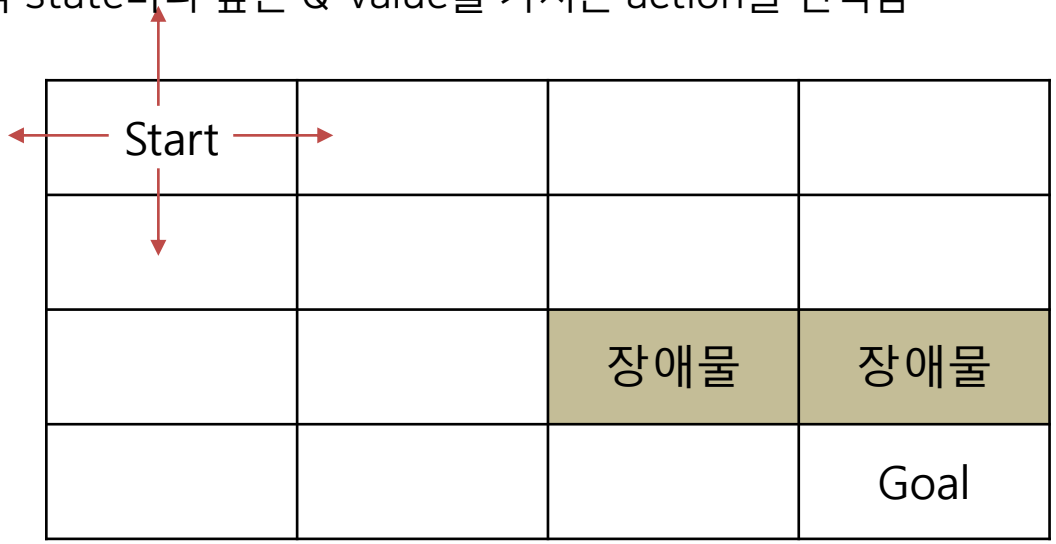
Reinforcement Learning

강화학습

- Start에서 Goal로 가고 싶음

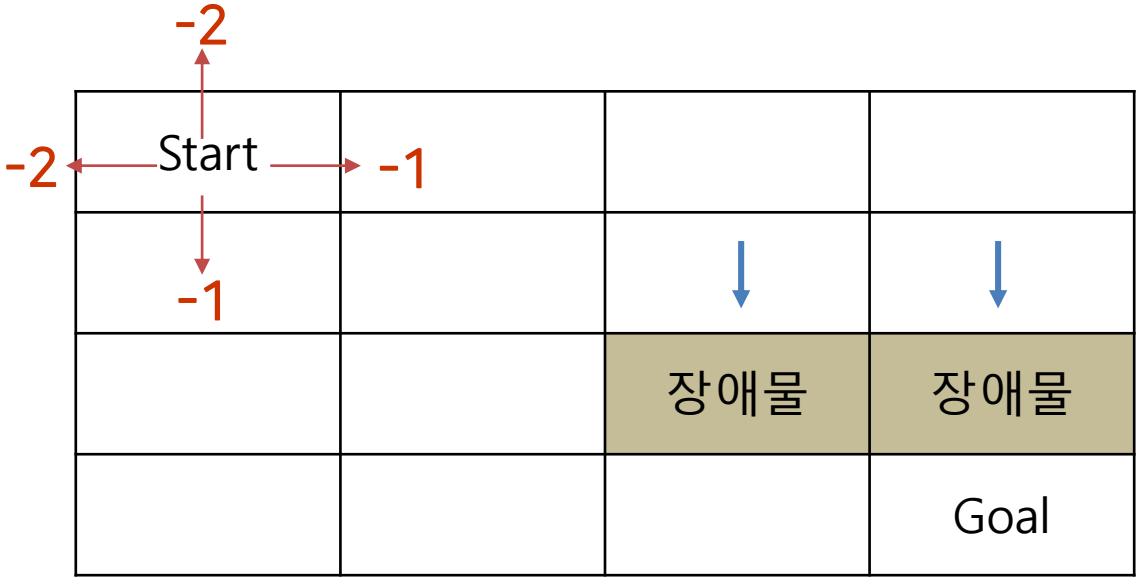
강화학습

- 컴퓨터가 가게 하고 싶음… 어떻게?
- (1, 1)에서 취할 수 있는 행동은 네가지. 이때의 (1, 1)을 현재 상태(state), 행동을 action이라 정의
- (1, 1)에서 왼쪽으로 가는 것과 위로 가는 것은 안좋음. 컴퓨터에게 이 행동은 좋지 않다라는 feedback을 주어야함. 이때의 feedback이 reward
- Feedback을 통해 현재 상태에서 어떤 action을 취하는게 좋은지 학습하게 됨. 이때의 action에 대한 값이 Q-value라 함
- 강화학습은 각 state마다 높은 Q-value를 가지는 action을 선택함



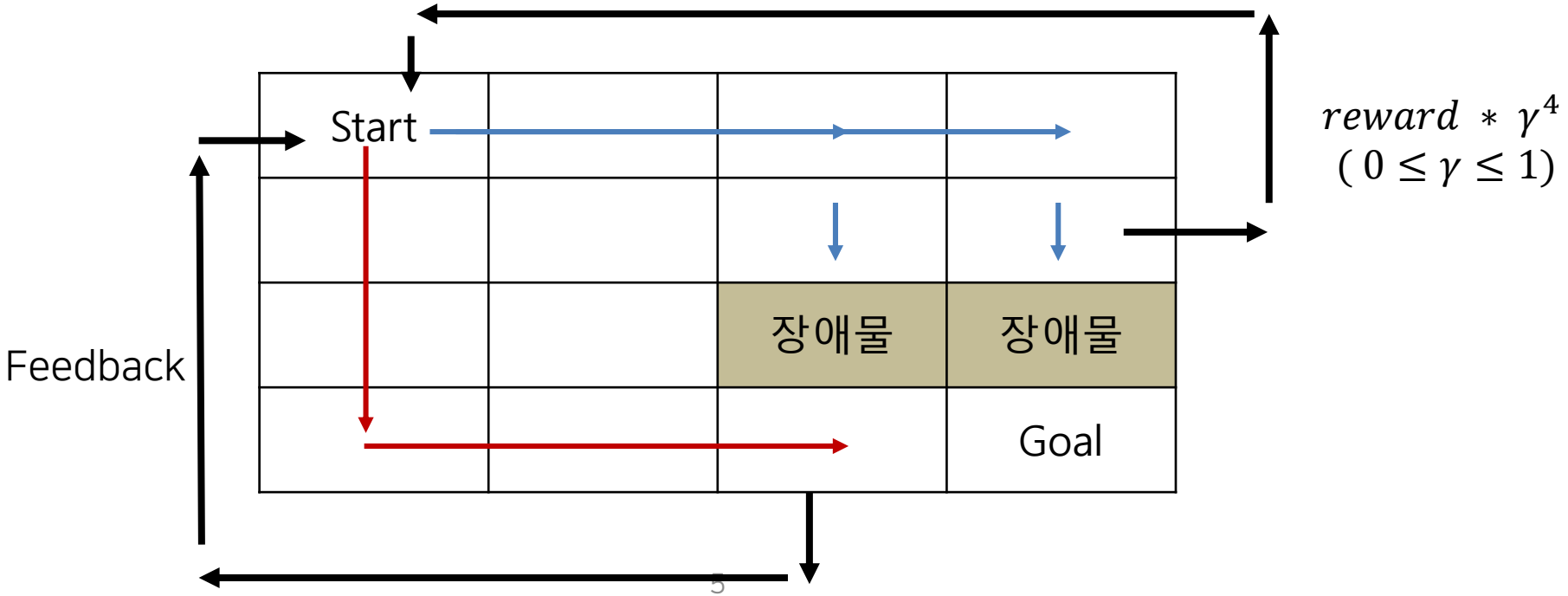
강화학습

- 강화학습 : 달성하고자 하는 목표에 대해서 각 state에서 action에 대한 최적의 Q-value를 학습하는 것
- Reward를 설정해보자. 한 칸 이동하면 -1, 이동을 못하면 -2
- (1, 1)에서 여러 번 경험을 통해서 각 action(left, right, top, down)에 대한 q-value는 (-2, -2, -1,-1) 이 됨.
- 그런데, right로 가면 장애물을 만날 확률이 올라감. (1, 1)에서 (right, down)이 같은 qvalue를 가지다면 50% 확률로 Goal 에 도착하지 못함



강화학습

- (1, 1)에서 오른쪽으로 간다면 미래에 좋지 않다 라는 feed back이 필요함
- 반대로 아래쪽으로 간다면 미래에 더 좋다 라는 feed back이 필요함
- 이 Feedback을 반영하기 위해 Discount factor를 도입
- 이 Feedback을 반영하기 위해 각 state에서 미래 보상의 누적합을 최대화 하는것을 목적으로함



강화학습

- 강화학습 : 달성하고자 하는 목표에 대해서 각 state에서 action에 대한 최적의 Q-value를 학습하는 것 -> 현재 state에서 미래 보상의 합을 최대로 하게끔 학습

$$V^{\pi}(s_t) = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots = \sum_{i=0}^{\infty} \gamma^i r_{t+i}$$

$$\pi^* = argmax_{\pi} V^{\pi}(s) \qquad for\ all\ s$$

  - 학습하고자 하는 정책(목표, policy) π 는 모든 states s에 대하여 $V^{\pi}(s)$ 를 최대화 하는것을 목적으로 함

강화학습

State s, action a, reward r, r(s, a) 즉시 reward, γ discount factor, π 정책(학습하고자 하는 것)

$$V^{\pi}(s_t) = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots = \sum_{i=0}^{\infty} \gamma^i r_{t+i}$$

$$\pi^* = argmax_{\pi} V^{\pi}(s) \qquad for\ all\ s$$

State에 따른 optimal $V^*(s)$와 $\pi^*(s)$를 다음과 같이 쓸 수 있음

$$V^*(s) = max_a \left[r(s,a) + \gamma V^*\left(\delta(s,a)\right)\right]$$

$$\pi^*(s) = argmax_a \left[r(s,a) + \gamma V^*\left(\delta(s,a)\right)\right] \rightarrow Q(s,a)$$

$$V^*(s) = max_a\ Q(s,a)$$

$$\pi^*(s) = argmax_a\ Q(s,a)$$

2 Inflearn Q-learning

Q learning

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

즉시 reward + 다음 state에서의 가장 큰 q값 * discount factor

Q learning algorithm

For each (s,a) initialize the table entry $\hat{Q}(s,a)$ to zero

Repeat (for each episode)

Observe the initial state s

Repeat (for each step of episode)

Select an action a from state s (e.g. ε-greedy) and execute it

Receive immediate reward r

Observe the new state s'

Update the table entry for $\hat{Q}(s,a)$ as follows:

$\hat{Q}(s,a) \leftarrow r + \gamma \, max_{a'} \hat{Q}(s',a')$

$s \leftarrow s'$

Until s is terminal (goal state)

Exploration (탐험)

2 Inflearn Q-learning

Q learning

Exploitation (착취, 이용)

- 이전에 경험했던 것을 다시 이용

Exploration (탐험)

- 더 좋은 policy를 찾기 위한 행동

강화학습에서는 이 두 요소를 적당히 다루는 것이 주 요소(특히, exploration)

2 Inflearn Q-learning(예시)

Start			
		장애물	
	장애물	Goal	

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

초기 Q-value

모든 State, 모든 action에 대한 Q-value : 0.5

Discount factor : 0.9

2 Inflearn Q-learning(예시)

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \, \hat{Q}(s',a')$$

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

Index	왼쪽	오른쪽	위	아래
1	0.5	0.5	0.5	0.5
2	0.5	0.5	0.5	0.5
3	0.5	0.5	0.5	0.5
4	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	0.5
6	0.5	0.5	0.5	0.5
7	0.5	0.5	0.5	0.5
8	0.5	0.5	0.5	0.5
9	0.5	0.5	0.5	0.5
10	0.5	0.5	0.5	0.5
11	0.5	0.5	0.5	0.5
12	0.5	0.5	0.5	0.5

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

7, down : 1

3, down : -10 + 0.9 x (0.5) = - 9.55

2, right : -1 + 0.9 x (0.5) = - 0.55

1, right : -1 + 0.9 x (0.5) = - 0.55

Index	왼쪽	오른쪽	위	아래
1	0.5	0.5	0.5	0.5
2	0.5	0.5	0.5	0.5
3	0.5	0.5	0.5	0.5
4	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	0.5
6	0.5	0.5	0.5	0.5
7	0.5	0.5	0.5	0.5
8	0.5	0.5	0.5	0.5
9	0.5	0.5	0.5	0.5
10	0.5	0.5	0.5	0.5
11	0.5	0.5	0.5	0.5
12	0.5	0.5	0.5	0.5

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

Index	왼쪽	오른쪽	위	아래
1	0.5	0.5	0.5	0.5
2	0.5	0.5	0.5	0.5
3	0.5	0.5	0.5	0.5
4	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	0.5
6	0.5	0.5	0.5	0.5
7	0.5	0.5	0.5	0.5
8	0.5	0.5	0.5	0.5
9	0.5	0.5	0.5	0.5
10	0.5	0.5	0.5	0.5
11	0.5	0.5	0.5	0.5
12	0.5	0.5	0.5	0.5

Index	왼쪽	오른쪽	위	아래
1	0.5	-0.55	0.5	0.5
2	0.5	-0.55	0.5	0.5
3	0.5	0.5	0.5	-9.55
4	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	0.5
6	0.5	0.5	0.5	0.5
7	0.5	0.5	0.5	1
8	0.5	0.5	0.5	0.5
9	0.5	0.5	0.5	0.5
10	0.5	0.5	0.5	0.5
11	0.5	0.5	0.5	0.5
12	0.5	0.5	0.5	0.5

2 Inflearn Q-learning(예시)

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

10, right : 1

9, right : -10 + 0.9 x (0.5) = - 9.55

5, down : -1 + 0.9 x (0.5) = - 0.55

1, down : -1 + 0.9 x (0.5) = - 0.55

1, right : -2 + 0.9 x (0.5) = - 1.55

1, up : -2 + 0.9 x (0.5) = - 1.55

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

Index	왼쪽	오른쪽	위	아래
1	0.5	-0.55	0.5	0.5
2	0.5	-0.55	0.5	0.5
3	0.5	0.5	0.5	-9.55
4	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	0.5
6	0.5	0.5	0.5	0.5
7	0.5	0.5	0.5	1
8	0.5	0.5	0.5	0.5
9	0.5	0.5	0.5	0.5
10	0.5	0.5	0.5	0.5
11	0.5	0.5	0.5	0.5
12	0.5	0.5	0.5	0.5

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

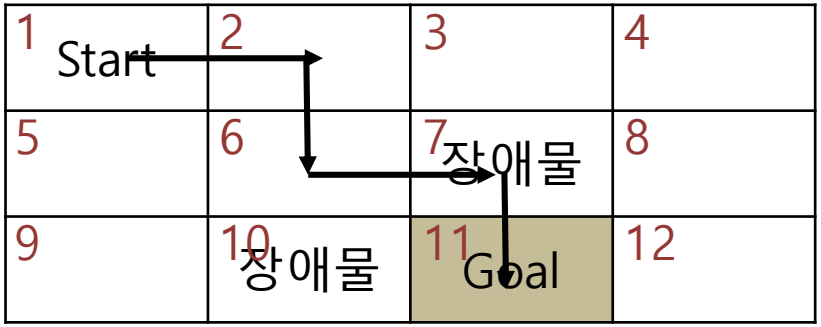
Index	왼쪽	오른쪽	위	아래
1	0.5	-0.55	0.5	0.5
2	0.5	-0.55	0.5	0.5
3	0.5	0.5	0.5	-9.55
4	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	0.5
6	0.5	0.5	0.5	0.5
7	0.5	0.5	0.5	1
8	0.5	0.5	0.5	0.5
9	0.5	0.5	0.5	0.5
10	0.5	0.5	0.5	0.5
11	0.5	0.5	0.5	0.5
12	0.5	0.5	0.5	0.5

Index	왼쪽	오른쪽	위	아래
1	-1.55	-0.55	-1.55	-0.55
2	0.5	-0.55	0.5	0.5
3	0.5	0.5	0.5	-9.55
4	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	-0.55
6	0.5	0.5	0.5	0.5
7	0.5	0.5	0.5	1
8	0.5	0.5	0.5	0.5
9	0.5	-9.55	0.5	0.5
10	0.5	1	0.5	0.5
11	0.5	0.5	0.5	0.5
12	0.5	0.5	0.5	0.5

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1



$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

7, down : 1

6, right : -10 + 0.9 x (1) = - 9.1

2, down : -1 + 0.9 x (0.5) = - 0.55

1, right : -1 + 0.9 x (0.5) = - 0.55

Index	왼쪽	오른쪽	위	아래
1	-1.55	-0.55	-1.55	-0.55
2	0.5	-0.55	0.5	0.5
3	0.5	0.5	0.5	-9.55
4	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	-0.55
6	0.5	0.5	0.5	0.5
7	0.5	0.5	0.5	1
8	0.5	0.5	0.5	0.5
9	0.5	-9.55	0.5	0.5
10	0.5	1	0.5	0.5
11	0.5	0.5	0.5	0.5
12	0.5	0.5	0.5	0.5

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

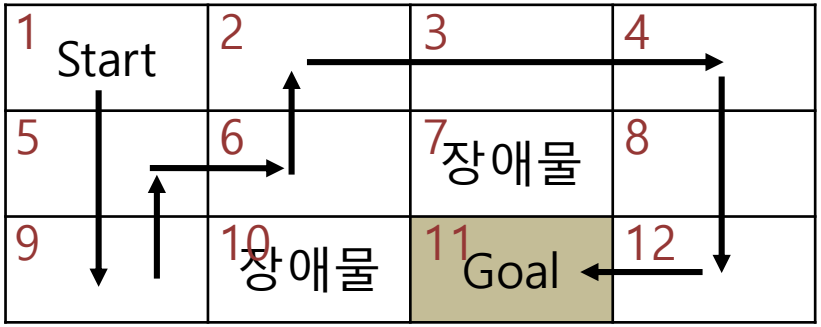
Index	왼쪽	오른쪽	위	아래
1	-1.55	-0.55	-1.55	-0.55
2	0.5	-0.55	0.5	0.5
3	0.5	0.5	0.5	-9.55
4	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	-0.55
6	0.5	0.5	0.5	0.5
7	0.5	0.5	0.5	1
8	0.5	0.5	0.5	0.5
9	0.5	-9.55	0.5	0.5
10	0.5	1	0.5	0.5
11	0.5	0.5	0.5	0.5
12	0.5	0.5	0.5	0.5

Index	왼쪽	오른쪽	위	아래
1	-1.55	-0.55	-1.55	-0.55
2	0.5	-0.55	0.5	-0.55
3	0.5	0.5	0.5	-9.55
4	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	-0.55
6	0.5	-9.1	0.5	0.5
7	0.5	0.5	0.5	1
8	0.5	0.5	0.5	0.5
9	0.5	-9.55	0.5	0.5
10	0.5	1	0.5	0.5
11	0.5	0.5	0.5	0.5
12	0.5	0.5	0.5	0.5

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1



$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

12, left : 1
8, down : -1 + 0.9 x (0.5) = - 0.55
4, down : -1 + 0.9 x (0.5) = - 0.55
3, right : -1 + 0.9 x (0.5) = - 0.55
2, right : -1 + 0.9 x (0.5) = - 0.55
6, up : -1 + 0.9 x (0.5) = - 0.55
5, right : -1 + 0.9 x (0.5) = - 0.55
9, up : -1 + 0.9 x (0.5) = - 0.55
5, down : -1 + 0.9 x (0.5) = - 0.55
1, down : -1 + 0.9 x (0.5) = - 0.55

Index	왼쪽	오른쪽	위	아래
1	-1.55	-0.55	-1.55	-0.55
2	0.5	-0.55	0.5	-0.55
3	0.5	0.5	0.5	-9.55
4	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	-0.55
6	0.5	-9.1	0.5	0.5
7	0.5	0.5	0.5	1
8	0.5	0.5	0.5	0.5
9	0.5	-9.55	0.5	0.5
10	0.5	1	0.5	0.5
11	0.5	0.5	0.5	0.5
12	0.5	0.5	0.5	0.5

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

Index	왼쪽	오른쪽	위	아래
1	-1.55	-0.55	-1.55	-0.55
2	0.5	-0.55	0.5	-0.55
3	0.5	0.5	0.5	-9.55
4	0.5	0.5	0.5	0.5
5	0.5	0.5	0.5	-0.55
6	0.5	-9.1	0.5	0.5
7	0.5	0.5	0.5	1
8	0.5	0.5	0.5	0.5
9	0.5	-9.55	0.5	0.5
10	0.5	1	0.5	0.5
11	0.5	0.5	0.5	0.5
12	0.5	0.5	0.5	0.5

Index	왼쪽	오른쪽	위	아래
1	-1.55	-0.55	-1.55	-0.55
2	0.5	-0.55	0.5	-0.55
3	0.5	-0.55	0.5	-9.55
4	0.5	0.5	0.5	-0.55
5	0.5	-0.55	0.5	-0.55
6	0.5	-9.1	-0.55	0.5
7	0.5	0.5	0.5	1
8	0.5	0.5	0.5	-0.55
9	0.5	-9.55	-0.55	0.5
10	0.5	1	0.5	0.5
11	0.5	0.5	0.5	0.5
12	1	0.5	0.5	0.5

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

12, left : 1
8, down : -1 + 0.9 x (1) = - 0.1
7, right : -1 + 0.9 x (0.5) = - 0.55
6, right : -10 + 0.9 x (1) = -9.1
10, up : -1 + 0.9 x (1) = - 0.55
9, right : -10 + 0.9 x (1) = - 9.1
5, down : -1 + 0.9 x (1) = - 0.55
1, down : -1 + 0.9 x (1) = - 0.55

Index	왼쪽	오른쪽	위	아래
1	-1.55	-0.55	-1.55	-0.55
2	0.5	-0.55	0.5	-0.55
3	0.5	-0.55	0.5	-9.55
4	0.5	0.5	0.5	-0.55
5	0.5	-0.55	0.5	-0.55
6	0.5	-9.1	-0.55	0.5
7	0.5	0.5	0.5	1
8	0.5	0.5	0.5	-0.55
9	0.5	-9.55	-0.55	0.5
10	0.5	1	0.5	0.5
11	0.5	0.5	0.5	0.5
12	1	0.5	0.5	0.5

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

Index	왼쪽	오른쪽	위	아래
1	-1.55	-0.55	-1.55	-0.55
2	0.5	-0.55	0.5	-0.55
3	0.5	-0.55	0.5	-9.55
4	0.5	0.5	0.5	-0.55
5	0.5	-0.55	0.5	-0.55
6	0.5	-9.1	-0.55	0.5
7	0.5	0.5	0.5	1
8	0.5	0.5	0.5	-0.55
9	0.5	-9.55	-0.55	0.5
10	0.5	1	0.5	0.5
11	0.5	0.5	0.5	0.5
12	1	0.5	0.5	0.5

Index	왼쪽	오른쪽	위	아래
1	-1.55	-0.55	-1.55	-0.55
2	0.5	-0.55	0.5	-0.55
3	0.5	-0.55	0.5	-9.55
4	0.5	0.5	0.5	-0.55
5	0.5	-0.55	0.5	-0.55
6	0.5	-9.1	-0.55	0.5
7	0.5	-0.55	0.5	1
8	0.5	0.5	0.5	-0.1
9	0.5	-9.55	-0.55	0.5
10	0.5	1	-0.55	0.5
11	0.5	0.5	0.5	0.5
12	1	0.5	0.5	0.5

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

1 Start	2	3	4
5	6	7 장애물	8
9	10 장애물	11 Goal	12

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

이동을 했을 때, 주변에 장애물이나 Goal 이 없다면 -0.55
이동을 했을 때, 제자리이면 -1.55
이동을 했을 때, 장애물이면 -9.55
모두 업데이트!!

Index	왼쪽	오른쪽	위	아래
1	-1.55	-0.55	-1.55	-0.55
2	0.5	-0.55	0.5	-0.55
3	0.5	-0.55	0.5	-9.55
4	0.5	0.5	0.5	-0.55
5	0.5	-0.55	0.5	-0.55
6	0.5	-9.1	-0.55	0.5
7	0.5	-0.55	0.5	1
8	0.5	0.5	0.5	-0.1
9	0.5	-9.55	-0.55	0.5
10	0.5	1	-0.55	0.5
11	0.5	0.5	0.5	0.5
12	1	0.5	0.5	0.5

Q-learning(예시)

1 Start	2	3	4
5	6	7 장애물	8
9	10 장애물	11 Goal	12

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

이동을 했을 때, 주변에 장애물이나 Goal 이 없다면 -0.55
이동을 했을 때, 제자리이면 -1.55
이동을 했을 때, 장애물이면 -9.55
모두 업데이트!!

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

Index	왼쪽	오른쪽	위	아래
1	-1.55	-0.55	-1.55	-0.55
2	-0.55	-0.55	-1.55	-0.55
3	-0.55	-0.55	-1.55	-9.55
4	-0.55	-1.55	-1.55	-0.55
5	-1.55	-0.55	-0.55	-0.55
6	-0.55	-9.1	-0.55	-9.55
7	-0.55	-0.55	-0.55	1
8	-9.55	-1.55	-0.55	-0.1
9	-1.55	-9.55	-0.55	-1.55
10	-0.55	1	-0.55	-1.55
11	0.5	0.5	0.5	0.5
12	1	-1.55	-0.55	-1.55

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

1 Start	2	3	4
5	6	7 장애물	8
9	10 장애물	11 Goal	12

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

1 위치에서, 오른쪽이나 아래로 가는 것으로 선택
2 위치에서, 위로 가는 것 제외하고 선택
3 위치에서, 오른쪽이나 왼쪽으로 가는 것 선택
4 위치에서 왼쪽이나 아래쪽으로 가는 것 선택
..
12 위치에서 왼쪽으로 가는 것 선택

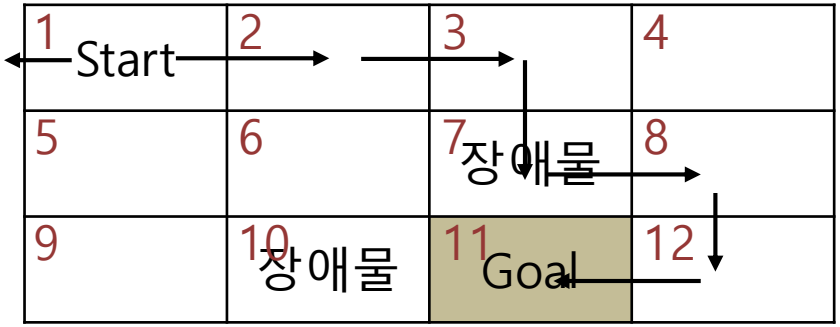
(한 단계 앞만 바라 봤을 때 최적의 방향성 제시하게 끔 학습됨)

Index	왼쪽	오른쪽	위	아래
1	-1.55	**-0.55**	-1.55	**-0.55**
2	**-0.55**	**-0.55**	-1.55	**-0.55**
3	**-0.55**	**-0.55**	-1.55	-9.55
4	**-0.55**	-1.55	-1.55	**-0.55**
5	-1.55	**-0.55**	**-0.55**	**-0.55**
6	**-0.55**	-9.1	**-0.55**	-9.55
7	-0.55	-0.55	-0.55	**1**
8	-9.55	-1.55	-0.55	**-0.1**
9	-1.55	-9.55	**-0.55**	-1.55
10	-0.55	**1**	-0.55	-1.55
11	0.5	0.5	0.5	0.5
12	**1**	-1.55	-0.55	-1.55

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1



$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

12, left : 1
8, down : -1 + 0.9 x (1) = - 0.1
7, right : -1 + 0.9 x (-0.1) = -1.09
3, down : -10 + 0.9 x (1) = -9.1
2, right : -1 + 0.9 x (-0.55) = - 1.495
1, right : -1 + 0.9 x (-0.55) = - 1.495
1, left : -2 + 0.9 x (-0.55) = - 2.495

Index	왼쪽	오른쪽	위	아래
1	-1.55	**-0.55**	-1.55	**-0.55**
2	**-0.55**	**-0.55**	-1.55	**-0.55**
3	**-0.55**	**-0.55**	-1.55	-9.55
4	**-0.55**	-1.55	-1.55	**-0.55**
5	-1.55	**-0.55**	**-0.55**	**-0.55**
6	**-0.55**	-9.1	**-0.55**	-9.55
7	-0.55	-0.55	-0.55	**1**
8	-9.55	-1.55	-0.55	**-0.1**
9	-1.55	-9.55	**-0.55**	-1.55
10	-0.55	**1**	-0.55	-1.55
11	0.5	0.5	0.5	0.5
12	**1**	-1.55	-0.55	-1.55

2 Inflearn Q-learning(예시)

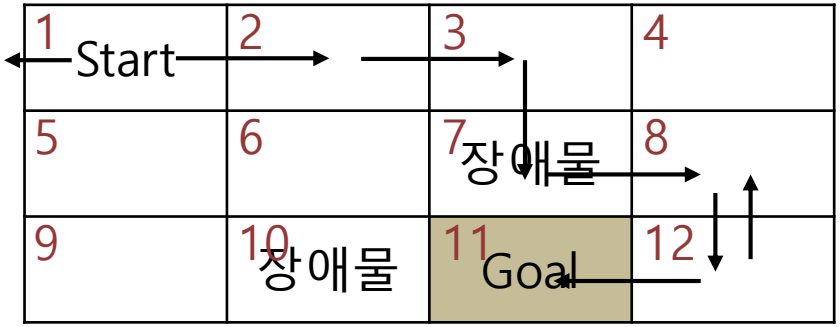
reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

Index	왼쪽	오른쪽	위	아래
1	-1.55	**-0.55**	-1.55	**-0.55**
2	**-0.55**	**-0.55**	-1.55	**-0.55**
3	**-0.55**	**-0.55**	-1.55	-9.55
4	**-0.55**	-1.55	-1.55	**-0.55**
5	-1.55	**-0.55**	**-0.55**	**-0.55**
6	**-0.55**	-9.1	**-0.55**	-9.55
7	-0.55	-0.55	-0.55	**1**
8	-9.55	-1.55	-0.55	**-0.1**
9	-1.55	-9.55	**-0.55**	-1.55
10	-0.55	**1**	-0.55	-1.55
11	0.5	0.5	0.5	0.5
12	**1**	-1.55	-0.55	-1.55

Index	왼쪽	오른쪽	위	아래
1	-2.495	**-1.495**	-1.55	**-0.55**
2	**-0.55**	-1.495	-1.55	**-0.55**
3	**-0.55**	**-0.55**	-1.55	-9.1
4	**-0.55**	-1.55	-1.55	**-0.55**
5	-1.55	**-0.55**	**-0.55**	**-0.55**
6	**-0.55**	-9.1	**-0.55**	-9.55
7	-0.55	-1.09	-0.55	**1**
8	-9.55	-1.55	-0.55	**-0.1**
9	-1.55	-9.55	**-0.55**	-1.55
10	-0.55	**1**	-0.55	-1.55
11	0.5	0.5	0.5	0.5
12	**1**	-1.55	-0.55	-1.55

Q-learning(예시)



$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

12, left : 1
12, up : -1 + 0.9 x (-0.1) = -1.09
8, down : -1 + 0.9 x (1) = - 0.1
7, right : -1 + 0.9 x (-0.1) = -1.09
3, down : -10 + 0.9 x (1) = -9.1
2, right : -1 + 0.9 x (-0.55) = - 1.495
1, right : -1 + 0.9 x (-0.55) = - 1.495
1, left : -2 + 0.9 x (-0.55) = - 2.495

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

Index	왼쪽	오른쪽	위	아래
1	-2.495	-1.495	-1.55	**-0.55**
2	**-0.55**	-1.495	-1.55	**-0.55**
3	**-0.55**	**-0.55**	-1.55	-9.1
4	**-0.55**	-1.55	-1.55	**-0.55**
5	-1.55	**-0.55**	**-0.55**	**-0.55**
6	**-0.55**	-9.1	**-0.55**	-9.55
7	-0.55	-1.09	-0.55	**1**
8	-9.55	-1.55	-0.55	**-0.1**
9	-1.55	-9.55	**-0.55**	-1.55
10	-0.55	**1**	-0.55	-1.55
11	0.5	0.5	0.5	0.5
12	**1**	-1.55	-0.55	-1.55

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

1 Start	2	3	4
5	6	7 장애물	8
9	10 장애물	11 Goal	12

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

이동을 했을 때, 주변에 장애물이나 Goal 이 없다면 -1.495
이동을 했을 때, 제자리이면 -2.495
이동을 했을 때, 장애물이면 -9.1
Goal 직전의 직전이면 -0.1
모두 업데이트!!

Index	왼쪽	오른쪽	위	아래
1	-2.495	-1.495	-1.55	**-0.55**
2	**-0.55**	-1.495	-1.55	**-0.55**
3	**-0.55**	**-0.55**	-1.55	-9.1
4	**-0.55**	-1.55	-1.55	**-0.55**
5	-1.55	**-0.55**	**-0.55**	**-0.55**
6	**-0.55**	-9.1	**-0.55**	-9.55
7	-0.55	-1.09	-0.55	**1**
8	-9.55	-1.55	-0.55	**-0.1**
9	-1.55	-9.55	**-0.55**	-1.55
10	-0.55	**1**	-0.55	-1.55
11	0.5	0.5	0.5	0.5
12	**1**	-1.55	-0.55	-1.55

Q-learning(예시)

1 Start	2	3	4
5	6	7 장애물	8
9	10 장애물	11 Goal	12

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

이동을 했을 때, 주변에 장애물이나 Goal 이 없다면 -1.495
이동을 했을 때, 제자리이면 -2.495
이동을 했을 때, 장애물이면 -9.1
Goal 직전의 직전이면 -0.1
모두 업데이트!!

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

Index	왼쪽	오른쪽	위	아래
1	-2.495	-1.495	-1.495	-1.495
2	-1.495	-1.495	-2.495	-1.495
3	-1.495	-1.495	-2.495	-9.1
4	-1.495	-2.495	-2.495	-1.495
5	-2.495	-1.495	-1.495	-1.495
6	-1.495	-9.1	-1.495	-9.55
7	-1.495	-1.09	-1.495	1
8	-9.55	-2.495	-1.495	-0.1
9	-2.495	-9.55	-1.495	-2.495
10	-1.495	1	-1.495	-2.495
11	0.5	0.5	0.5	0.5
12	1	-2.495	-1.09	-2.495

Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

1 Start	2	3	4
5	6	7 장애물	8
9	10 장애물	11 Goal	12

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

4, down : -1 + 0.9 x (-0.1) = -1.09
3, right : -1 + 0.9 x (-1.495) = -2.345
2, right : -1 + 0.9 x (-1.495) = -2.345
1, right : -1 + 0.9 x (-1.495) = -2.345

Index	왼쪽	오른쪽	위	아래
1	-2.495	-1.495	-1.495	-1.495
2	-1.495	-1.495	-2.495	-1.495
3	-1.495	-1.495	-2.495	-9.1
4	-1.495	-2.495	-2.495	-1.495
5	-2.495	-1.495	-1.495	-1.495
6	-1.495	-9.1	-1.495	-9.55
7	-1.495	-1.09	-1.495	1
8	-9.55	-2.495	-1.495	-0.1
9	-2.495	-9.55	-1.495	-2.495
10	-1.495	1	-1.495	-2.495
11	0.5	0.5	0.5	0.5
12	1	-2.495	-1.09	-2.495

Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

7, down : 1
3, down : -10 + 0.9 x (1) = -9.1
2, right : -1 + 0.9 x (-1.495) = -2.345
1, right : -1 + 0.9 x (-1.495) = -2.345
1, up : -2 + 0.9 x (1.495) = -3.345

Index	왼쪽	오른쪽	위	아래
1	-2.495	-2.345	-1.495	-1.495
2	-1.495	-2.345	-2.495	-1.495
3	-1.495	-2.345	-2.495	-9.1
4	-1.495	-2.495	-2.495	-1.09
5	-2.495	-1.495	-1.495	-1.495
6	-1.495	-9.1	-1.495	-9.55
7	-1.495	-1.09	-1.495	1
8	-9.55	-2.495	-1.495	-0.1
9	-2.495	-9.55	-1.495	-2.495
10	-1.495	1	-1.495	-2.495
11	0.5	0.5	0.5	0.5
12	1	-2.495	-1.09	-2.495

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \, \hat{Q}(s',a')$$

Q-value가 더 이상 바뀌지 않음(학습완료)

Goal로 가는 경로의 Q-value가 순차적으로 바뀜
(현재 12, 8, 7까지 학습 완료)

기본 이동과 제자리이동도 바뀜

Index	왼쪽	오른쪽	위	아래
1	-2.495	-2.345	-3.345	-1.495
2	-1.495	-2.345	-2.495	-1.495
3	-1.495	-2.345	-2.495	-9.1
4	-1.495	-2.495	-2.495	-1.09
5	-2.495	-1.495	-1.495	-1.495
6	-1.495	-9.1	-1.495	-9.55
7	-1.495	-1.09	-1.495	1
8	-9.55	-2.495	-1.495	-0.1
9	-2.495	-9.55	-1.495	-2.495
10	-1.495	1	-1.495	-2.495
11	0.5	0.5	0.5	0.5
12	1	-2.495	-1.09	-2.495

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

1 Start	2	3	4
5	6	7 장애물	8
9	10 장애물	11 Goal	12

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

8, up : -1 + 0.9 x (-1.09) = -1.981
4, down : -1 + 0.9 x (-0.1) = -1.09
3, right : -1 + 0.9 x (-1.09) = 1.981
2, right : -1 + 0.9 x (-2.345) = -3.11
1, right : -1 + 0.9 x (-2.345) = -3.11
1, left : -2 + 0.9 x (-2.345) = -4.11

Index	왼쪽	오른쪽	위	아래
1	-3.345	-2.345	-3.345	-2.345
2	-2.345	-2.345	-3.345	-2.345
3	-2.345	-2.345	-3.345	-9.1
4	-2.345	-3.345	-3.345	-1.09
5	-3.345	-2.345	-2.345	-2.345
6	-2.345	-9.1	-2.345	-9.1
7	-2.345	-1.09	-2.345	1
8	-9.55	-3.345	-2.345	-0.1
9	-3.345	-9.55	-2.345	-3.345
10	-2.345	1	-2.345	-3.345
11	0.5	0.5	0.5	0.5
12	1	-3.345	-1.09	-3.345

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

1 Start	2	3	4
5	6	7 장애물	8
9	10 장애물	11 Goal	12

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

8, up : -1 + 0.9 x (-1.09) = -1.981
4, down : -1 + 0.9 x (-0.1) = -1.09
3, right : -1 + 0.9 x (-1.09) = 1.981
2, right : -1 + 0.9 x (-1.981) = -2.78
1, right : -1 + 0.9 x (-3.11) = -3.8
1, left : -2 + 0.9 x (3.11) = -4.8

Index	왼쪽	오른쪽	위	아래
1	-4.11	-3.11	-4.11	-3.11
2	-3.11	-2.78	-4.11	-3.11
3	-3.11	-1.981	-4.11	-9.1
4	-3.11	-4.11	-4.11	-1.09
5	-4.11	-3.11	-3.11	-3.11
6	-3.11	-9.1	-3.11	-9.1
7	-3.11	-1.09	-3.11	1
8	-9.55	-4.11	-1.981	-0.1
9	-4.11	-9.55	-3.11	-4.11
10	-3.11	1	-3.11	-4.11
11	0.5	0.5	0.5	0.5
12	1	-4.11	-1.09	-4.11

2 Inflearn Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

1 Start	2	3	4
5	6	7 장애물	8
9	10 장애물	11 Goal	12

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

2, right : -1 + 0.9 x (-1.981) = -2.78
1, right : -1 + 0.9 x (-2.78) = -3.5
1, left : -2 + 0.9 x (3.8) = -5.42
1, down : -1 + 0.9 x (3.8) = -4.42

Index	왼쪽	오른쪽	위	아래
1	-4.8	-3.8	-4.8	-3.8
2	-3.8	-2.78	-4.8	-3.8
3	-3.8	-1.981	-4.8	-9.1
4	-3.8	-4.8	-4.8	-1.09
5	-4.8	-3.8	-3.8	-3.8
6	-3.8	-9.1	-3.8	-9.1
7	-3.8	-1.09	-3.8	1
8	-9.55	-4.8	-1.981	-0.1
9	-4.8	-9.55	-3.8	-4.8
10	-3.8	1	-3.8	-4.8
11	0.5	0.5	0.5	0.5
12	1	-4.8	-1.09	-4.8

Q-learning(예시)

reward

이동 -1 / 제자리 이동 -2
장애물 -10
Goal +1

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \, max_{a'} \hat{Q}(s',a')$$

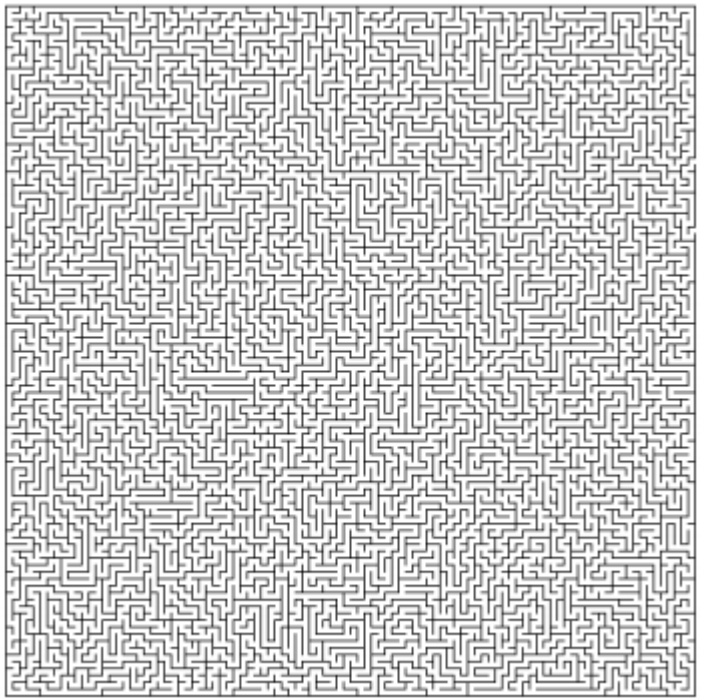
학습완료

1의 max(q-value)가 바뀌었으므로, (5, up)등 추가학습은 가능함

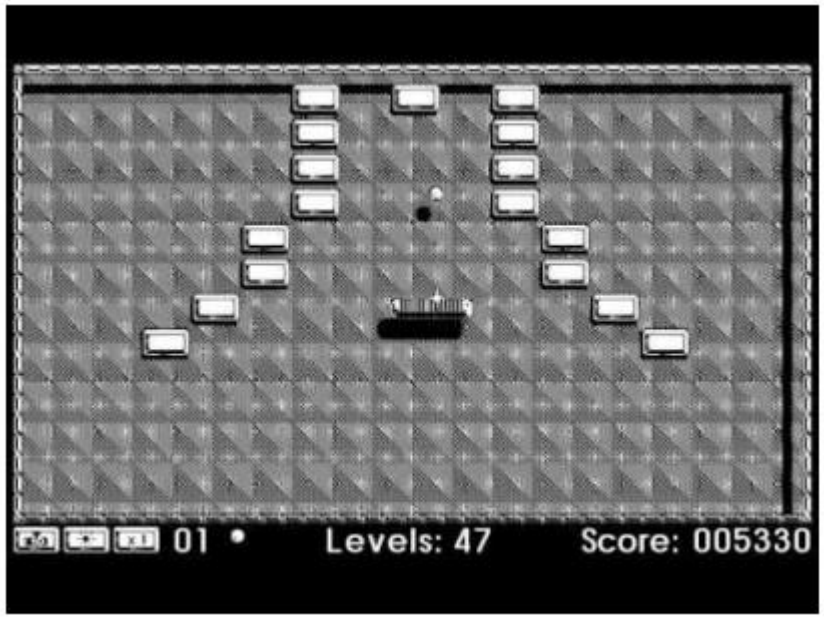
Index	왼쪽	오른쪽	위	아래
1	-5.42	-3.5	-5.42	-4.42
2	-4.42	-2.78	-5.42	-4.42
3	-4.42	-1.981	-5.42	-9.1
4	-4.42	-5.42	-5.42	-1.09
5	-5.42	-4.42	-4.42	-4.42
6	-4.42	-9.1	-4.42	-9.1
7	-4.42	-1.09	-4.42	1
8	-9.55	-4.11	-1.981	-0.1
9	-5.42	-9.55	-4.42	-5.42
10	-4.42	1	-4.42	-5.42
11	0.5	0.5	0.5	0.5
12	1	-5.42	-1.09	-5.42

Q-network

- Q-Table을 현실에 적용하기엔 무리가 있음



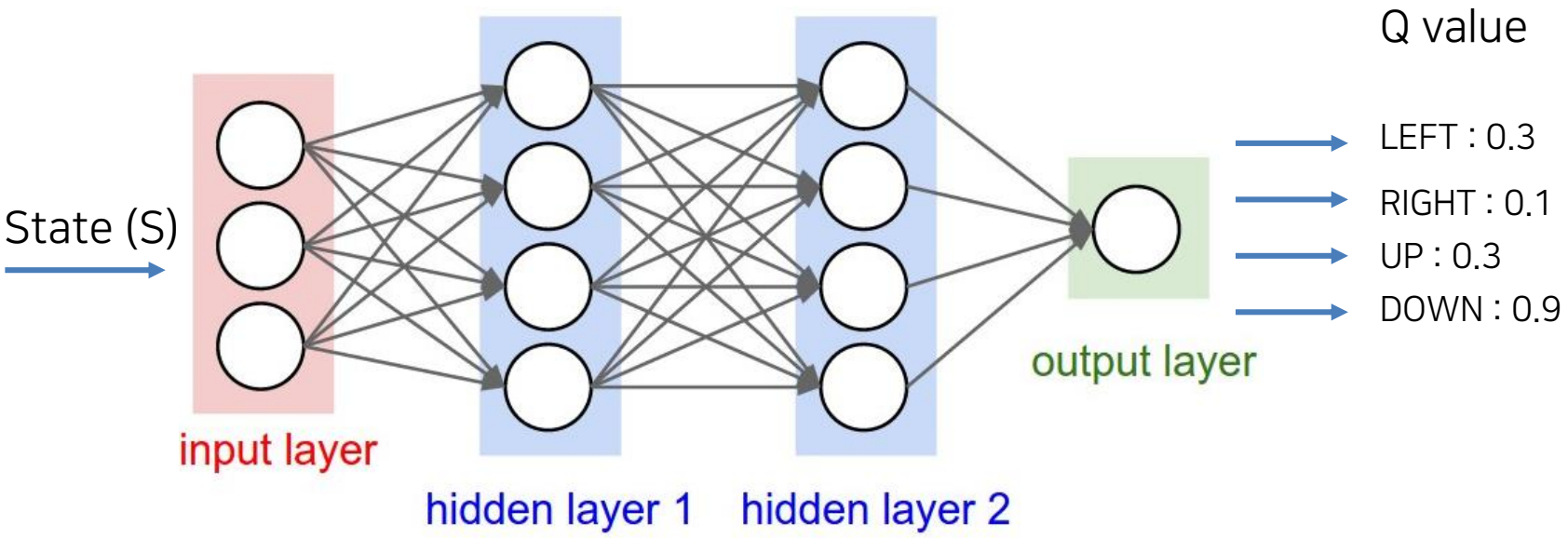
100 x 100 image



80 x 80 image x 2 color

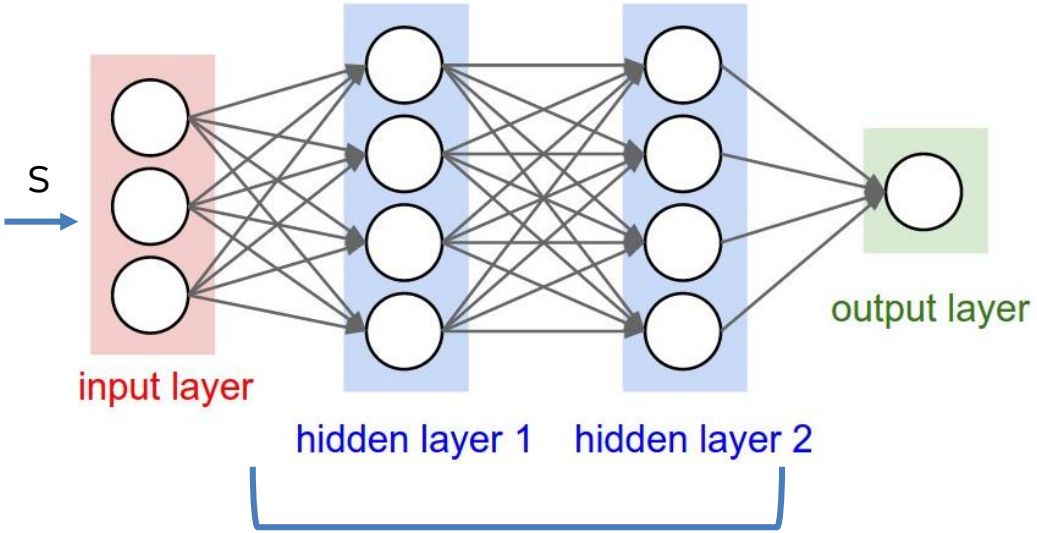
Q-network

- Neural network로 Q-function Approximation



Q-network

- Approximate $\boldsymbol{Q}^*$ function using $\boldsymbol{\theta}$ $\hat{\boldsymbol{Q}}(\boldsymbol{s},\boldsymbol{a}|\boldsymbol{\theta}) \sim \boldsymbol{Q}^*$(s,a)



$Ws\ (\ \hat{Q}(s,a|\theta)\ )$

- $cost(W) = (\ Ws\ - y)^2$
- $y = \underbrace{r + \gamma \max Q(s')}_{Q^*}$
- $min \sum_{t=0}^{T} [\hat{Q}(s_t, a_t|\theta) - (r_t + \gamma \max_{a'} \hat{Q}(s_{t+1}, a'|\theta))^2]$

3 Inflearn Q-network

- Algorithm

Initialize action-value function Q with random weights
For episode = 1, M **do**
Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequenced $\phi_1 = \phi(S_1)$
For t = 1, T **do**
With probability ϵ select a random action a_t
otherwise select $a_t = max_a\, Q^*(\phi(S_t), a; \theta)$
Execute action a_t in emulator and observe reward r_t and image x_{t+1}
Set $S_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$

Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma\, max_{a'}\, Q(\phi_{j+1}, a'; \theta)) & \text{for non-terminal } \phi_{j+1} \end{cases}$

Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$
end for
end for

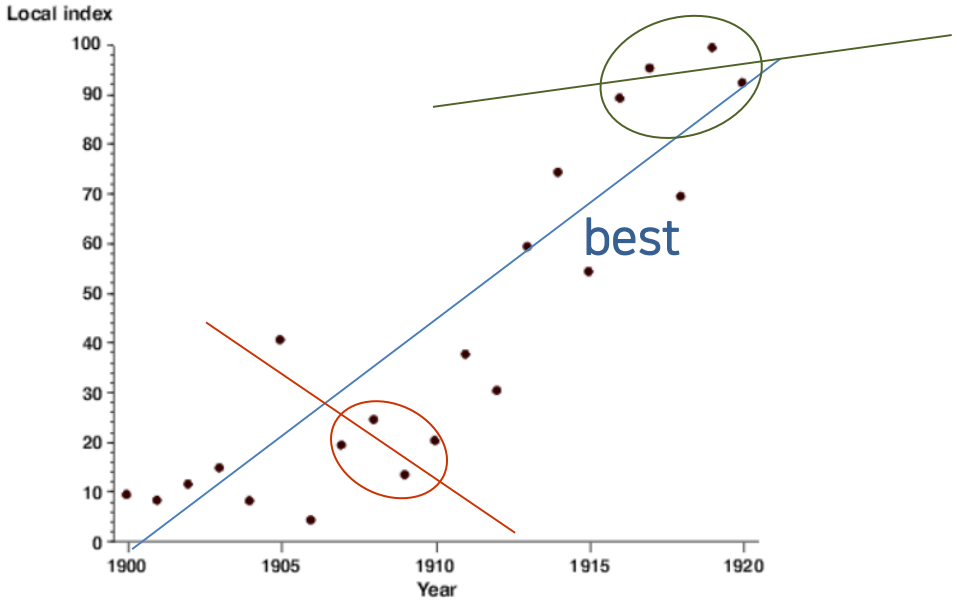
Q-network

- 단점
  - $min \sum_{t=0}^{T} [\hat{Q}(s_t, a_t|\theta) - (r_t + \gamma \max \hat{Q}(s_{t+1}, a'|\theta))^2]$
  - 수렴하지 않을 수 있음
    - Correlations between samples

      Non-stationary target

Q-network

- 단점
  - Correlations between samples
    (t-1 시점의 state와 t시점의 state는 유사할 것)



3 Inflearn Q-network

- 단점
  - Non-stationary target

$$\min_{\theta} \sum_{t=0}^{T} [\hat{Q}(s_t, a_t|\theta) - (r_t + \gamma \max \hat{Q}(s_{t+1}, a'|\theta))^2]$$

pred　　target

$$\hat{Y} = \hat{Q}(s_t, a_t|\theta) \qquad Y = r_t + \gamma \max_{a'} \hat{Q}(s_{t+1}, a'|\theta)$$

  - pred == target이 되도록 학습 하고자 하는데, 같은 θ를 업데이트하게 되면 pred값과 target값이 함께 이동함 (단순 regression을 생각하면 Y는 고정되어있음)
  - 화살을 쏘는 데 과녁도 움직임

3 Inflearn Q-network

- Q-network 단점에 대한 Solution

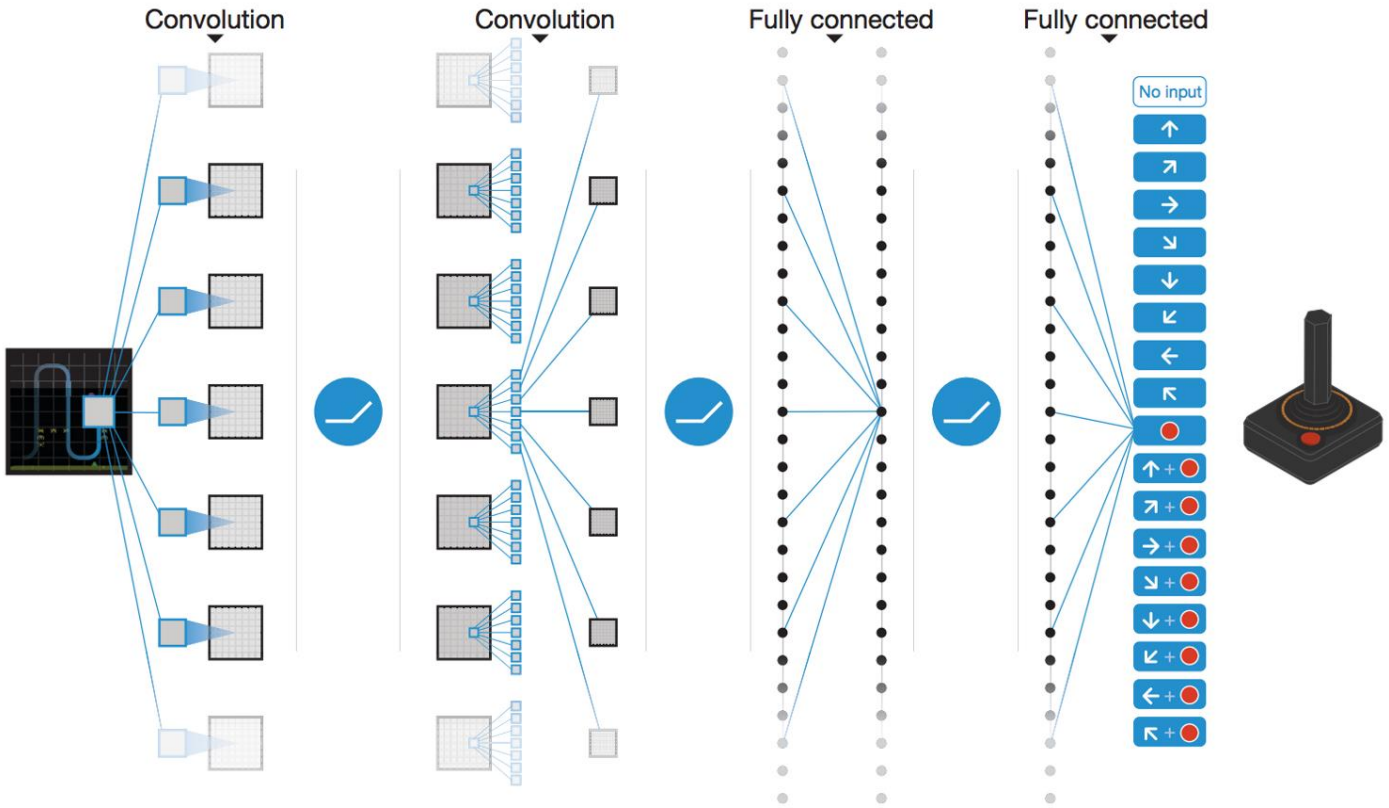
1. Go Deep

2. Capture and replay
   - correlation between samples

3. Separate networks : create a target network
   - Non-stationary targets

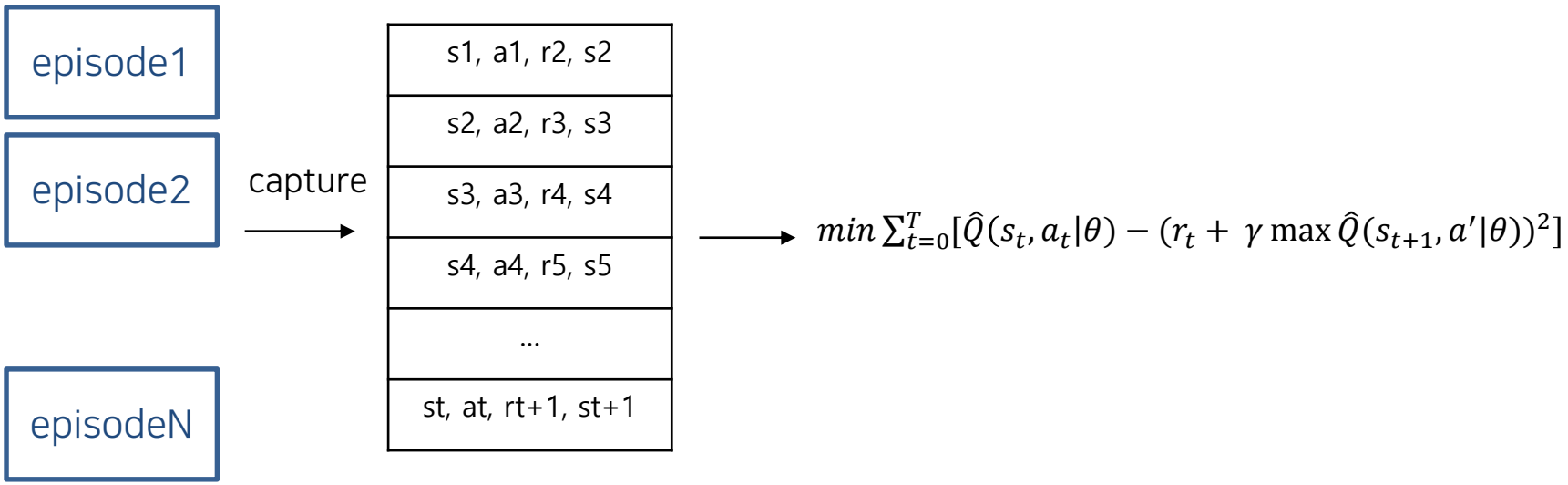
DQN (Deep Q-Network)

- Go deep
  - Deep 하게 학습시킴으로써 더 학습이 잘 되도록



DQN (Deep Q-Network)

- Capture and replay



기존 : Epsiode1(step1) -> backpropagation,
Epsiode1(step2) -> backpropagation ···

DQN : epsiode1 -> episode2 -> episodeN
-> random sample -> backpropagation ···

DQN (Deep Q-Network)

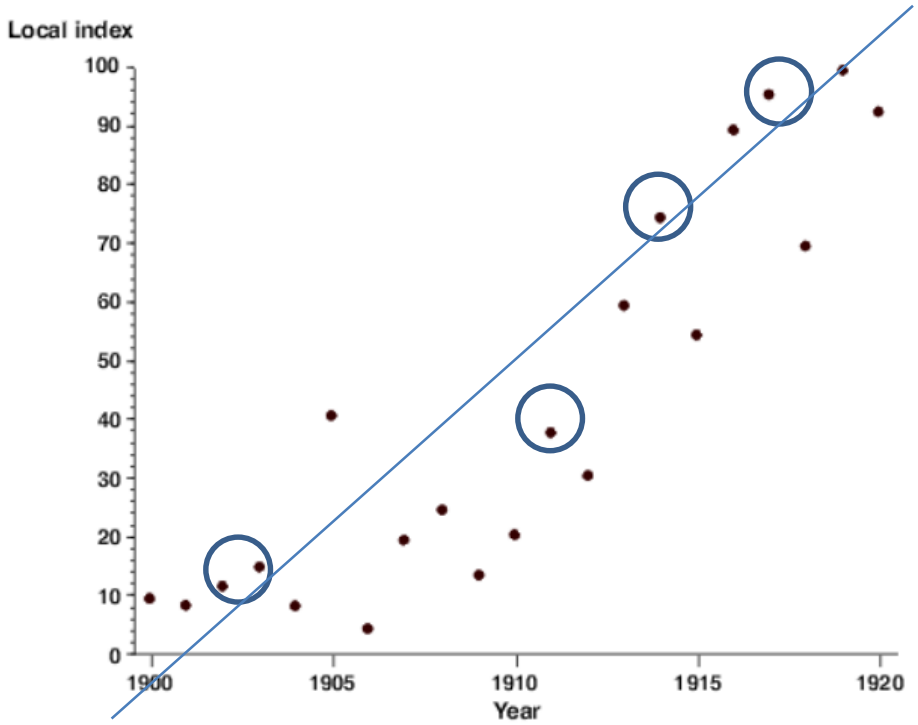
- Capture and replay

Initialize action-value function Q with random weights
For episode = 1, M **do**
Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequenced $\phi_1 = \phi(S_1)$
For t = 1, T **do**
With probability ϵ select a random action a_t
otherwise select $a_t = max_a\, Q^*(\phi(S_t), a; \theta)$
Execute action a_t in emulator and observe reward r_t and image x_{t+1}
Set $S_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
Store transition $(\boldsymbol{\phi_t, a_t, r_t, \phi_{t+1}})$ *in D*
Sample random minibatch of trainsitions $(\boldsymbol{\phi_t, a_t, r_t, \phi_{t+1}})$ *from D*

Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma\, max_{a'}\, Q(\phi_{j+1}, a'; \theta)) & \text{for non-terminal } \phi_{j+1} \end{cases}$
Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$
end for
end for

DQN (Deep Q-Network)

- Capture and replay



DQN (Deep Q-Network)

- Separate networks : create a target network

$$\min_{\theta} \sum_{t=0}^{T} [\hat{Q}(s_t, a_t|\theta) - (r_t + \gamma \max \hat{Q}(s_{t+1}, a'|\theta))^2]$$

$$\min_{\theta} \sum_{t=0}^{T} [\hat{Q}(s_t, a_t|\theta) - (r_t + \gamma \max \hat{Q}(s_{t+1}, a'|\bar{\theta}))^2]$$

pred target

$$\hat{Y} = \hat{Q}(s_t, a_t|\theta) \qquad Y = r_t + \gamma \max_{a'} \hat{Q}(s_{t+1}, a'|\bar{\theta})$$

Y를 고정시키고 q-network를 update 시킨 후 일정 step 이후, q-network를 target network로 복사

* Human-level control through deep reinforcement learning(2015)

DQN (Deep Q-Network)

- Separate networks : create a target network

Initialize action-value function Q with random weights
For episode = 1, M **do**
  Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequenced $\phi_1 = \phi(S_1)$
  For t = 1, T **do**
    With probability ϵ select a random action a_t
    otherwise select $a_t = max_a\, Q^*(\phi(S_t), a; \theta)$
    Execute action a_t in emulator and observe reward r_t and image x_{t+1}
    Set $S_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\boldsymbol{\phi_t, a_t, r_t, \phi_{t+1}})$ $\boldsymbol{in\ D}$
    Sample random minibatch of trainsitions $(\boldsymbol{\phi_t, a_t, r_t, \phi_{t+1}})$ $\boldsymbol{from\ D}$

    Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma\, max_{a'}\, \hat{Q}(\phi_{j+1}, a'; \theta)) & \text{for non-terminal } \phi_{j+1} \end{cases}$

    Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ **with respect to the network parameter $\boldsymbol{\theta}$**
    Every C steps reset $\hat{\boldsymbol{Q}} = \boldsymbol{Q}$
  end for
end for

DQN (Deep Q-Network)

- Separate networks : create a target network

Initialize action-value function Q with random weights
For episode = 1, M **do**
  Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequenced $\phi_1 = \phi(S_1)$
  For t = 1, T **do**
    With probability ϵ select a random action a_t
    otherwise select $a_t = max_a\, Q^*(\phi(S_t), a; \theta)$
    Execute action a_t in emulator and observe reward r_t and image x_{t+1}
    Set $S_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\boldsymbol{\phi_t, a_t, r_t, \phi_{t+1}})$ $\boldsymbol{in\ D}$
    Sample random minibatch of trainsitions $(\boldsymbol{\phi_t, a_t, r_t, \phi_{t+1}})$ $\boldsymbol{from\ D}$

    Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma\, max_{a'}\, \hat{Q}(\phi_{j+1}, a'; \theta)) & \text{for non-terminal } \phi_{j+1} \end{cases}$

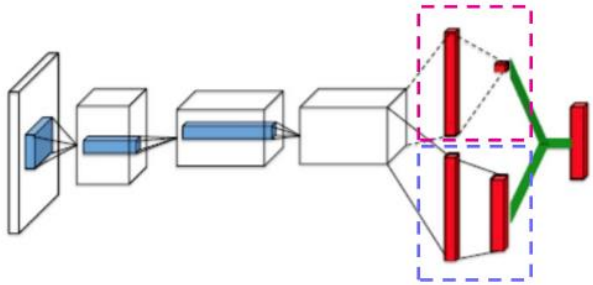
    Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ **with respect to the network parameter $\boldsymbol{\theta}$**
    Every C steps reset $\hat{\boldsymbol{Q}} = \boldsymbol{Q}$
  end for
end for

Dueling DQN

Dueling DQN

- State Value function을 도입

- 상황에 따라 학습을 집중 해야할 상태를 파악



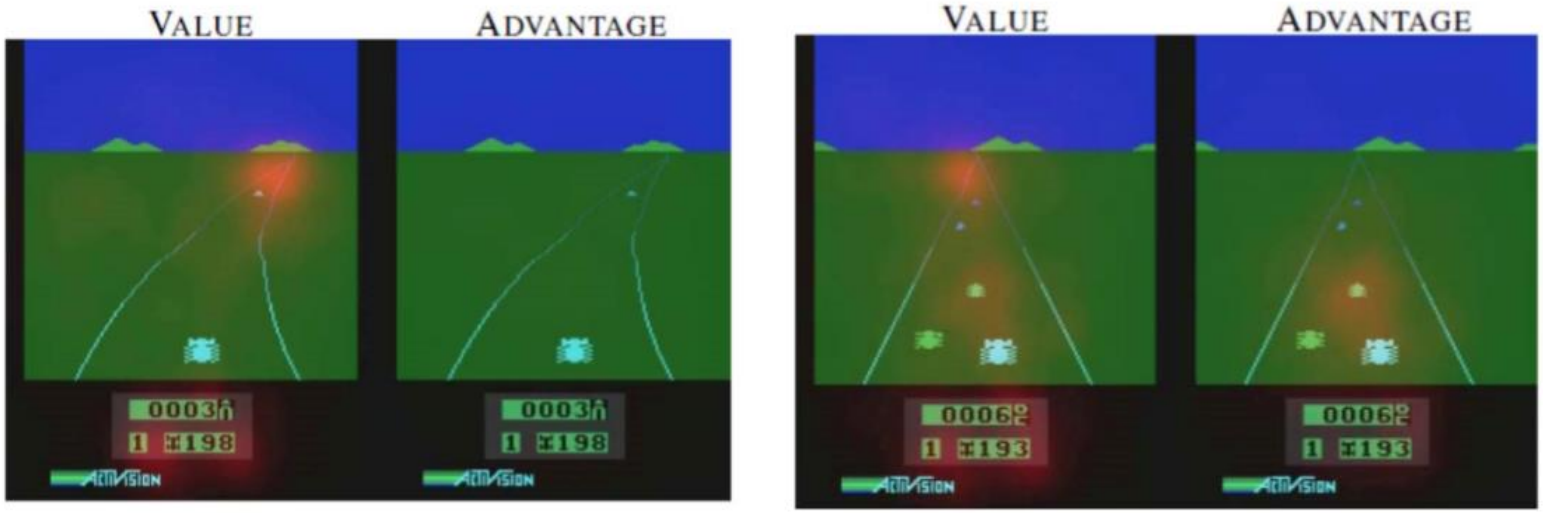
: state-value funcition(scalar), $V(s)$
state-dependent , action-independent
→ Goodness of state s

: advantage function(vector), $A(s,a)$
a state, action dependent
→ Goodness of taking action a in state s

$$Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + \left(A(s,a;\theta,\alpha) - \max_{a' \in |\mathcal{A}|} A(s,a';\theta,\alpha)\right)$$

Dueling DQN

Dueling DQN



PER DQN

Prioritized Experience Replay DQN (2016.02)

- Random하게 replay memory에서 뽑는게 아니라 학습에 중요한 episode에 가중치를 두어서 샘플링 하는 것

- 학습하기에 얼마나 가치가 있는지 평가할 지표

$$\text{TD-error } \delta_j = R_j + \gamma_j Q_{\text{target}}\left(S_j, \arg\max_a Q(S_j, a)\right) - Q(S_{j-1}, A_{j-1})$$

- Reward + 다음 State의 Qvalue – 현재 state의 Qvalue
  보상을 많이 받고, 다음 state와 현재 state의 value가 크면 클수록 학습할 가치가 크다.

NoiseNets

Noise Nets for Exploration (2017.06)

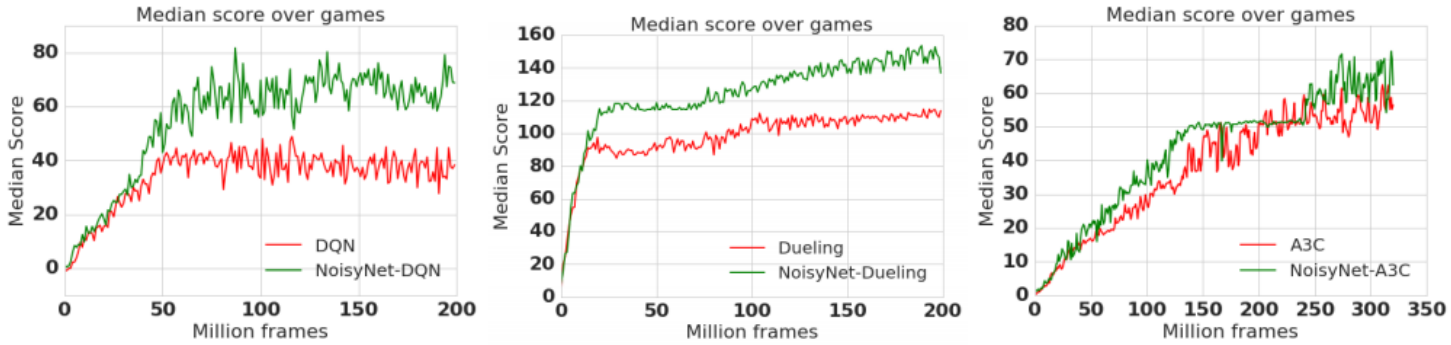
- Standard linear

$$y = wx + b$$

- Noise linear

$$y \overset{\text{def}}{=} (\mu^w + \sigma^w \odot \varepsilon^w)x + \mu^b + \sigma^b \odot \varepsilon^b$$

Learnable parameter Random variable



Distributional DQN

Distributional DQN (2017.07)

$$Q^{\pi}(s,a) = E[R_t]$$

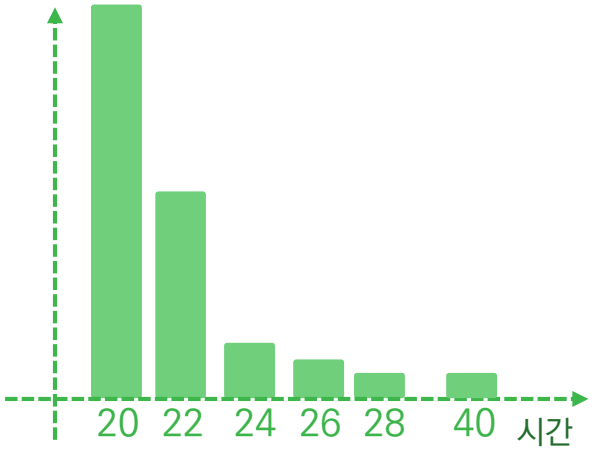
Q learning은 Q_함수를 미래의 총 보상의 기댓값으로 근사시키는 것.

[π]: policy, [s]: state, [a]: action, $R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots$

기대값은 분포의 특성을 나타 낼 수 있는 좋은 추정치

Distributional DQN

Distributional DQN (2017.07)



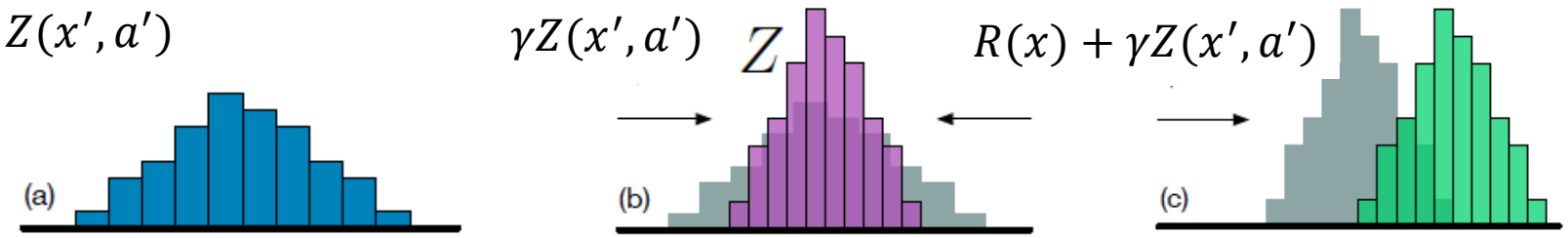
- 출퇴근 시간 지하철 연착/사고 발생 등을 고려하면 기대 값은 25분
- 사고가 발생한다면 버스를 이용하는 것이 합리적
- 기대 값은 합리적으로 보일 수 있지만 분포를 고려하는 것이 더 합리적

Distributional DQN

Distributional DQN (2017.07)

$$Q(s_t, a_i) = r(a_i) + \gamma \max_i Q(s_{t+1}, a_{i+1})$$

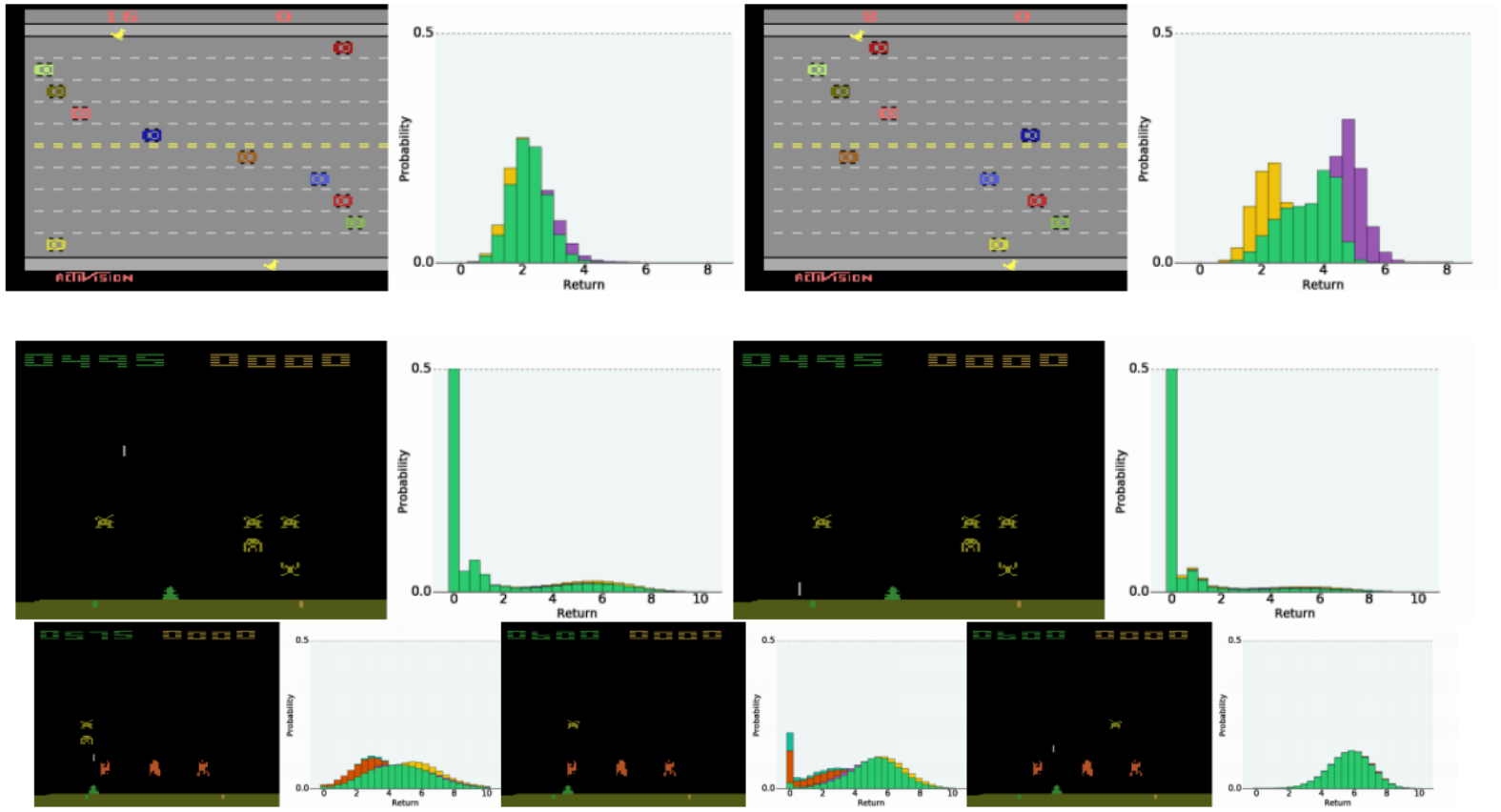
$$Z(x, a) = R(x) + \gamma Z(x', a') \quad \text{Z는 value distribution}$$



1. 다음 분포를 (a)그림처럼 생겼다고 가정.
2. Discount factor가 곱해지면, 분포는 중심으로 축소될 것.
3. Reward를 더한다면, 분포는 이동!

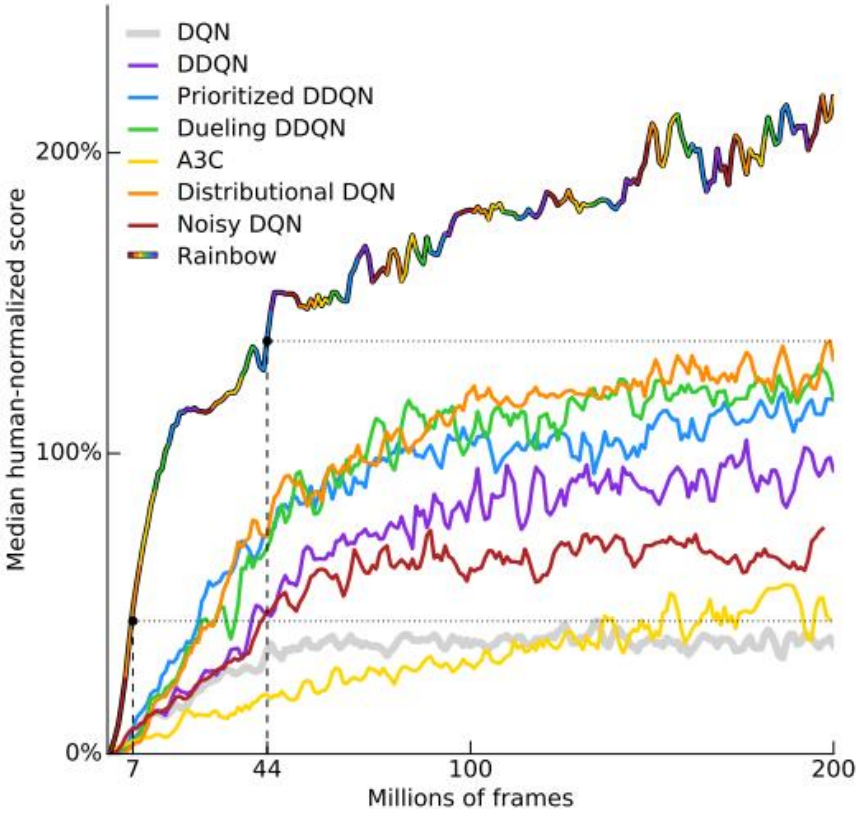
Distributional DQN

Distributional DQN (2017.07)



Rainbow DQN

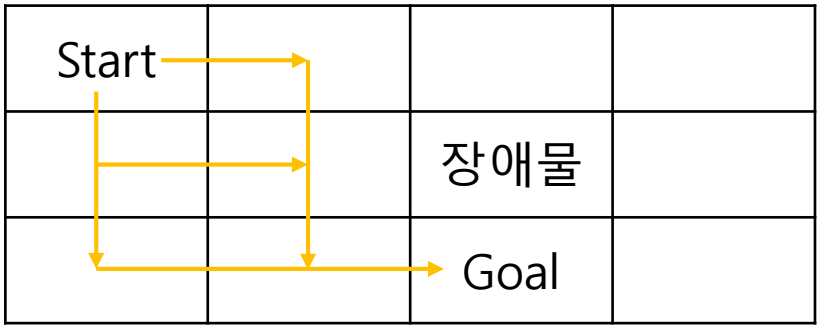
Rainbow DQN (2017.10)



Policy based RL

Policy based RL

- Value-based RL은 Q라는 action value function에 초점을 맞추고 policy를 구하는 방식
- Policy-based RL은 Policy 자체를 approximate (확률에 따라 action을 선택)



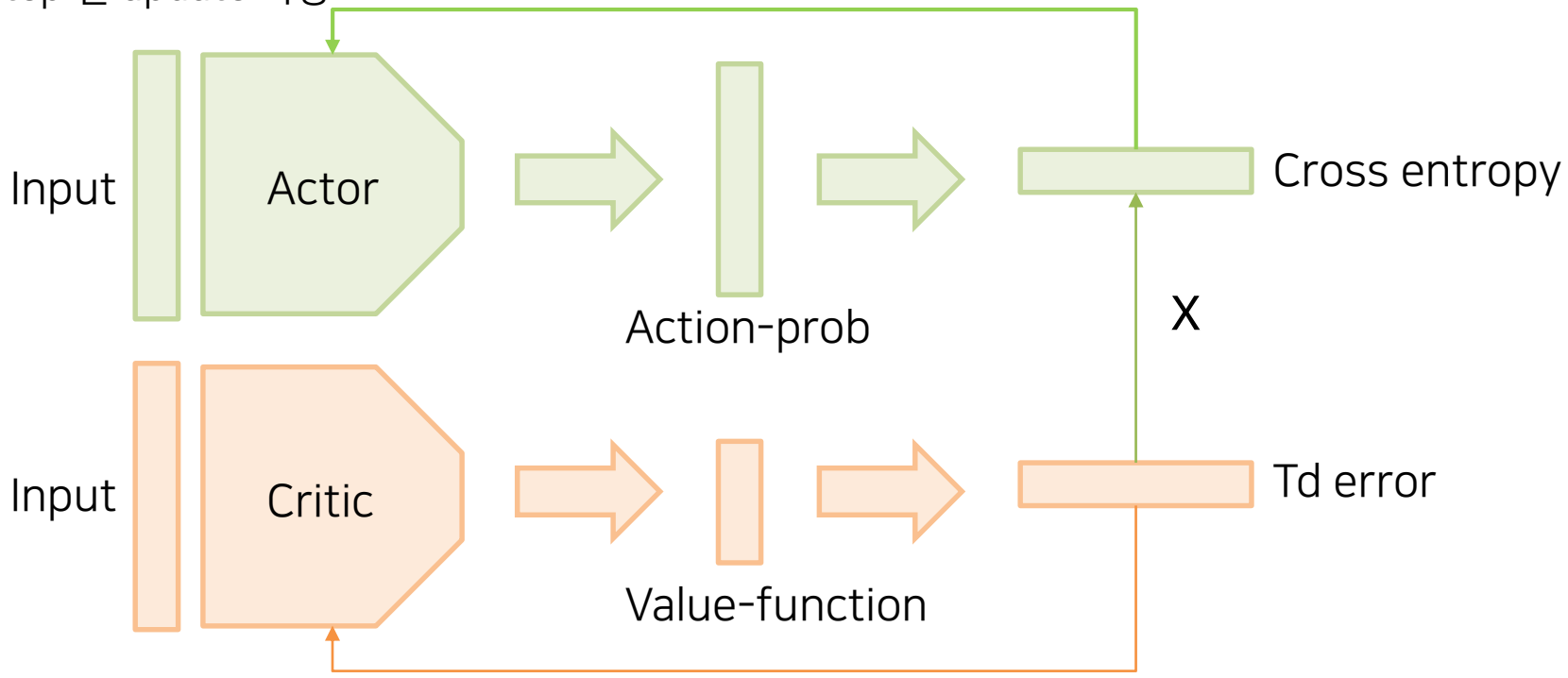
- Start 지점부터 Goal 지점까지 가는 방법(Policy)은 여러가지
  Value based RL은 한가지 방법만 제시 / Policy based RL은 여러 방법을 제시

Actor-Critic Network

Actor-Critic Network

- 정책을 근사하는 Actor Network + 가치 함수를 근사하는 Critic Network

- Step 별 update 가능



Self-imitation learning

$$\mathcal{L}^{sil} = \mathbb{E}_{s,a,R\in\mathcal{D}}\left[\mathcal{L}^{sil}_{policy} + \beta^{sil}\mathcal{L}^{sil}_{value}\right] \quad (1)$$

$$\mathcal{L}^{sil}_{policy} = -\log \pi_\theta(a|s)\left(R - V_\theta(s)\right)_+ \quad (2)$$

$$\mathcal{L}^{sil}_{value} = \frac{1}{2}\left\|\left(R - V_\theta(s)\right)_+\right\|^2 \quad (3)$$

기본적인 AC loss와 같으나 loss안에 operater $(\cdot)_+ = \max(\cdot, 0)$ 가 껴있다라는 점만 다름.

$(R > V_\theta)$ 인 경우에만, network parameter update

만약, 과거의 R값이 현재의 value 보다 크다면, agent로 하여금 과거의 state에서 선택했던 action을 선택하도록 학습시키는 것. 반면에 R값이 현재 value보다 작거나 같다면, network의 parameter를 update 시키지 않음. 기대했던 것보다 더 큰 보상을 return해주는 과거의 decision을 agent가 모방하도록 하는 것.

Self-imitation learning

Hard Exploration Atari Games

Atari game에서 가장 sparse 한 reward 환경인 6개의 게임에 대해 실험을 진행

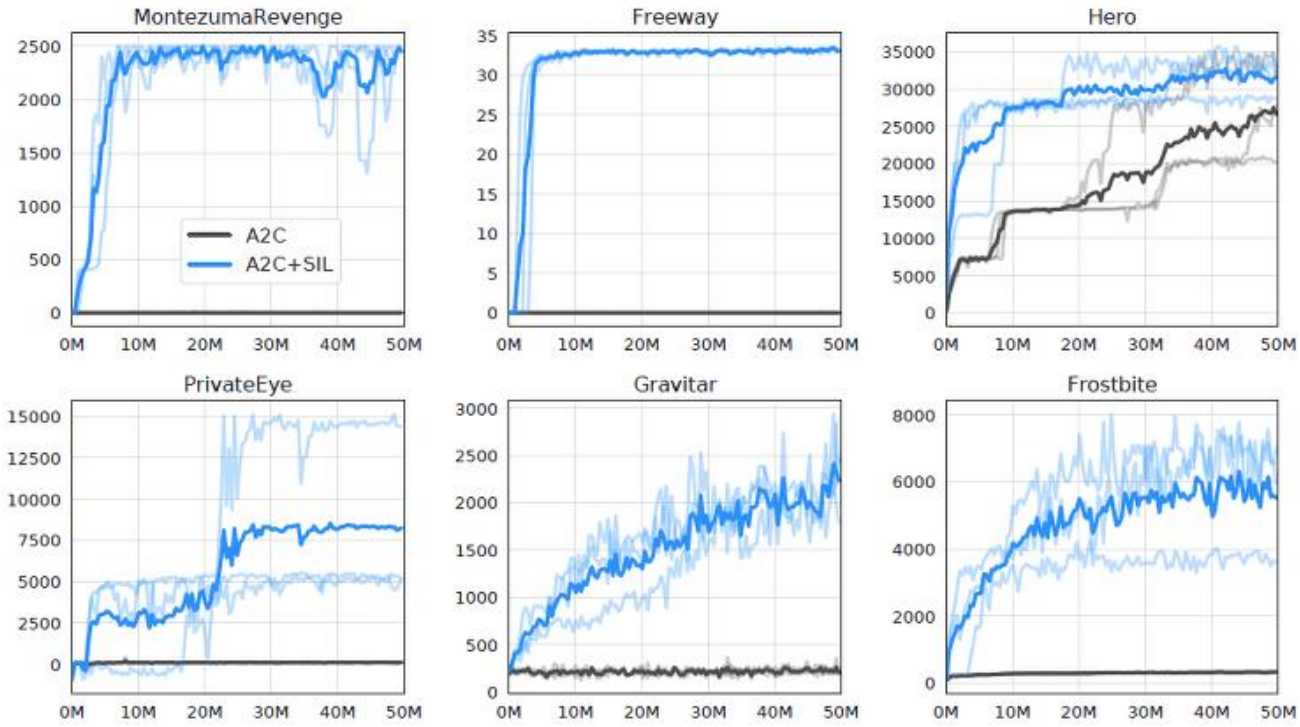


Figure 3. Learning curves on hard exploration Atari games. X-axis and y-axis represent steps and average reward respectively.

10 Inflearn Sparse Reward Problem

Sparse Reward Problem

Exploration을 잘하자 !

-> Exploration bonus

1. Count based exploration bonus

$$i_t = 1/n_t(\boldsymbol{s}) \text{ and } i_t = 1/\sqrt{n_t(\boldsymbol{s})}$$

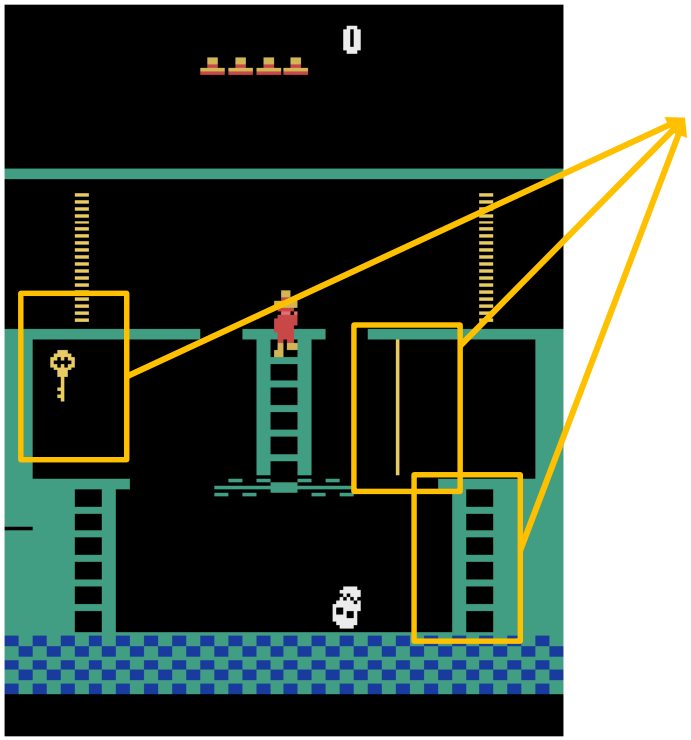
Tabular-setting : Table 활용
Nontabular-setting : state의 density를 측정하는 방법

2. Curiosity (Prediction error)

10 Sparse Reward Problem

Hierarchical Deep Reinforcement Learning

- 본 논문은 sparse reward에 대한 해결책으로 계층적으로 학습하는 강화학습 구조를 제안함.

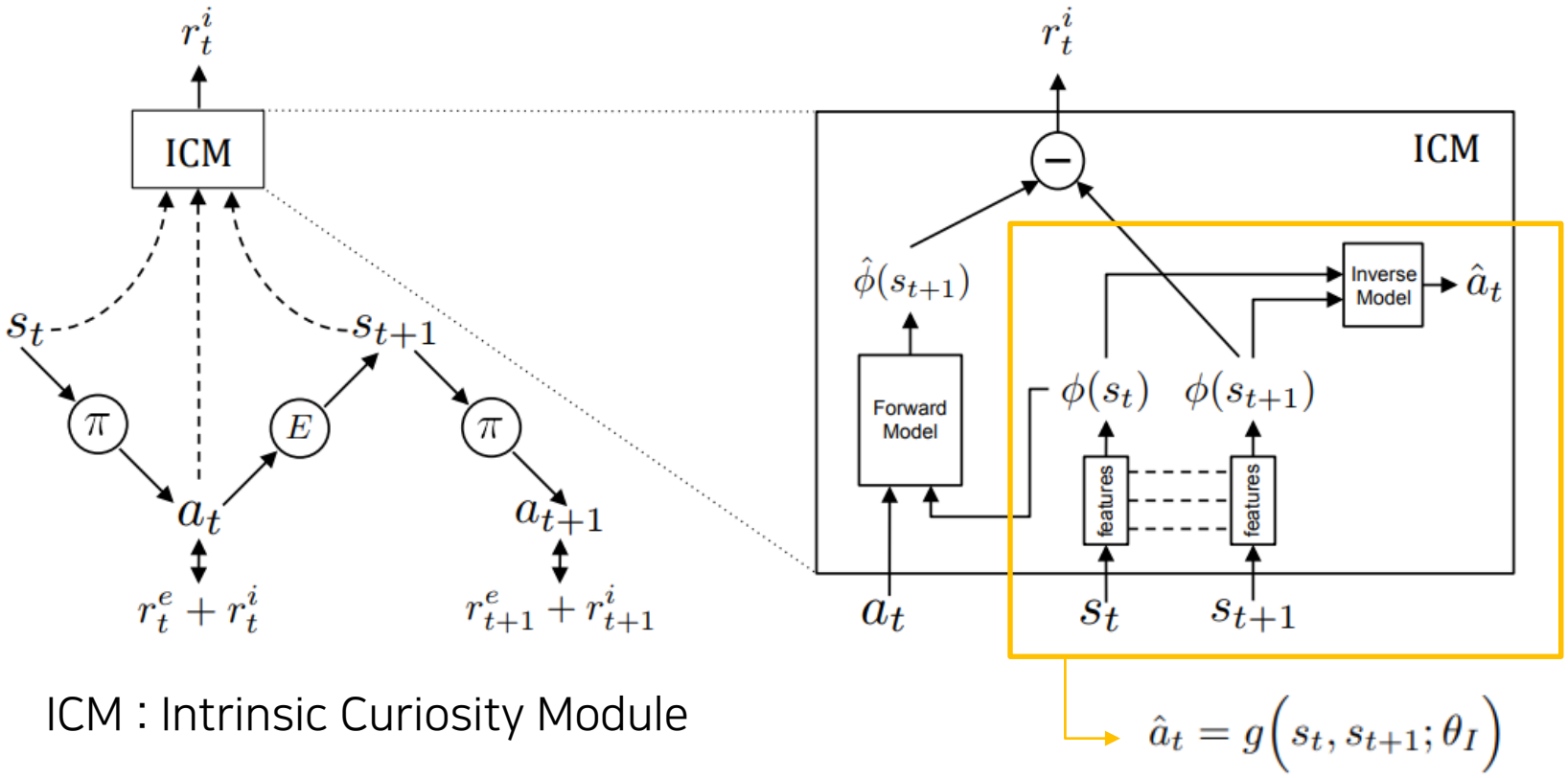


(c) Atari - Montezuma's

- 밧줄타기, 사다리타기, Key획득하기 등 여러가지의 Goal을 설정하여 학습
- HDQN (Hierarchical Deep Reinforcement Learning)은 Goal을 관리하는 meta-controller와, 각 goal을 학습시키는 controller로 구성되어 있음.
- Controller : 각 Goal 내에서 reward가 최대가 되도록 학습(Goal을 달성 할 수 있도록) 하는 DQN 모형
- Meta-controller : Episode의 총 reward가 커지도록 (게임을 clear할 수 있도록) 하는 goal을 찾아내는 DQN 모형

10 Sparse Reward Problem

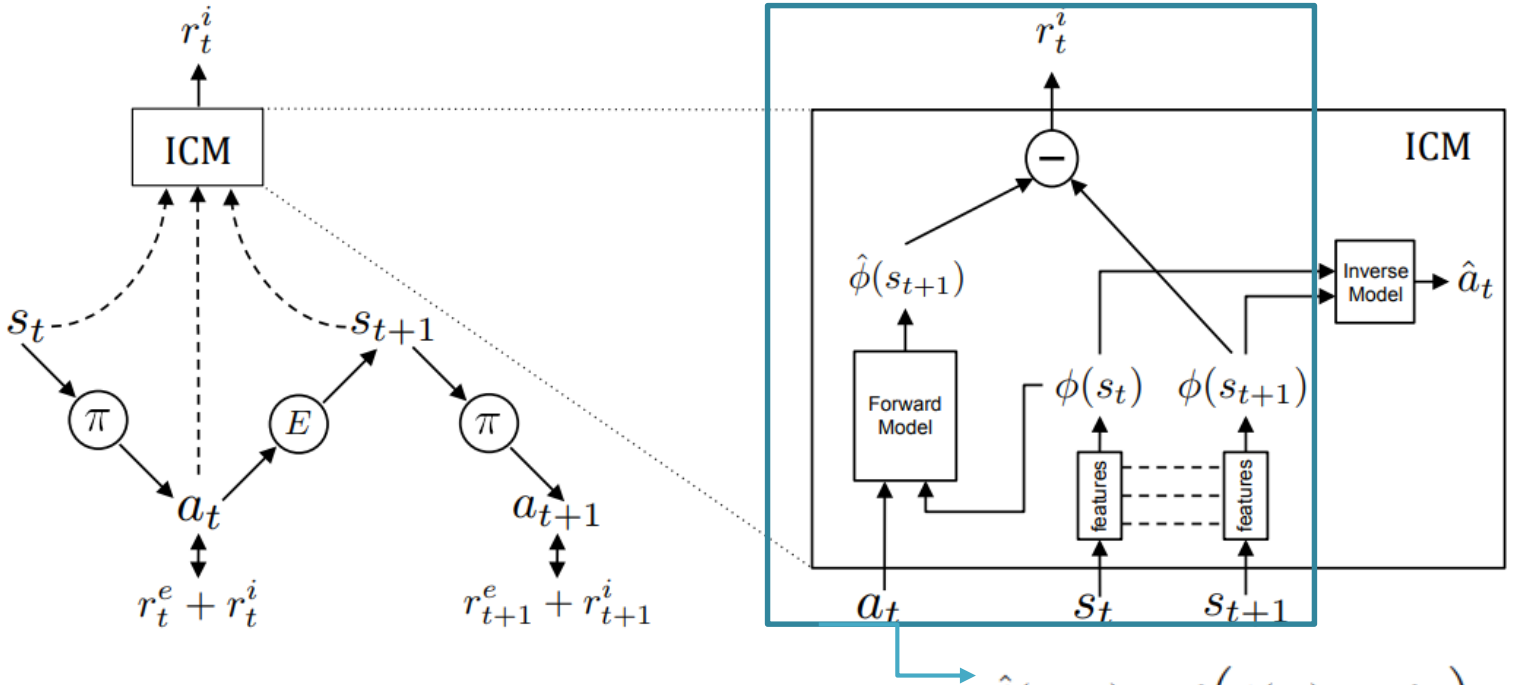
Curiosity-driven Exploration by Self-supervised Prediction



ICM : Intrinsic Curiosity Module

10 Sparse Reward Problem

Curiosity-driven Exploration by Self-supervised Prediction



State와 action을 통해 다음 State 예측
Intrinsic reward : 실제 다음 State – 예측 다음 state

$$\hat{\phi}(s_{t+1}) = f\Big(\phi(s_t), a_t; \theta_F\Big)$$

$$L_F\Big(\phi(s_t), \hat{\phi}(s_{t+1})\Big) = \frac{1}{2}\|\hat{\phi}(s_{t+1}) - \phi(s_{t+1})\|_2^2 \qquad r_t^i = \frac{\eta}{2}\|\hat{\phi}(s_{t+1}) - \phi(s_{t+1})\|_2^2$$

$\eta = scaling\ factor$

10 Sparse Reward Problem

Curiosity-driven Exploration by Self-supervised Prediction

State와 Next State로 Action을 예측

논문에 나와있는 구절

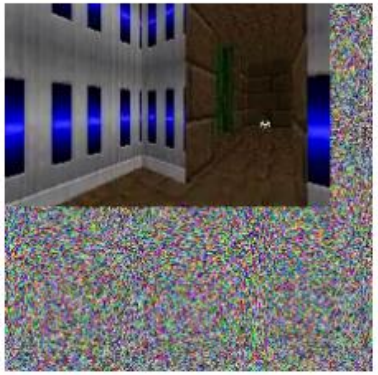
"The use of inverse models has been investigated to learn features for recognition tasks"

바람에 흔들리는 나뭇잎이 state에 있다고 가정해보자. State와 다음 State에서는 나뭇잎의 위치가 매우 다를 것. 그런데 state와 다음 state를 input으로 하여 action을 '잘' 예측하게 한다면, state와 다음 state의 feature가 나뭇잎에 대하여 robust할 것임!. 즉, 앞서 언급한 Prediction error가 curiosity의 좋은 척도가 되기 위한 올바른 feature space를 만들기 위한 model인 것 !

10 Inflearn Sparse Reward Problem

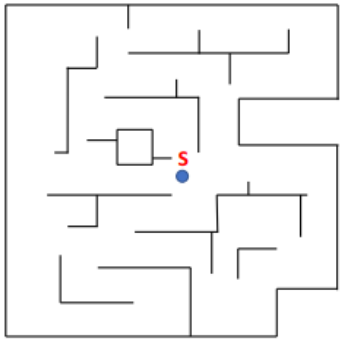
Curiosity-driven Exploration by Self-supervised Prediction

(a) Input snapshot in VizDoom

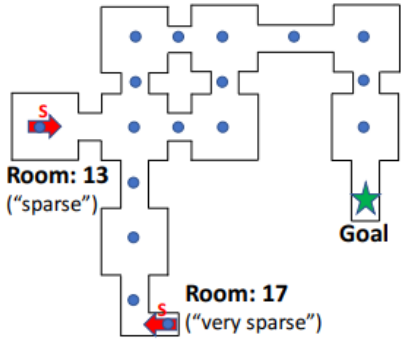


(b) Input w/ noise

Figure 3. Frames from VizDoom 3-D environment which agent takes as input: (a) Usual 3-D navigation setup; (b) Setup when uncontrollable noise is added to the input.



(a) Train Map Scenario



(b) Test Map Scenario

Figure 4. Maps for VizDoom 3-D environment: (a) For generalization experiments (c.f. Section 4.3), map of the environment where agent is pre-trained only using curiosity signal without any reward from environment. 'S' denotes the starting position. (b) Testing map for VizDoom experiments. Green star denotes goal location. Blue dots refer to 17 agent spawning locations in the map in the "dense" case. Rooms 13, 17 are the fixed start locations of agent in "sparse" and "very sparse" reward cases respectively. Note that textures are also different in train and test maps.

10 Sparse Reward Problem

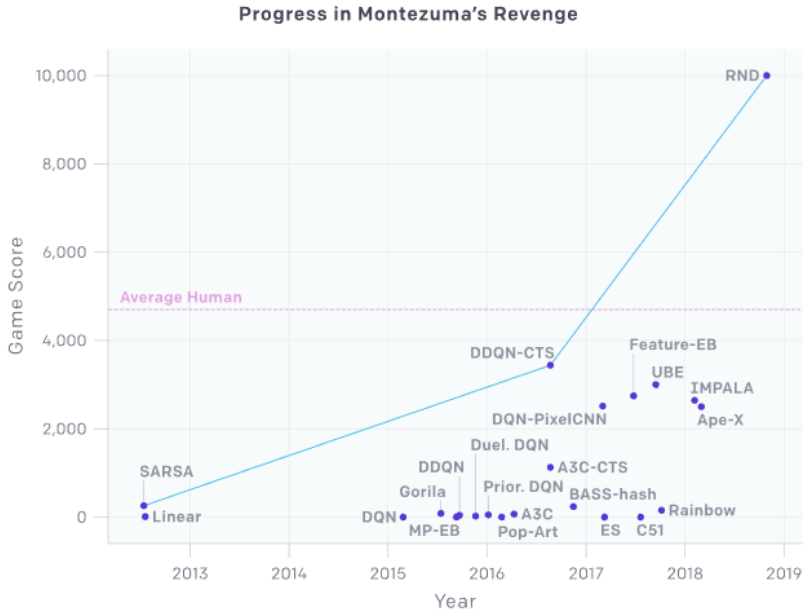
Exploration by Random Network Distillation

- Curiosity (prediction error)의 문제점 – ‘noisy-TV problem’

10 Inflearn Sparse Reward Problem

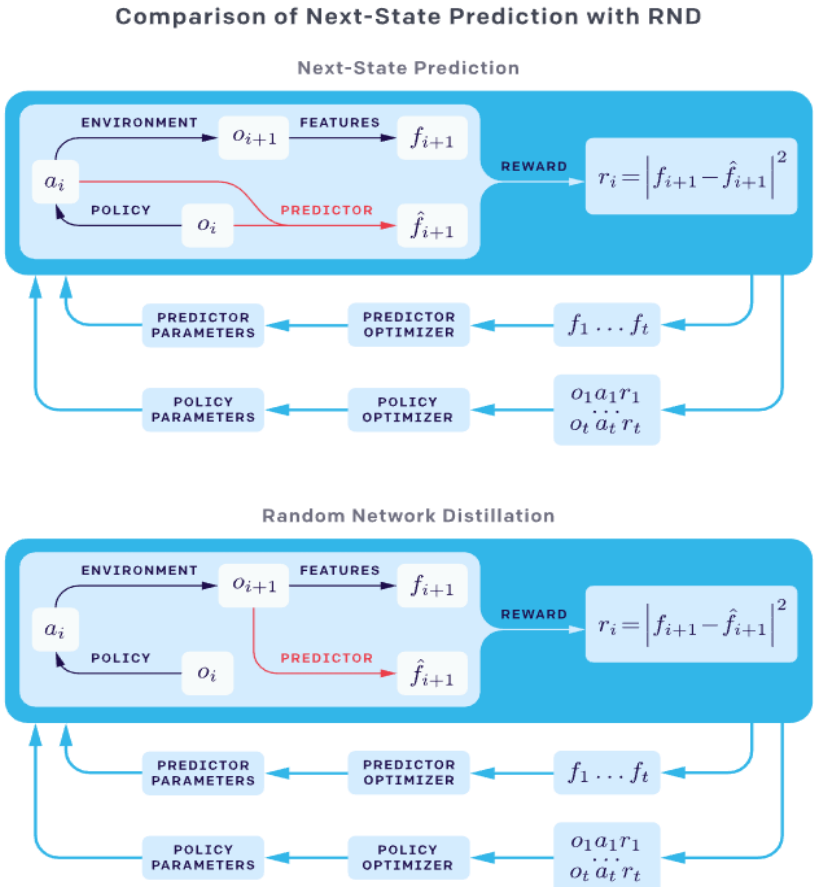
Exploration by Random Network Distillation

- Exploration bonus를 도입, Montezuma 게임에서 최초로 인간의 성능을 뛰어넘음
- Bonus : Fixed randomly initialized NN의 state feature를 예측한 오차
- Extrinsic reward없이도 Montezuma에서 절반이상 클리어



10 Sparse Reward Problem

Exploration by Random Network Distillation



1. A fixed and randomly initialized target network which sets the prediction problem

2. A predictor network trained on data collected by the agent

The target network takes an observation to an embedding

$$f : \mathcal{O} \to \mathbb{R}^k$$

Predictor neural network $\hat{f} : \mathcal{O} \to \mathbb{R}^k$ is trained by gradient descent to minimize the expected MSE $\|\hat{f}(\mathrm{x};\theta) - f(\mathrm{x})\|^2$

This process distills a randomly initialized neural network into a trained one

The prediction error is expected to be higher for novel states dissimilar to the ones the predictor has been trained on

'다음 state의 feature space가 predictive network가 잘 예측하지 못한 state라면 더 많은 reward를 줌 '

10 Inflearn Sparse Reward Problem

Exploration by Random Network Distillation

- Pure exploration
  - Intrinsic reward만 가지고 실험

동일 state를 만나면서 return이 감소

새로운 state를 만나면서 return이 올라감

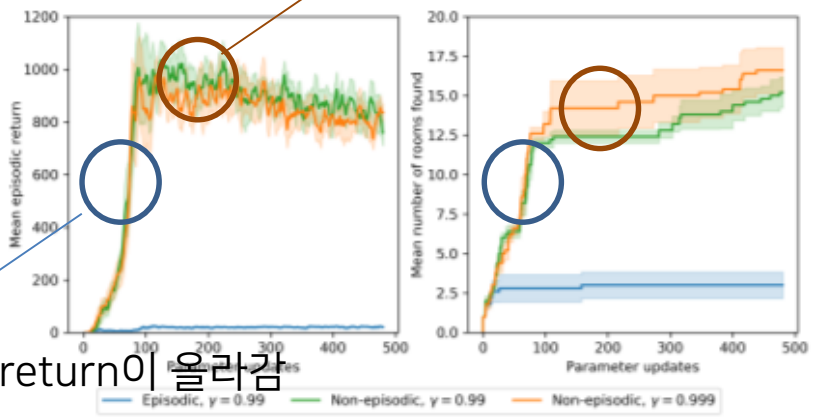


Figure 3: Mean episodic return and number of rooms found by pure exploration agents on Montezuma's Revenge trained without access to the extrinsic reward. The agents explores more in the non-episodic setting (see also Section 2.3)

감사합니다.