딥러닝특론

12강. 강화학습

정보과학과 정재화

학습목차

12강. 강화학습

1. 강화학습
2. Markov Decision Process
3. Q-Learning과 DQN

12강. 강화학습

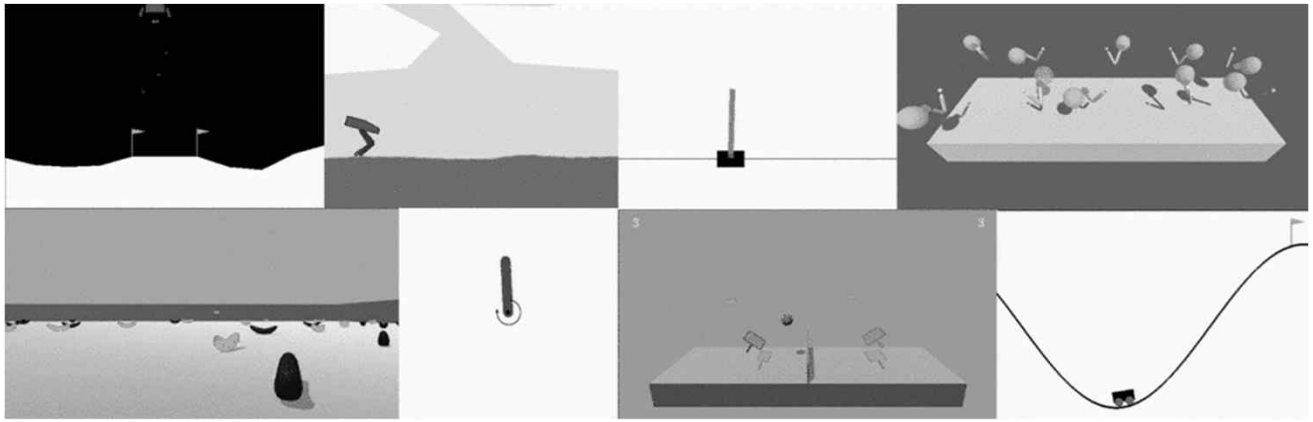
Deep Learning

1. 강화학습

강화학습
강화학습의 활용 분야
용어 및 문제정의

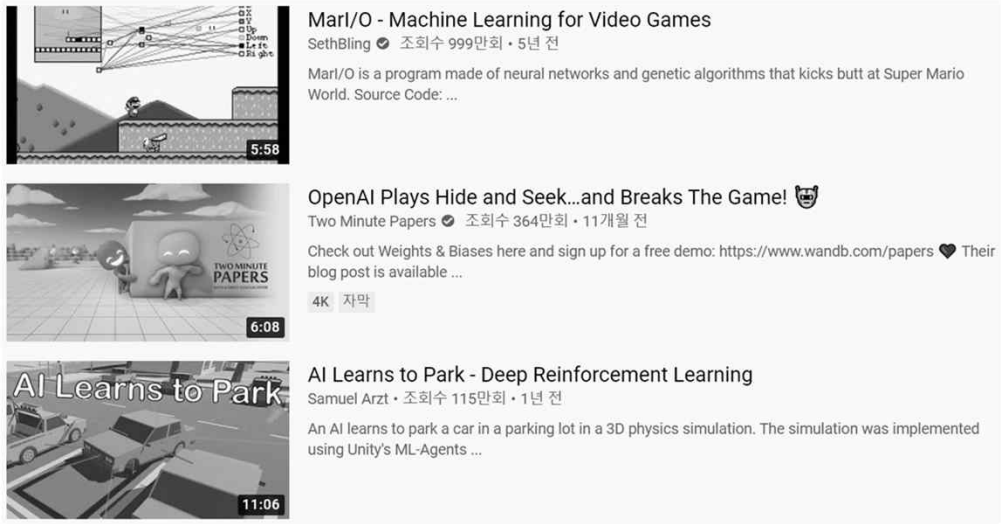
1 기계학습의 하위분야

- Supervised Learning: 입력 x에 대해 label y 학습
  - Ex) Classification
- Unsupervised Learning: label 없이 x의 성질을 학습
  - Ex) Clustering, AutoEncoder
- Reinforcement Learning (강화학습):
  주어진 환경에서 Agent의 보상을 최대화



1 강화학습의 특징

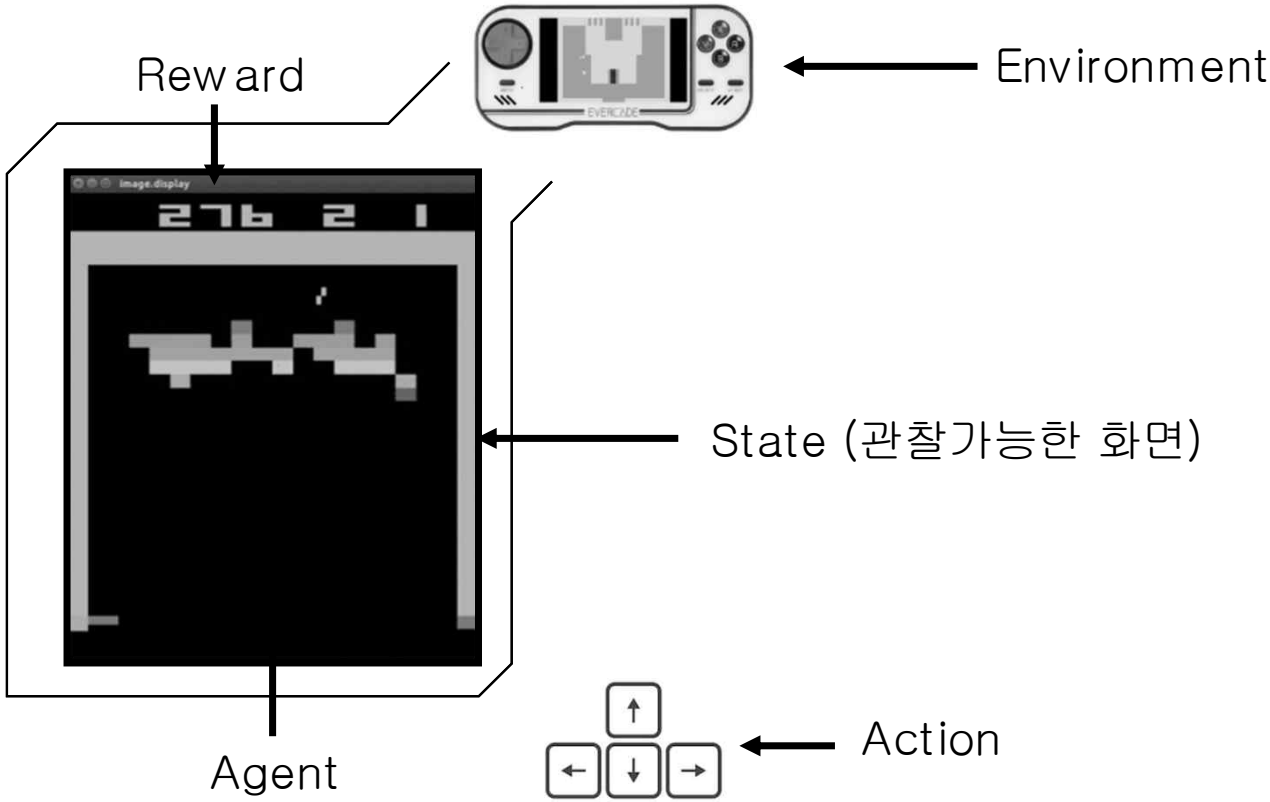
- 주어진 환경에서 Player (Agent)가 특정 임무를 수행
- 기존 방법으로는 학습하기 어려움
  - Supervised 라면 label은?
  - Unsupervised라면 loss function은?



1 용어 및 문제 정의

- Environment: 게임 환경
  - 다양한 객체 간 상호작용이 일어나는 공간
  - Agent도 객체 중 하나
- Agent: 플레이어
  - Environment 상에서 Action을 수행할 수 있는 존재
  - 특정 임무를 수행 => Reward로 모델링
- State: 상태: (화면)
  - Agent가 관찰가능한 Environment의 상태
- Reward: 점수
  - Agent의 Action으로 인해 발생한 상호작용으로 Agent가 얻게 되는 보상

1 용어 및 문제 정의



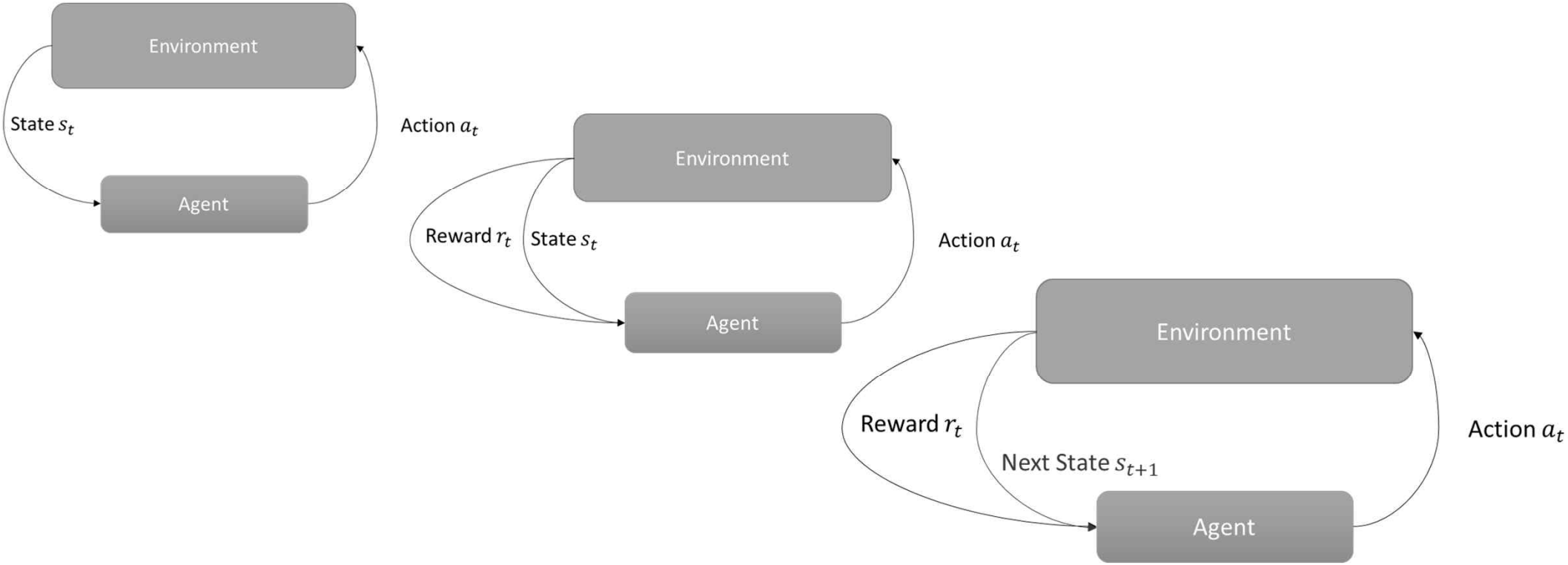
12강. 강화학습 *Deep Learning*

2. Markov Decision Process

Markov Decision Process
Value Function와 Q-value Function
Bellman Equation

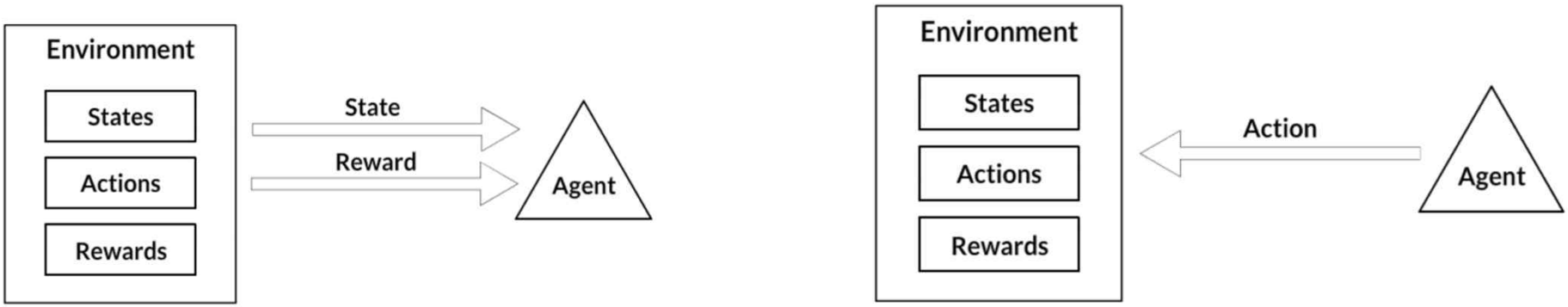
2 Markov Decision Process

- Environment, Agent, Reward의 관계를 도식화



2 Markov Decision Process

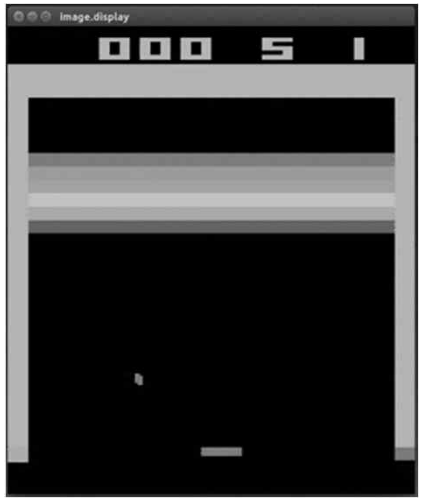
- Environment, Agent, Reward의 관계를 도식화



2 Markov Decision Process

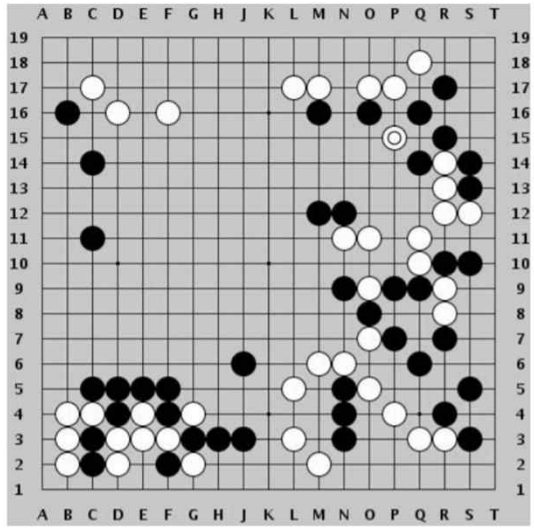
- Environment, Agent, Reward의 관계를 도식화
- Atari Game
  - 목표: 모든 블록 파괴
  - State: 블록위치, 공 위치, 공 속도
  - Actions: Left, Right, no/op
  - Reward: 블록 하나 파괴 시

→ Reward 최대화 ~ 목표 달성



2 Markov Decision Process

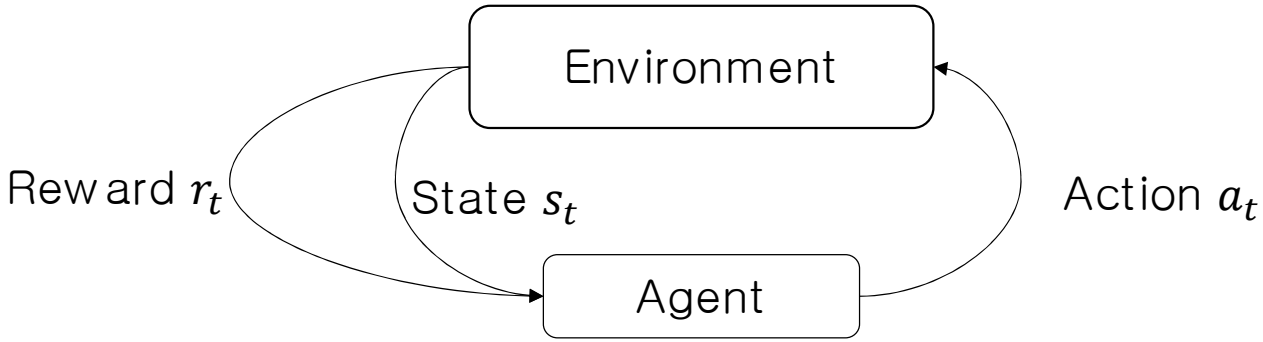
- Environment, Agent, Reward의 관계를 도식화
- 바둑
  - 목표: 상대의 돌 수보다 많아야 함
  - State: 흑백 돌의 위치
  - Actions: 빈 곳에 돌을 둠
  - Reward: 바둑판 위 내 돌의 수



→ Reward 최대화 ~ 목표 달성

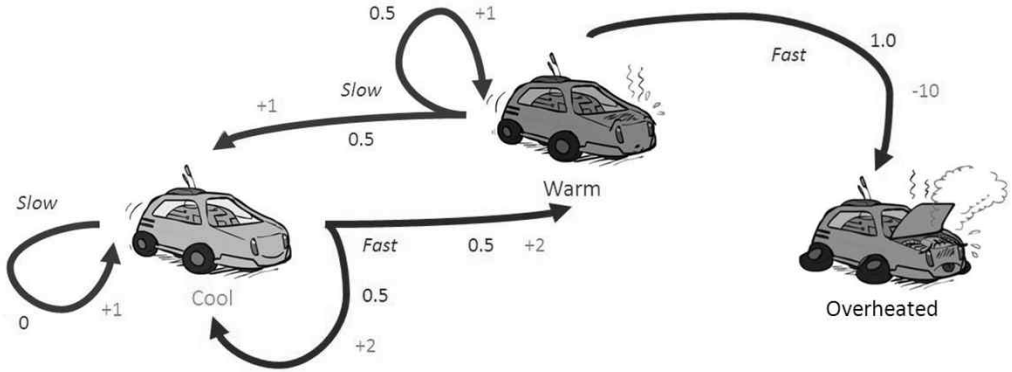
2 Markov Property

- 확률 과정(stochastic process)에서, 현재 State만으로 Environment의 성질을 담아 내고자 하는 성질
- 오직 현재의 값을 이용하여 미래를 유추하며, 이 과정에서 과거의 값들은 어떤 추가 정보도 제공하지 않음



2 Markov Decision Process

- 수식화: $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathbb{P}, \gamma)$
  - $\mathcal{S}$: 모든 state들의 집합
  - $\mathcal{A}$: 모든 action의 집합
  - $\mathcal{R}$: 주어진 (state, action) pair에 대한 reward 확률분포
  - $\mathbb{P}$: 주어진 (state, action) pair에 대한 다음 state의 확률분포
  - γ: 미래 보상에 대한 감가율



https://medium.com/@sanchittanwar75/markov-chains-and-markov-decision-process-e91cda7fa8f2

2 Markov Decision Process

- 수식화: $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathbb{P}, \gamma)$
- 초기 상태 $(t = 0)$: $\mathrm{s_0}$
- For $t = 0$ until done:
  - Agent가 s_t를 분석하여 a_t를 선택
  - Environment는 보상 $\mathrm{r_t}$를 다음과 같이 sampling: $\mathrm{r_t} \sim \mathcal{R}(.|s_t, a_t)$
  - Environment는 다음 상태 s_{t+1}을 다음과 같이 sampling: $s_{t+1} \sim \mathbb{P}(.|s_t, a_t)$
  - Agent는 보상 $\mathrm{r_t}$와 s_{t+1}를 전달받음

Agent가 제어 가능한 부분

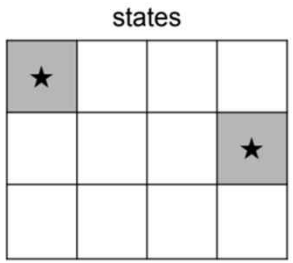
Agent가 제어하지 못하는 부분

- Agent가 제어 가능한 부분을 수학적으로 모델링해보자: policy $\boldsymbol{\pi}$

2 Markov Decision Process

- policy π
  - 주어진 state s에서 action을 선택하는 함수
  - Deterministic or Stochastic
- MDP로 본 강화학습의 목적
  - 다음을 최적화하는 π를 만들어보자!
  - $\sum_{t>0} \gamma^t r_t = r_0 + \gamma \cdot r_1 + \gamma^2 \cdot r_1 + \cdots + \gamma^T \cdot r_T$
- 길찾기 예제

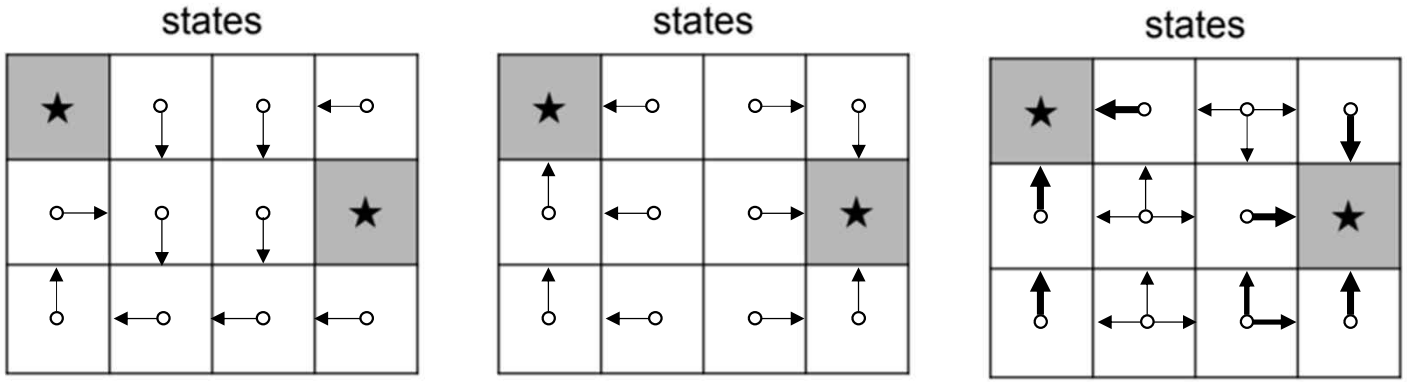
actions = {
1. right
2. left
3. up
4. down
}



Set a negative "reward"
for each transition
(e.g. r = -1)

2 Markov Decision Process

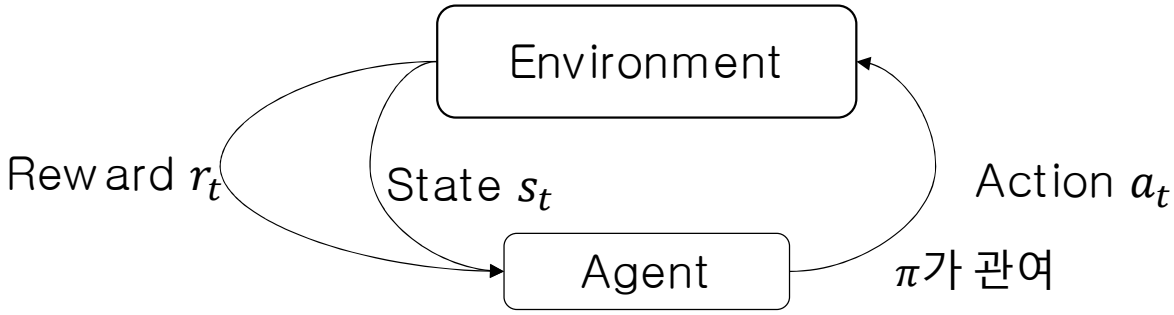
- 길찾기 예제에서 policy π 만들기
  - 목적: 임의의 cell에서 start했을 때, 별표 친 cell까지 빠르게 이동
  - Reward: action을 수행할 때마다 -1



나쁜 π 좋은 π 좋은 π (stochastic)

2 이상적 π^* 를 수식적으로 정의하기

- π^*는 보상을 최대화시키는 것이 목적
- $\mathbb{E}[\sum_{t\geq 0}\gamma^t r_t|\pi^*]$ 값이 최대값이 되어야함
  - 이때, $a_t \sim \pi^*(.|s_t)$, $s_{t+1} \sim \mathbb{P}(.|s_t, a_t)$
- 초기 상태 s_0에 대해 policy π environment가 다음 sequence를 생성
  - $s_0, a_o, r_0, s_1, a_1, r_1, \ldots$



2 Value Function와 Q-value Function

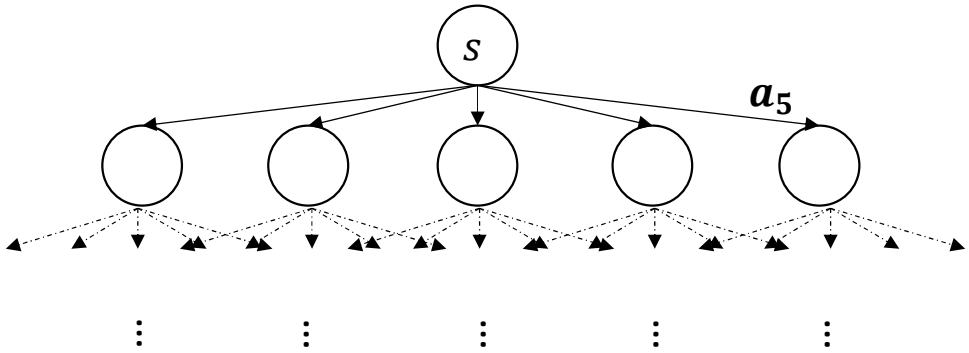
- Value Function: State s 가 주어졌을 때,
  π가 생성할 향후 누적보상에 대한 기대값
  - $V^{\pi}(s) = \mathbb{E}[\sum_{t\geq 0}\gamma^{t}r_{t}|s,\pi]$
- Q-Value Function: s 에서 action a를 선택했을 때,
  π가 생성할 향후 누적보상에 대한 기대값
  - $Q^{\pi}(s,a) = \mathbb{E}[\sum_{t\geq 0}\gamma^{t}r_{t}|s,a,\pi]$

2 Optimal Q-value Function

- Optimal Q-Value Function: s 에서 action a를 선택했을 때, 얻을 수 있는 Q-Value의 최댓값
  - $Q^*(s,a) \quad \max_\pi Q^\pi(s,a) = \max_\pi \mathbb{E}[\sum_{t\geq 0} \gamma^t r_t | s, a, \pi]$
  - 정의 목적
    - → $Q^*(s,a)$가 가장 높은 a를 선택하는 π 를 만들어보자.

2 Optimal Q-value Function

- $Q^*(s,a)$ $\max_\pi Q^\pi(s,a)$ $\max_\pi \mathbb{E}[\sum_{t\geq 0}\gamma^t r_t|s,a,\pi]$



$Q^*(s,a)$	10	24	-20	57	**80**

- 다음 action으로 가장 적합한 것은?

  → a_5

- Optimal Policy $\pi^*(s) = \arg\max_\pi Q^*(s,a)$

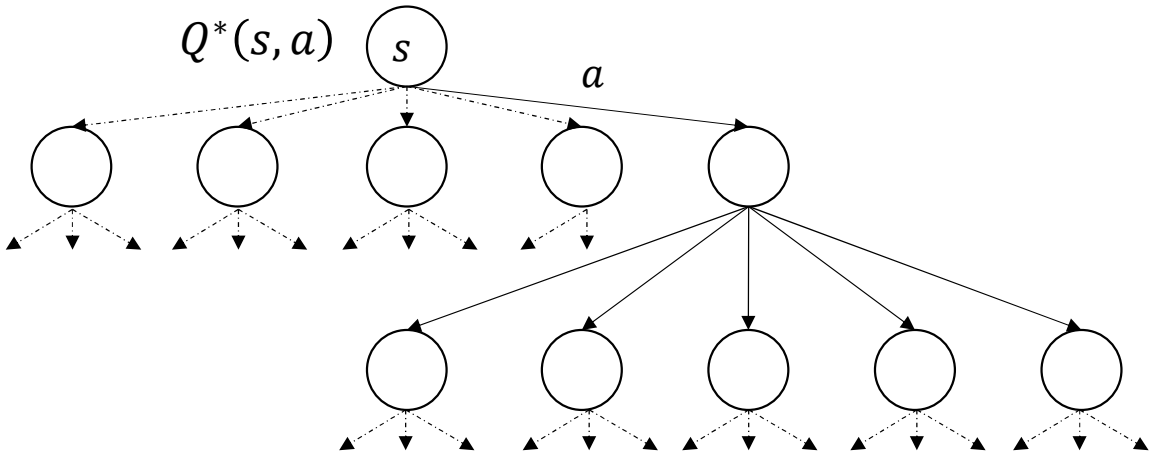
2 Q^*를 어떻게 구하는가?

- Optimal Policy $\pi^*(s) = \underset{\pi}{argmax}\ Q^*(s,a)$
- 가장 이상적인 Q-value Function Q^*
  - $Q^*(s,a) = \max_{\pi} \mathbb{E}[\sum_{t\geq 0} \gamma^t r_t | s, a, \pi]$

- 남은 문제: Q^*를 어떻게 구하는가?
  - Bellman Equation 속성을 이용
    - → $Q^*(s,a) = \mathbb{E}[r + \gamma \cdot \max_{a'} Q^*(s',a') | s, a]$

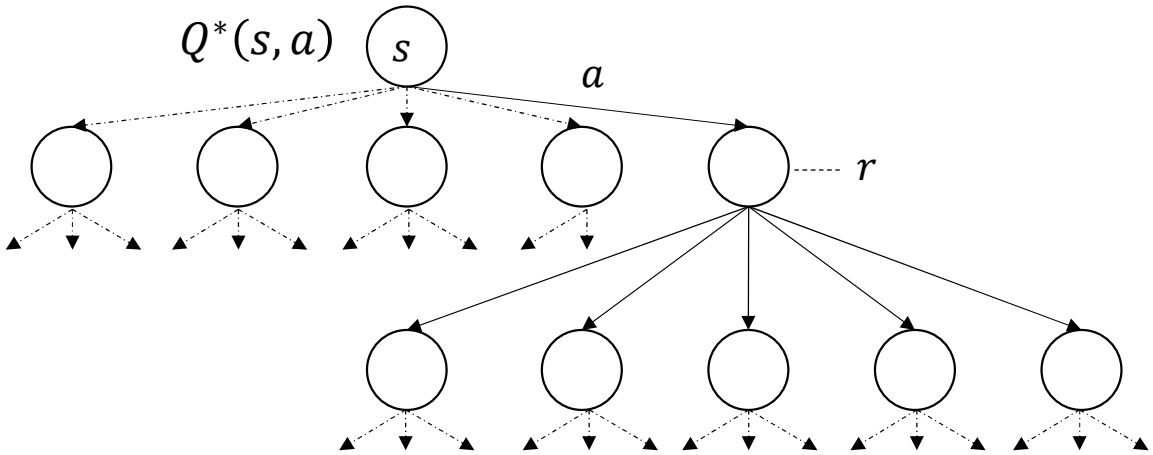
2 Q^*를 어떻게 구하는가?

- 남은 문제: Q^*를 어떻게 구하는가?
  - Bellman Equation 속성을 이용
    - → $Q^*(s,a) = \mathbb{E}[r + \gamma \cdot \max_{a'} Q^*(s',a')|s,a]$



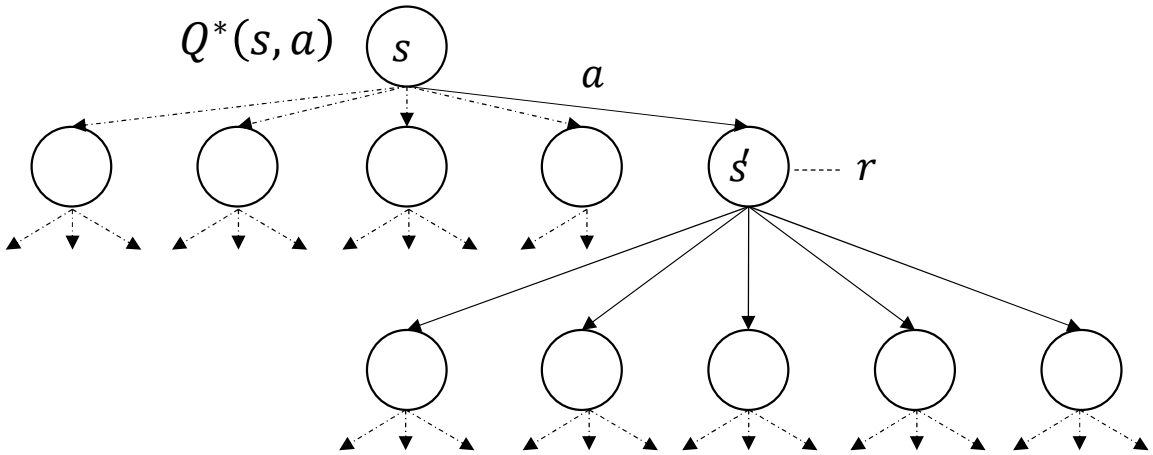
2 Q^*를 어떻게 구하는가?

- 남은 문제: Q^*를 어떻게 구하는가?
  - Bellman Equation 속성을 이용
    - → $Q^*(s,a) = \mathbb{E}[r + \gamma \cdot \max_{a'} Q^*(s',a')|s,a]$



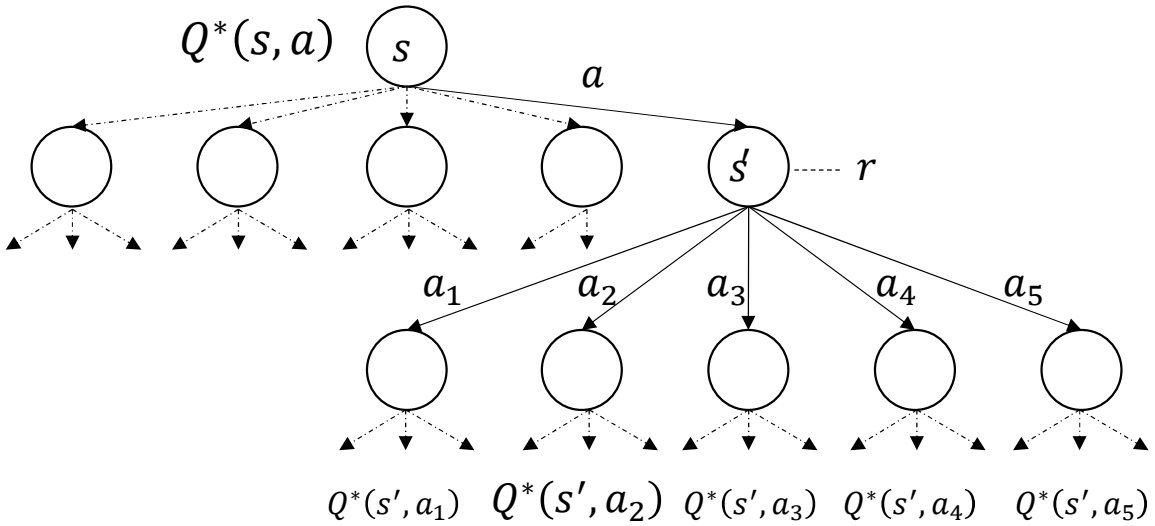
2 Q^*를 어떻게 구하는가?

- 남은 문제: Q^*를 어떻게 구하는가?
  - Bellman Equation 속성을 이용
    - → $Q^*(s,a) = \mathbb{E}[r + \gamma \cdot \max_{a'} Q^*(s',a')|s,a]$



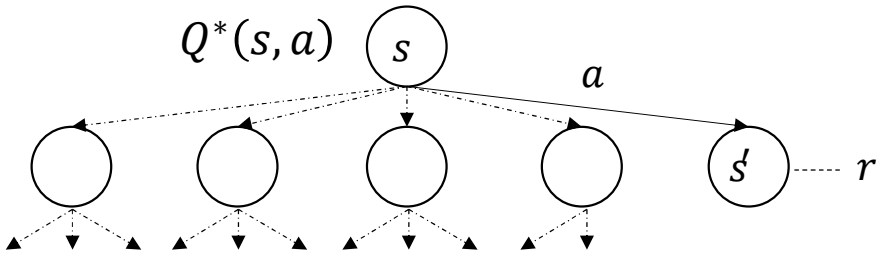
2 Q^*를 어떻게 구하는가?

- 남은 문제: Q^*를 어떻게 구하는가?
  - Bellman Equation 속성을 이용
    - → $Q^*(s,a) = \mathbb{E}[r + \gamma \cdot \max_{a'} Q^*(s',a')|s,a]$



2 Q^*를 어떻게 구하는가?

- 남은 문제: Q^*를 어떻게 구하는가?
  - Bellman Equation 속성을 이용
    - → $Q^*(s,a) = \mathbb{E}[r + \gamma \cdot \max_{a'} Q^*(s',a')|s,a]$

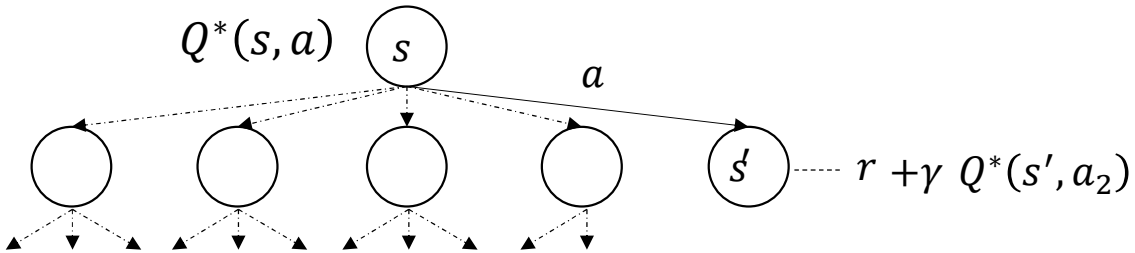


한 스텝 뒤의 보상이므로 감가비율 곱함

$+\gamma\ Q^*(s',a_2)$

2 Q^*를 어떻게 구하는가?

- 남은 문제: Q^*를 어떻게 구하는가?
  - Bellman Equation 속성을 이용
    - → $Q^*(s,a) = \mathbb{E}[r + \gamma \cdot \max_{a'} Q^*(s',a')|s,a]$



12강. 강화학습 *Deep Learning*

3. Q-Learning과 DQN

Q-Learning
Deep Q-Network

3 Q-Learning의 원리

- Value Iteration
  - s와 a에 대한 $Q_0(s,a)$를 random으로 초기화 후
  - Q_t가 수렴할 때 까지 아래 식을 반복
    - → $Q_{t+1}(s,a) := \mathbb{E}\left[r + \gamma \cdot \max_{a'} Q_t(s',a') | s,a\right]$

- Q-Learning with Value iteration
  - Q를 deep neural network로 예측해보자!

3 Q-Learning Learning 예제

Game Board:

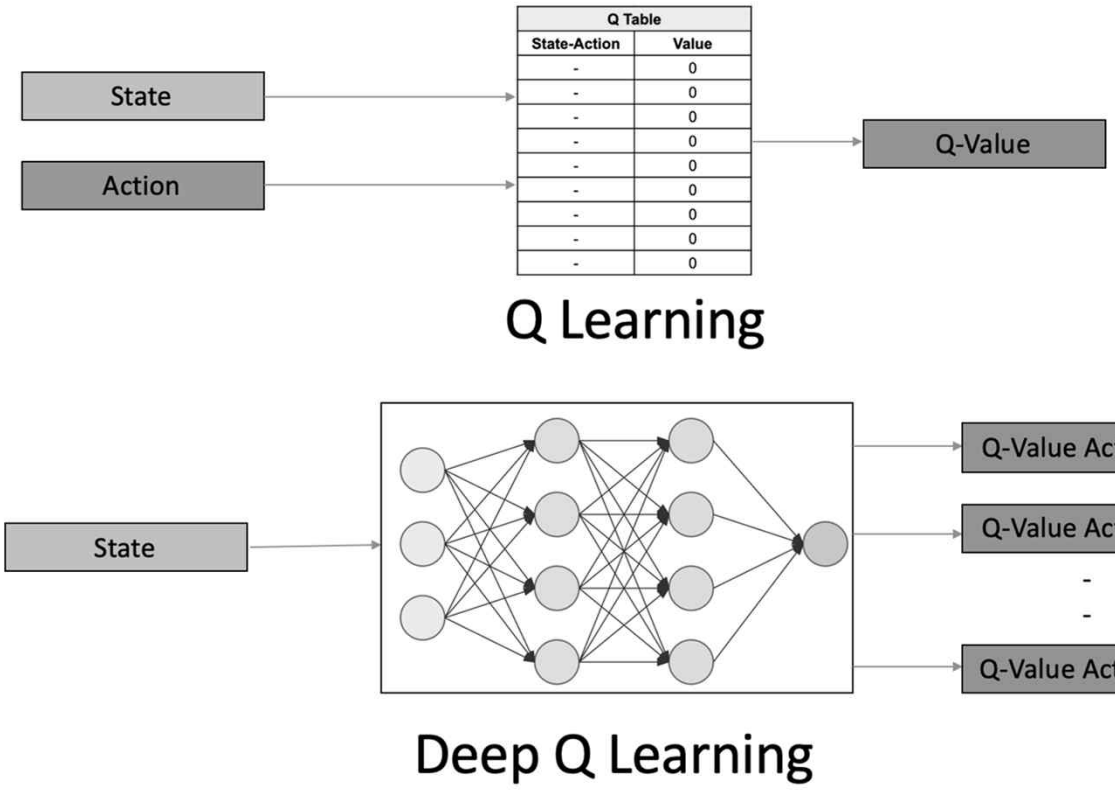
Current state (*s*): 0 0 0
0 1 0

Q Table: γ = 0.95

	0 0 0 1 0 0	0 0 0 0 1 0	0 0 0 0 0 1	1 0 0 0 0 0	0 1 0 0 0 0	0 0 1 0 0 0
⇧	0.2	0.3	1.0	-0.22	-0.3	0.0
⇩	-0.5	-0.4	-0.2	-0.04	-0.02	0.0
⇨	0.21	0.4	-0.3	0.5	1.0	0.0
⇦	-0.6	-0.1	-0.1	-0.31	-0.01	0.0

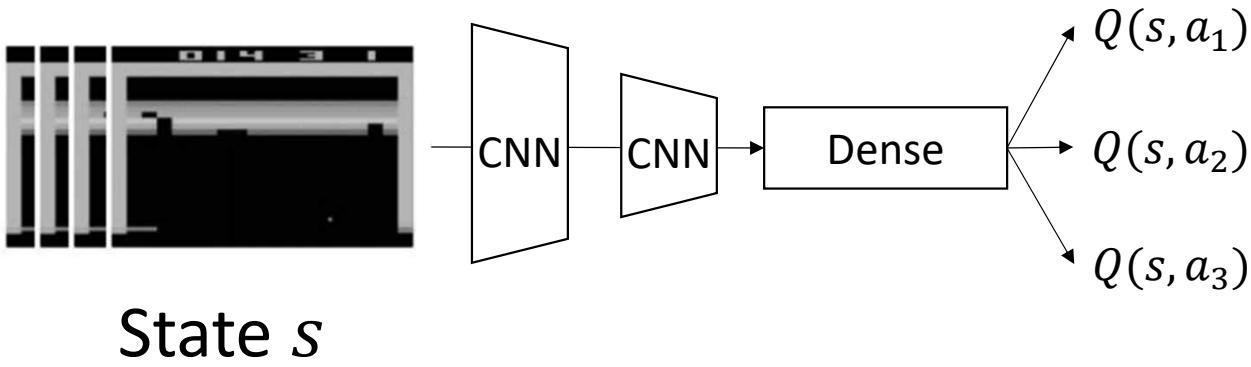
3 Deep Q-Network

- Q를 deep neural network로 대체



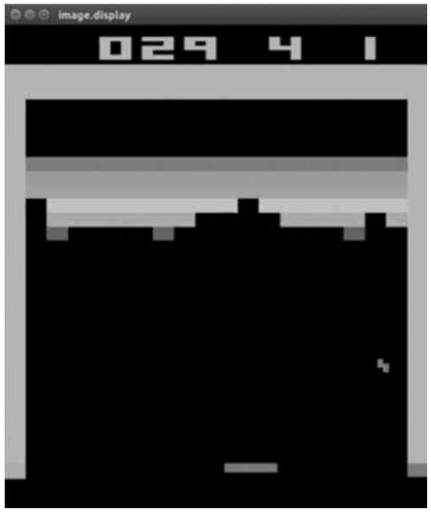
3 Deep Q-Network

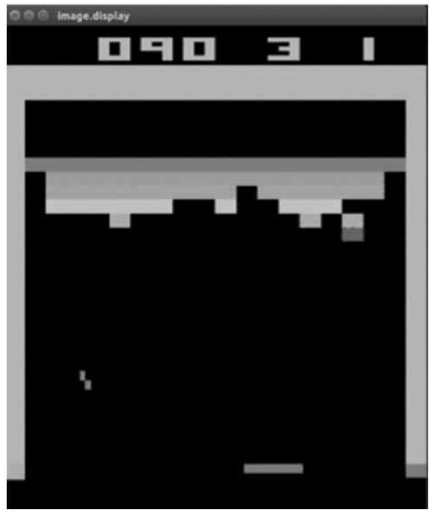
- 상태에 대한 handcrafted feature 추출 대신 이미지 자체를 입력으로 활용
- 입력: 84x84 이미지 프레임 4개
  - 공의 속도를 반영하기 위해 다중 프레임 사용 이유

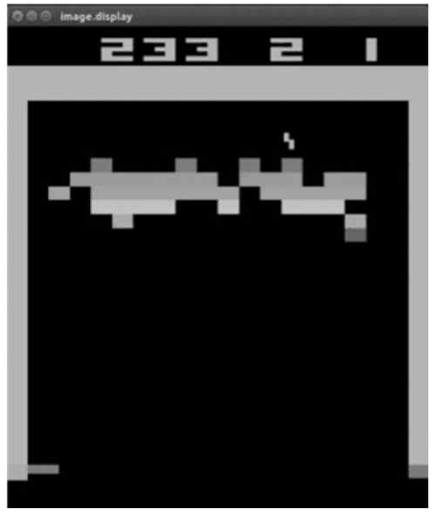


3 Deep Q-Network

- 학습된 결과







3 더 읽을 거리

- Kaelbling, Leslie P.; Littman, Michael L.; Moore, Andrew W. (1996). "Reinforcement Learning: A Survey". Journal of Artificial Intelligence Research. 4: 237–285.
- Sorokin, Ivan, et al. "Deep attention recurrent Q-network." arXiv preprint arXiv:1512.01693 (2015).
- Gu, Shixiang, et al. "Continuous deep q-learning with model-based acceleration." International Conference on Machine Learning. 2016.
- https://gym.openai.com/

딥러닝특론 다음시간안내

13강. GAN (1): GAN 기초