. Entropy loss function

A3C

1. Multi-step loss function

$$-\nabla_\theta log\pi_\theta(a_t|s_t)(r_{t+1} + \gamma V_v(s_{t+1}) - V_v(s_t))$$

↓

$$-\nabla_\theta log\pi_\theta(a_t|s_t)(r_{t+1} + \gamma r_{t+2} + \gamma^2 V_v(s_{t+2}) - V_v(s_t))$$

↓

$$-\nabla_\theta log\pi_\theta(a_t|s_t)(r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{19} V_v(s_{t+20}) - V_v(s_t))$$

1-step

multi-step

20 step을 가본 후에 loss function이 하나 나온다?

A3C

1. Multi-step

$$loss\ funtion = \nabla_\theta log\pi_\theta(a_t|s_t)(r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{19} V_v(s_{t+20}) - V_v(s_t))$$

$$+ \nabla_\theta log\pi_\theta(a_{t+1}|s_{t+1})(r_{t+2} + \gamma r_{t+3} + \cdots + \gamma^{18} V_v(s_{t+20}) - V_v(s_{t+1}))$$

$$+ \nabla_\theta log\pi_\theta(a_{t+2}|s_{t+2})(r_{t+3} + \gamma r_{t+4} + \cdots + \gamma^{17} V_v(s_{t+20}) - V_v(s_{t+2}))$$

$$+ \cdots +$$

$$\nabla_\theta log\pi_\theta(a_{t+19}|s_{t+19})(r_{t+19} + \gamma V_v(s_{t+20}) - V_v(s_{t+19}))$$

20 step 마다 20개의 loss function을 더한 것으로 업데이트

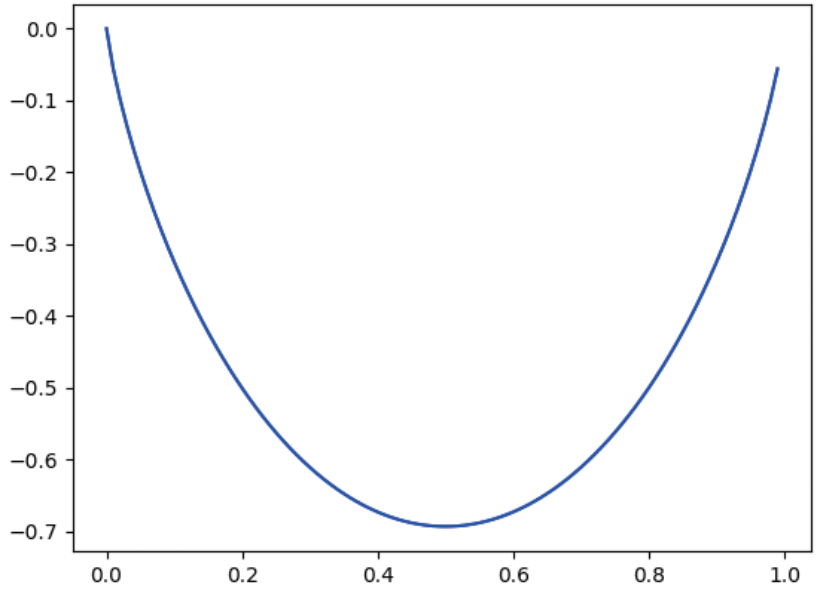
A3C

2. Entropy loss function

엔트로피의 정의: $-\sum_i p_i log p_i$

→ "거꾸로" $\sum_i p_i log p_i$

→ gradient descent 하기 위해!



행동이 두 가지 일 때 행동 확률에 따른 엔트로피의 그래프

A3C

2. Entropy loss function

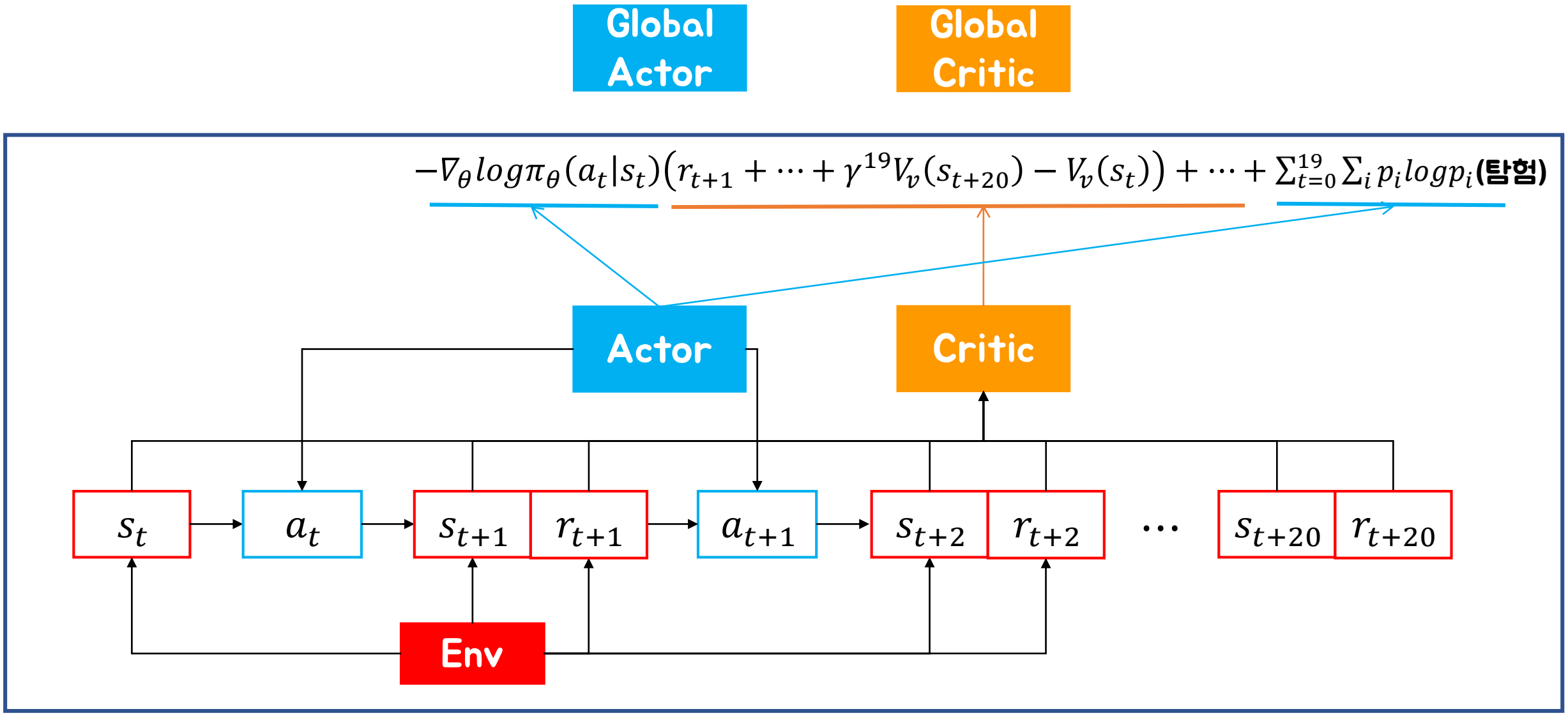
Actor-Critic의 loss function

$$-\nabla_\theta log\pi_\theta(a_t|s_t)(r_{t+1} + \gamma V_v(s_{t+1}) - V_v(s_t)\ )$$

A3C의 loss function

$$\underbrace{-\nabla_\theta log\pi_\theta(a_{t+19}|s_{t+19})(r_{t+1} + \cdots + \gamma^{19} V_v(s_{t+20}) - V_v(s_t)\ )\cdots}_{\text{20개의 cross entropy: exploitation}} + \underbrace{\sum_{t=0}^{19}\sum_i p_i log p_i}_{\text{Entropy: exploration}}$$

A3C



Thread 1 → 여러 개의 Thread

여기까지 A3C를
직관적 + 수식적으로
이해해봤습니다

질문

감사합니다