NOVA

AI / Data Engineering Track

Playing Atari with Deep Reinforcement Learning

V Mnih, K Kavukcuoglu, D Silver, A Graves, I Antonoglou, D Wierstra, M Riedmiller

arXiv preprint (2013)

숭실대학교 전기공학부 학술 소모임 NOVA
발표자 : 우상윤

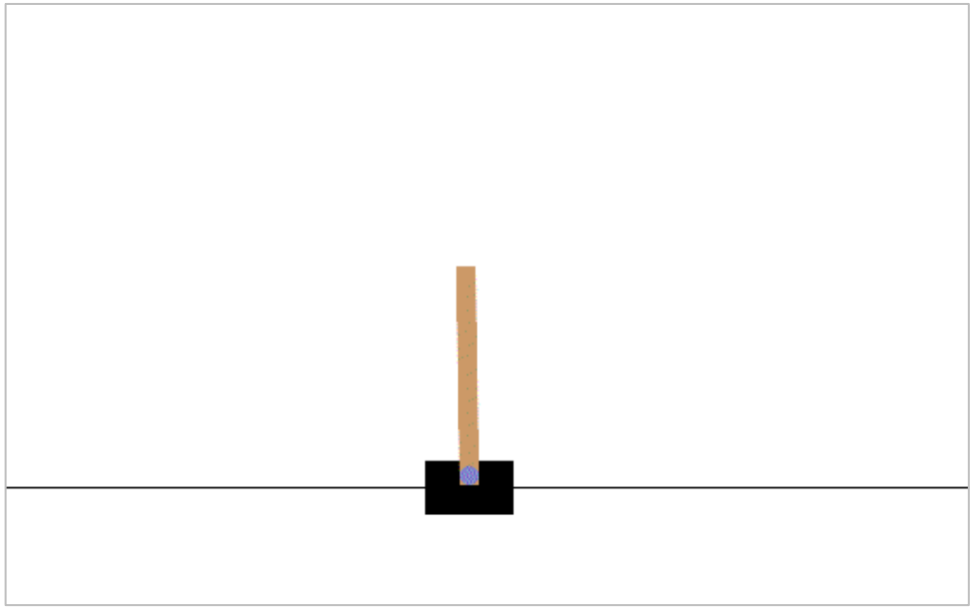
Contents

01 Introduction

강화학습

에이전트(행동 주체)가 정의된 환경에서 시행착오를 반복하며 보상을 최대화하는 방향으로 행동 방침을 학습하는 머신러닝 기법

이전에는 원시 데이터로부터 사람이 직접 특징을 추출하고, 강화학습 알고리즘이 학습



대표적인 강화학습 예제 (Cartpole)

카트를 제어해 막대의 균형을 유지하는 것이 목표

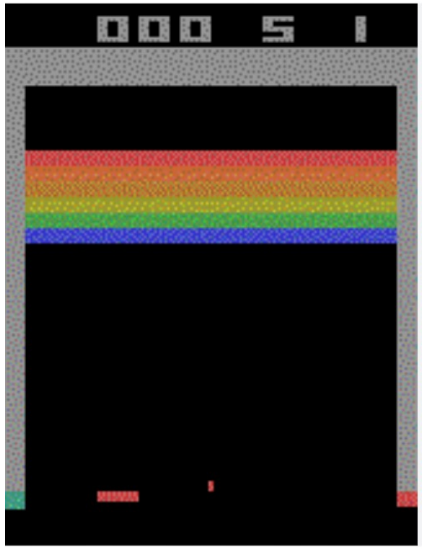
변수 : 카트의 위치, 카트의 속도, 막대의 각도, 막대의 각속도

변수도 적고 행동도 좌/우 이동 뿐이라 단순하다.

01 Introduction

환경이 복잡해진다면:

- 사람이 직접 특징을 만들어 제공하는 데에는 한계가 있음
- 상태 수에 따라 행동을 저장하는 공간(Q-table)이 기하급수적으로 늘어남
- 행동의 가치(Q)를 함수로 근사하는 데에는 한계가 있었음



만약 이 사진만을 그대로 입력으로 제공한다면:

한 프레임당 $210 \times 160 \times 3 = 100800$개의 값이 존재, 여기서 특징을 설계해야 함

그러나 특징은 매 게임마다 달라짐

01 Introduction

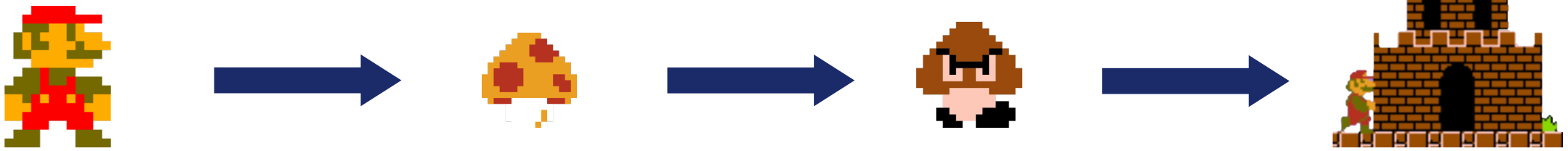
한편, 지도학습/비지도학습 분야에서는 딥러닝의 발전으로 인해 컴퓨터비전과 음성인식 등의 분야에서 큰 성과를 이룸

또한 원시 데이터로부터 고수준의 특징을 추출하는 것이 가능해짐

그러면 강화학습 분야에서도 좋은 성과를 기대할 수 있지 않을까?

01 Introduction

하지만 강화학습과 딥러닝을 결합하는 과정은 큰 문제점이 있었음



t=1

오른쪽으로 이동

t=5

아이템 획득

t=10

적 처치

t=30

완주 성공,

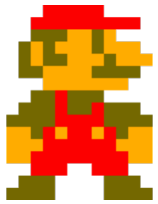
보상 + 100

근데 이 보상이 어떤 행동 덕분에 받은 거지?

(정답이 바로 주어지지 않음, 학습 신호가 다름)

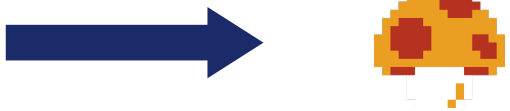
01 Introduction

하지만 강화학습과 딥러닝을 결합하는 과정은 큰 문제점이 있었음



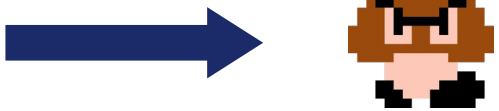
t=1

오른쪽으로 이동



t=5

아이템 획득



t=10

적 처치

t=30

완주 성공,

보상 + 100

보상이 0

(어떤 행동이 좋은 것인지 알 수 없음)

01 Introduction

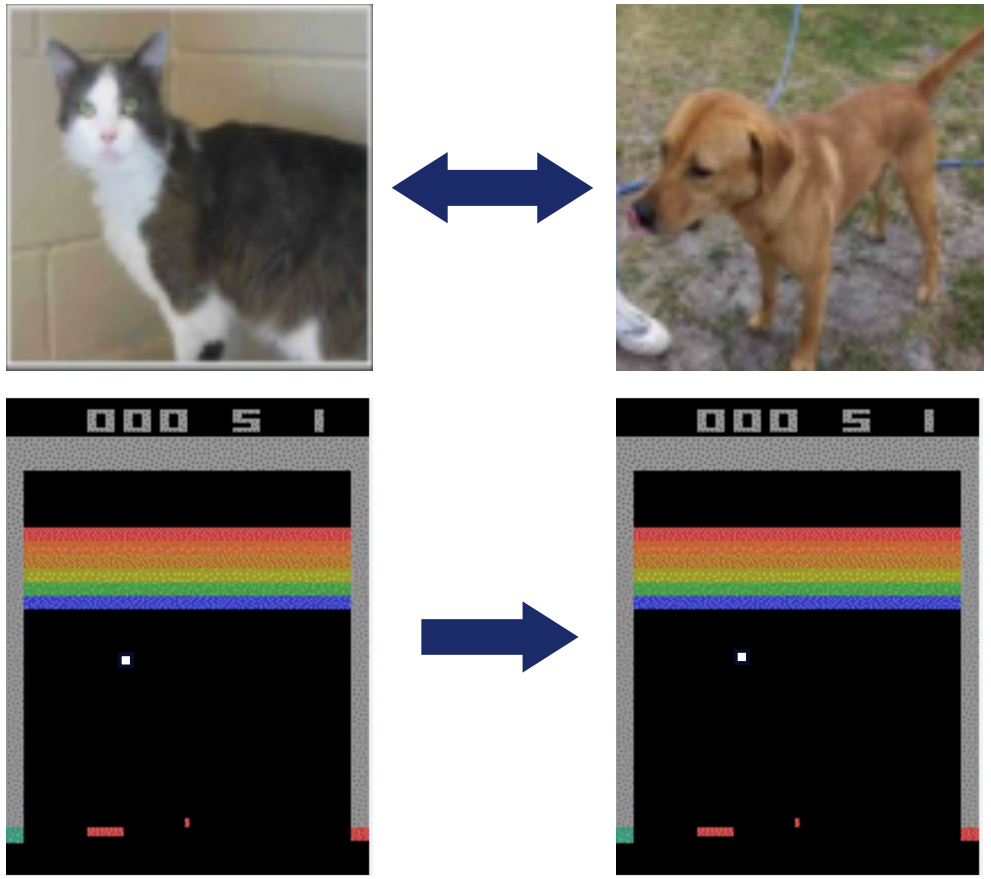
하지만 강화학습과 딥러닝을 결합하는 과정은 큰 문제점이 있었음

보상이 불안정하거나 운에 따라 흔들림

(학습 신호가 깨끗하지 않음)

01 Introduction

하지만 강화학습과 딥러닝을 결합하는 과정은 큰 문제점이 있었음



대부분의 딥러닝에서는

모든 데이터가 서로 독립적이고 무작위로 섞여 있음

강화학습 데이터는 연속이기에 강하게 상관되어 있음

특정 방향으로 업데이트가 편향되어 학습이 불안정

01 Introduction

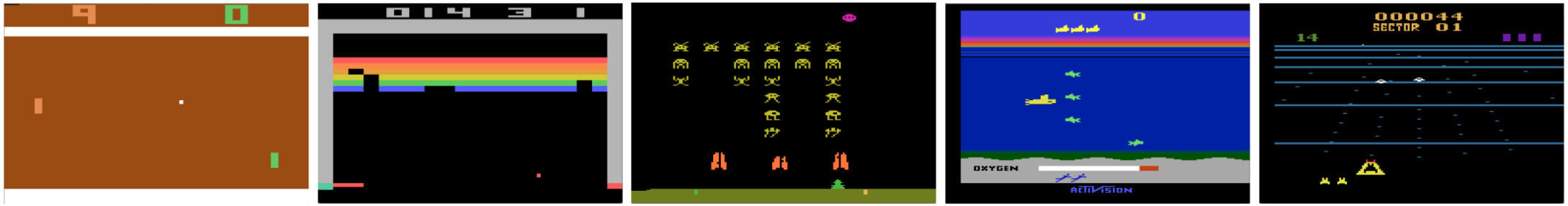
그러면 강화학습 알고리즘은 그대로 두고,

미래 보상을 예측하는 함수(Q-function)를 딥러닝(CNN)을 통해 근사해보면 어떨까?

데이터의 분포 문제와 상관성 문제는 Experience Replay 기법을 적용해 보자

= 이 아이디어 덕분에 CNN + RL 학습이 처음으로 안정적으로 동작하기 시작

여러 Atari 게임에서 성공적인 플레이를 보였음



02 Background

강화학습의 구성 요소

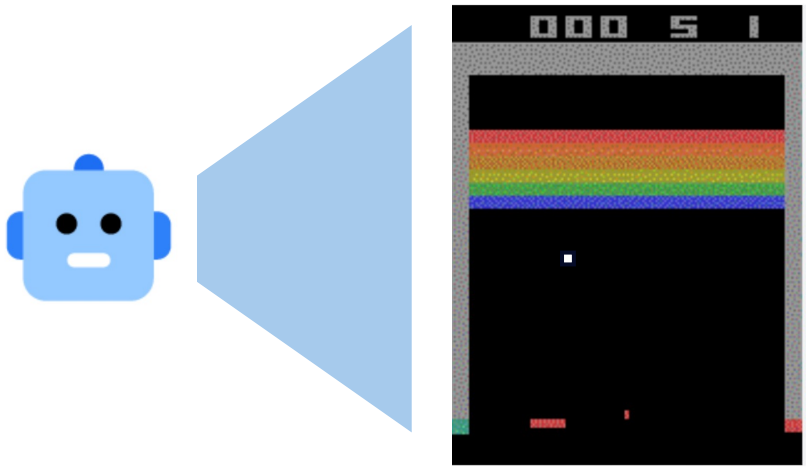
- Agent : 학습 주체
- Environment ($\mathcal{E}$) : 에이전트가 상호작용하는 외부 세계
- State (S) : 에이전트가 인식하는 현재 환경의 상황
- Action (A) : 에이전트가 취할 수 있는 동작
- Reward (R) : 에이전트의 행동에 대한 환경의 피드백
- Policy (π) : 상태에 따라 행동을 결정하는 에이전트의 행동 규칙
- Value Function : 현재 상태에서 앞으로 받을 보상의 누적 기대치

02 Background

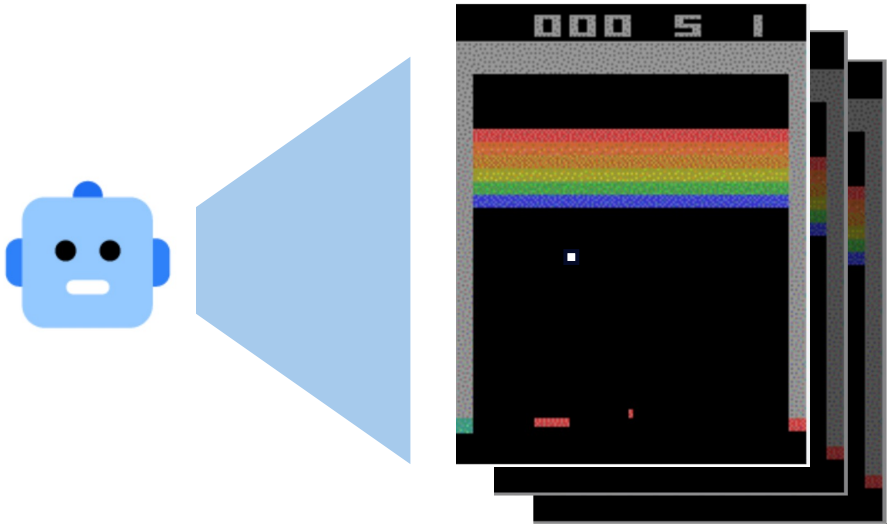
본 논문에서는 에이전트가 내부 상태를 직접 관찰할 수 없으며,

현재 화면을 나타내는 이미지를 벡터 $x_t \in \mathbb{R}^d$ 형태로 관측

단, 하나의 이미지 만으로는 현재 상황을 완전히 이해하기 어렵기에, 행동과 관측의 시퀀스를 상태로 정의



공이 어느 방향으로 움직이는지 알 수 없음



$$s_t = x_1, a_1, x_2, a_2, \ldots, a_{t-1}, x_t$$

02 Background

모든 시퀀스가 유한한 시간 내에 종료된다고 가정하고, 각 시퀀스를 하나의 상태로 가정한다면

표준적인 강화학습 방법을 적용할 수 있음 (Markov Decision Process, MDP)

Markov Decision Process (MDP)

강화학습 문제를 표현하는 수학적 모델

'다음 상태의 확률분포는 현재 상태와 행동만으로 결정된다'

02 Background

에이전트의 목표 : 미래 보상의 합을 최대화하도록 행동 선택

$$R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$$

에피소드 종료 시점

시간 t'에서 받는 보상

할인 계수 (Discount Factor) : 미래 보상의 중요도를 줄이는 계수 (현재 보상 > 미래 보상)

시간 t에서의 총 미래 보상

02 Background

최적 행동-가치 함수 $(Q^*(s,a))$

$$Q^*(s,a) = \max_{\pi} \mathbb{E}[R_t | s_t = s, a_t = a, \pi]$$

어떤 상태 시퀀스 s를 관측한 후 행동 a를 취하고 특정 정책 π를 따랐을 때 얻을 수 있는 최대 기대 보상

이는 Bellman Equation을 만족

02 Background

Bellman Equation

상태 s'에서 가능한 모든 행동 a'에 대해 최적 가치 $Q(s', a')$를 알고 있다면,

최적의 전략은 다음 값을 최대화하는 행동을 선택하는 것

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

따라서

$$Q^*(s, a) = \mathbb{E}_{s' \sim \mathcal{E}}[r + \gamma \max_{a'} Q^*(s', a') \mid s, a]$$

= 상태 s에서 행동 a의 가치는 현재 보상 + 다음 상태에서 얻을 수 있는 최고의 미래 가치

02 Background

많은 강화학습 알고리즘의 기본 아이디어 : BeIlman Equation을 반복하자

이론상 반복할수록 Q값이 Q^*(최적값)로 수렴하기 때문

그러나 실제로는 각 시퀀스별로 행동-가치 함수가 따로 계산되기 때문에 일반화 불가능

함수 근사를 통해 행동-가치 함수를 추정

선형 함수 근사기를 사용하지만, 신경망과 같은 비선형 함수 근사기를 사용하기도 하며,

가중치 θ를 가진 신경망 함수 근사기를 Q-Network라 부름

02 Background

Q-Network

Q-Network는 다음 손실 함수를 최소화하도록 학습됨

$$L_i(\theta_i) = \mathbb{E}_{s,\, a\sim\rho(\cdot)}\left[\left(y_i - Q(s,a;\theta_i)\right)^2\right]$$

$$\mathbb{E}_{s'\sim\varepsilon}\left[r + \gamma \max_{a'} Q(s',a';\theta_{i-1}) \middle| s,a\right]$$

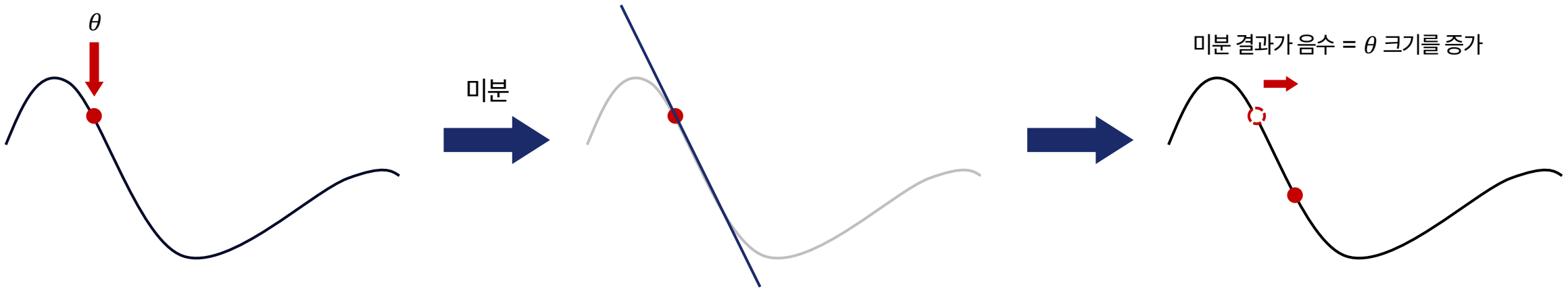
Bellman Equation을 통해 계산된 정답(target)값

지도학습에서는 target 값이 고정되어 있으나, 강화학습에서는 θ값에 따라 target 값이 바뀜

02 Background

Loss를 줄이기 위해서는 θ를 어떻게 변화시켜야 하는가?

함수를 미분해 어느 방향으로 가중치를 바꿔야 하는지를 결정하자 (경사 하강법)



현재 위치에서 더 낮은 곳으로 이동하려면

함수를 미분해 기울기를 계산하여

적절한 행동을 취하면 된다

02 Background

따라서 손실함수를 미분하면:

$$\nabla_{\theta_i} L_i(\theta_i) = \mathbb{E}_{s,\, a\sim\rho(\cdot);\, s'\sim\mathcal{E}}\left[\left(r + \gamma \max_{a'} Q(s', a'; \theta_{i-1}) - Q(s, a; \theta_i)\right)\nabla_{\theta_i} Q(s, a; \theta_i)\right]$$

원래는 모든 데이터에 대한 gradient를 계산해야 함

그러나 데이터 수가 너무 많아 이를 매번 계산하는 것은 너무 느림

따라서 대표 샘플을 임의로 뽑아 근사 (정확하지는 않지만 평균적으로는 올바른 방향) : 확률적 경사 하강법

02 Background

이 알고리즘은 model-free 방식 (환경을 수식으로 모델링하지 않음)

대신 직접 플레이해 얻은 데이터를 사용

행동은 최적 행동을 선택하되, 가끔씩 다른 랜덤한 선택을 하지만 (ε-greedy)

학습은 항상 가장 좋은 행동을 기준으로 진행함 (off-policy)

03 Related Work

TD-gammon

'백개먼' 이라는 게임을 플레이하는 강화학습 프로그램

그러나 체스, 바둑같은 게임에 적용하자 실패 (복잡한 환경)

강화학습 + 신경망

Q-Network가 발산하게 되어 학습이 되지 않음

강화학습 연구가 안정적인 수렴 특성을 가지는 선형 함수 근사기에 집중하게 됨

03 Related Work

이미지 인식에서 딥러닝이 성공적으로 작동함에 따라, 강화학습 + 딥러닝에 대한 연구가 다시 활발해짐

Q-Network의 발산 문제도 부분적으로 해결이 되었으나,

여전히 비선형 제어 문제까지 확장되지는 못했음

03 Related Work

Neural Fitted Q-learning (NFQ)

$$L_i(\theta_i) = \mathbb{E}_{s,\, a\sim\rho(\cdot)}\left[\left(y_i - Q(s,a;\theta_i)\right)^2\right]$$

기존 연구 중 DQN과 가장 유사한 구조

Loss 함수를 순차적으로 최적화하며 파라미터를 업데이트하는 방식

그러나 배치 업데이트 방식을 사용하기에 한 번의 반복에 필요한 계산량이 데이터셋 크기에 비례함

본 연구는 확률적 경사하강법 기반 업데이트를 사용하기에 반복 당 계산 비용이 적고 대규모 데이터셋으로 확장 가능

또한 NFQ와 달리 end-to-end 방식으로 강화학습을 수행하기에 직접적으로 유용한 특징 추출 가능

03 Related Work

이전의 Atari 게임 해결 방법

1. 초기 방법: 사람이 직접 특징을 만들어 주고 강화학습을 적용

   일반화가 어려우며, 선형 모델을 사용해 표현력이 제한적임

2. 더 많은 특징을 추출하고 압축해 사용 : 계산 효율 개선, 많은 특징 사용 가능

   여전히 특징은 사람이 설계하며, 정보 손실 존재 / 이미지를 직접적으로 이해하는 것이 아님

3. HyperNEAT (진화 알고리즘) : 학습이 아닌, 좋은 정책을 기반으로 진화를 시켜나가는 방법

   게임마다 구조가 달라지며, 버그성 전략을 이용함 / 올바른 정책을 학습한 것이라고 판단하기 어려움

04 Deep Reinforcement Learning

본 연구에서는 기존 방식 대신 Experience Replay라는 기법을 사용

현재 시점에서의 상태, 행동, 보상

$$e_t = (s_t, a_t, r_t, s_{t+1})$$

다음 시점의 상태

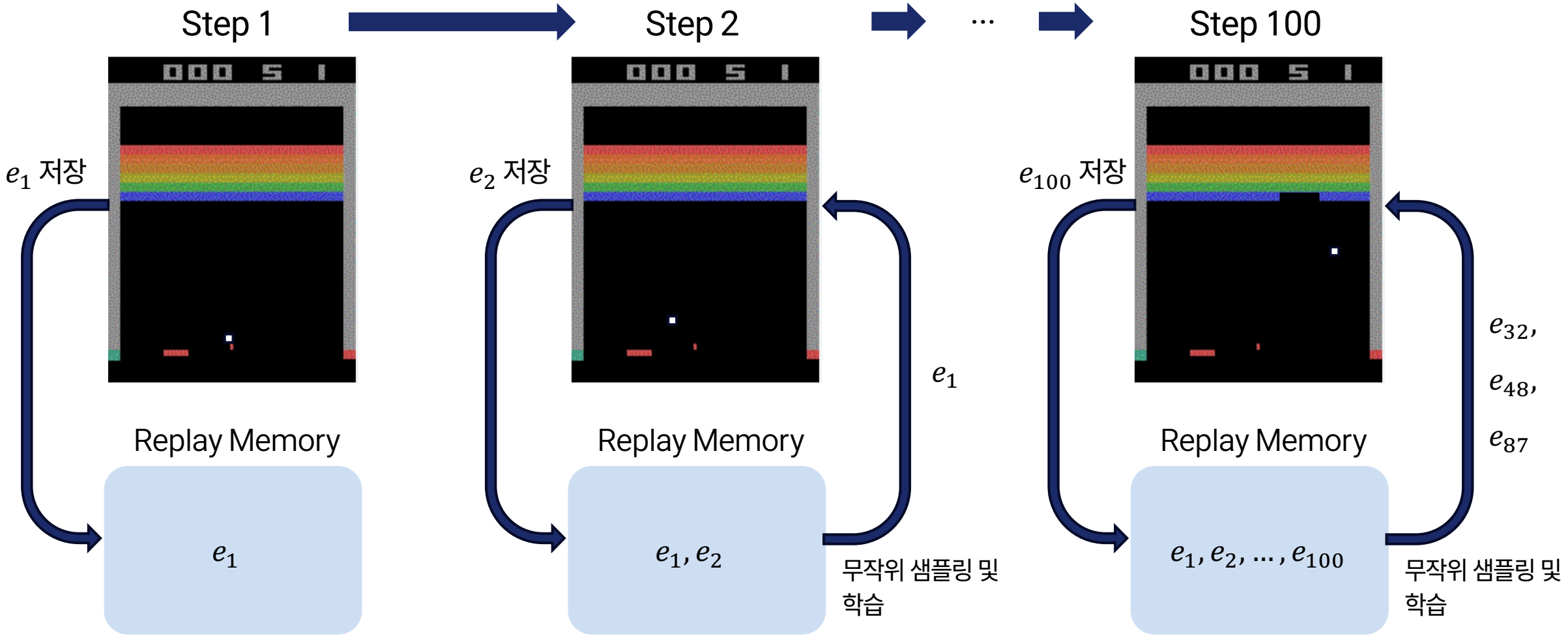
이 방법에서는 에이전트가 각 시간 단계에서 얻은 경험 e_t을 데이터셋에 저장하고,

여러 에피소드에 걸쳐 Replay memory로 유지함

내부에서는 Replay memory로부터 무작위 샘플을 선택해 Q-learning 업데이트 수행 (Experience replay)

Experience replay 수행 후 에이전트는 ε-greedy 정책에 따라 행동 선택 및 실행

04 Deep Reinforcement Learning



04 Deep Reinforcement Learning

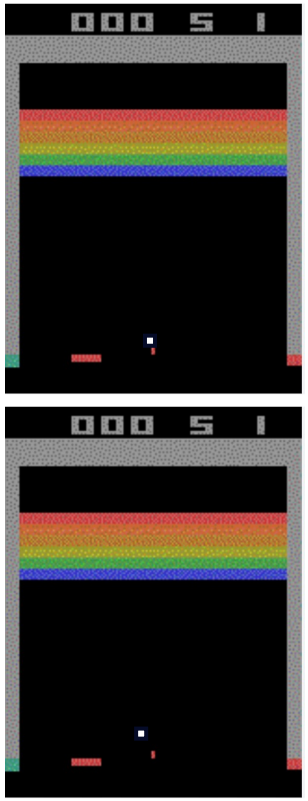
만약 Experience Replay가 없다면

기본 학습 루프

학습 순서

$e_1 \rightarrow e_2 \rightarrow e_3 \rightarrow e_4 \rightarrow \ldots$

- 데이터가 거의 비슷함
- 학습이 편향됨
- 발산 위험



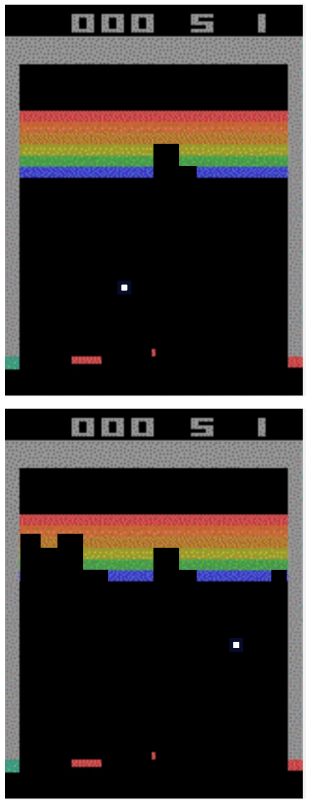
거의 차이 없음

Replay 사용 시

학습 순서

$e_{100} \rightarrow e_5 \rightarrow e_{472} \rightarrow \ldots$

- 다양한 상황이 융합
- 안정적 학습 가능



다양한 상황이 섞임

04 Deep Reinforcement Learning

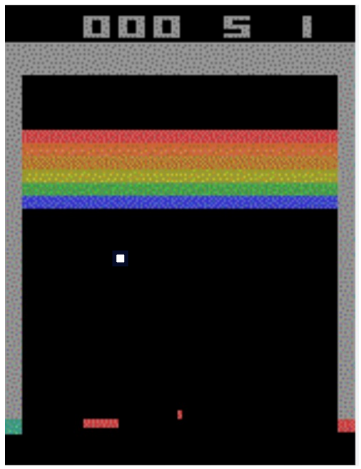
한 프레임만으로는 현재 상황을 알 수 없기에 시퀀스 형태로 상태를 정의한다고 하였으나,

실제 구현에서는 게임 진행 상황에 따라 길이가 계속해서 증가하기에 신경망 입력으로 사용할 수 없음

$t = 2$

학습 데이터

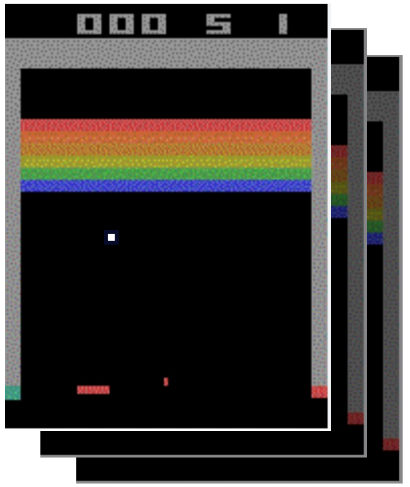
e_1 (길이 1)



$t = 100$

학습 데이터

$e_3, e_{20}, \ldots$ (길이 100)



길이가 계속해서 변하기 때문에

신경망 입력으로 사용 불가

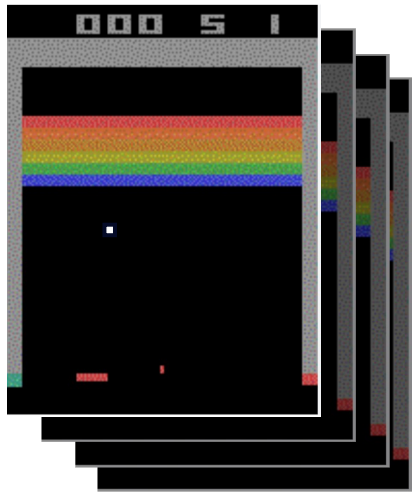
04 Deep Reinforcement Learning

따라서 과거 히스토리를 고정 길이로 압축하는 함수 ϕ를 사용해 길이 통일

$t = 2$

학습 데이터

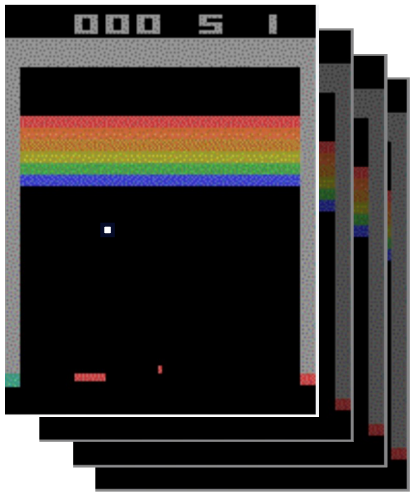
e_1, e_1, e_1, e_2 (길이 4)



$t = 100$

학습 데이터

$e_3, e_{20}, e_{57}, e_{82}$ (길이 4)



초기에 프레임 수가 부족한 경우, 프레임을 복제해서 입력으로 넣는다

04 Deep Reinforcement Learning

이러한 알고리즘은 Deep Q-learning이라 부름

Algorithm 1 Deep Q-learning with Experience Replay

Initialize replay memory $\mathcal{D}$ to capacity N
Initialize action-value function Q with random weights
for episode $= 1, M$ **do**
  Initialise 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 $(\phi_t, a_t, r_t, \phi_{t+1})$ in $\mathcal{D}$
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $\mathcal{D}$
    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 $(y_j - Q(\phi_j, a_j; \theta))^2$ according to equation 3
  end for
end for

04 Deep Reinforcement Learning

Deep Q-learning의 장점

1. 각 경험이 여러 번의 가중치 업데이트에 사용될 수 있어 데이터 효율성이 높음

2. 연속된 샘플은 서로 높은 상관관계를 띄고 있어 학습에 비효율적

   Deep Q-learning의 무작위 샘플링을 통해 상관관계를 깨고 업데이트의 분산을 감소시킬 수 있음

3. on-policy 방식은 원치 않는 피드백 루프가 생길 수 있고, 파라미터가 나쁜 지역 최적해에 갇히거나 발산할 수 있음

   그러나 해당 방식은 과거 여러 상태들의 데이터를 사용하게 되어 학습이 부드러워지고 진동과 발산을 방지할 수 있음

04 Deep Reinforcement Learning

이 방법의 한계는

- 중요한 경험과 그렇지 않은 경험을 구분하지 못함
- 메모리 크기가 제한되어 있어 오래된 경험은 삭제됨

이러한 한계는 중요한 경험을 우선적으로 선택하는 방식을 통해 개선 가능

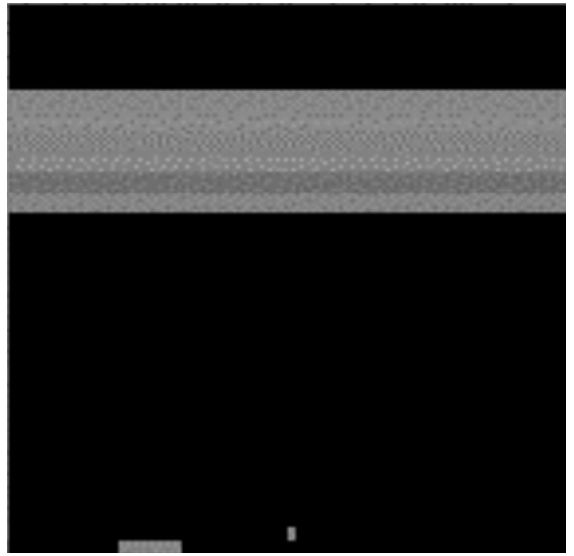
04 Deep Reinforcement Learning

실제 구현 – 이미지 전처리

210 x 160, 3채널(R,G,B)

입력 차원이 너무 크며,
색상은 별로 중요하지 않음
연산량 폭증, 학습 비효율

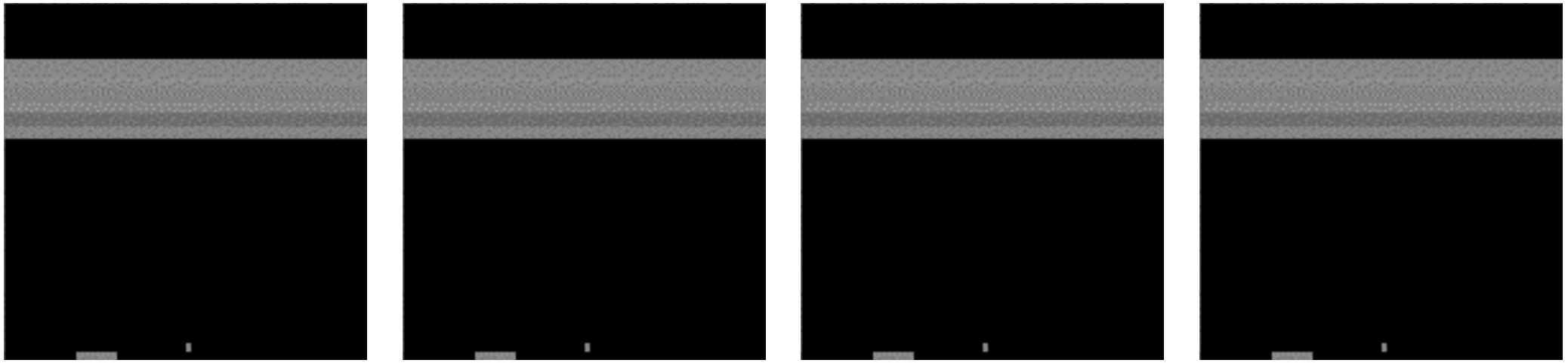
RGB(3차원)를 1차원으로 변환
해상도를 낮춰 불필요한 디테일 제거
학습에 중요한 게임 화면 부분만 남김



84 x 84, Grayscale, crop

04 Deep Reinforcement Learning

실제 구현 – 이미지 전처리

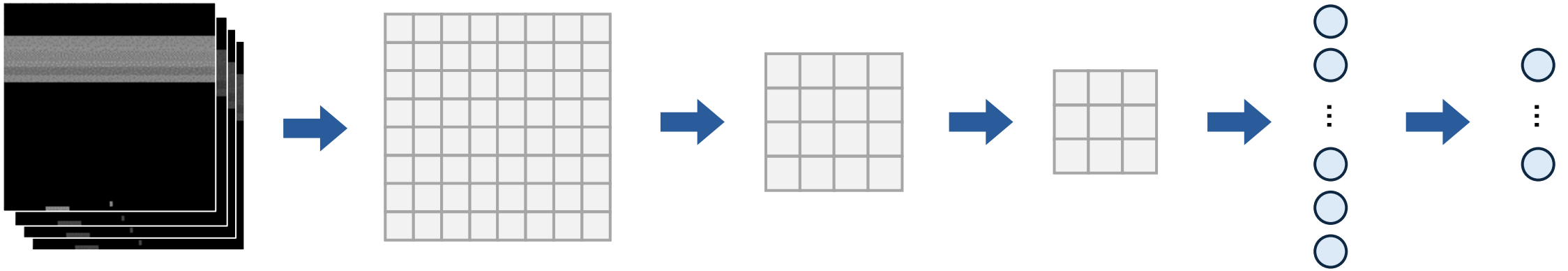


$t = T - 3$ $t = T - 2$ $t = T - 1$ $t = T$

과거 히스토리는 최근 4개의 히스토리를 전처리하여 구성

04 Deep Reinforcement Learning

실제 구현 – 모델 아키텍처



필터를 통해 중요한 정보를 추출하고 (convolution),
연산량을 줄이기 위해 데이터의 크기를 압축 (stride)

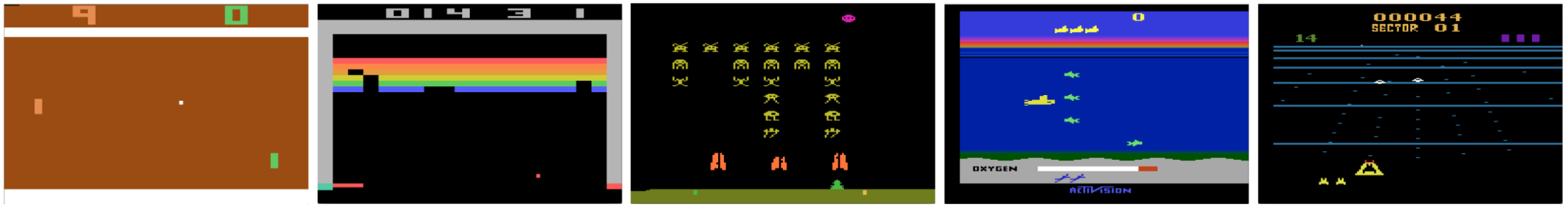
추출한 특징을 바탕으로 판단 진행

05 Experiments

실험은 Atari 게임 7개에서 수행되었으며,

모두 동일한 네트워크 구조와 학습 알고리즘, 하이퍼파라미터 설정을 사용함

게임은 수정되지 않은 환경에서 평가되었으나, 게임마다 점수의 스케일이 다르기에 보상 크기만 고정시켜 실험함

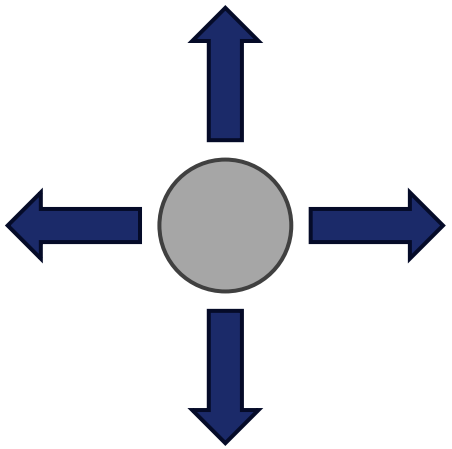


05 Experiments

학습은 1000만 프레임동안 진행되었으며,

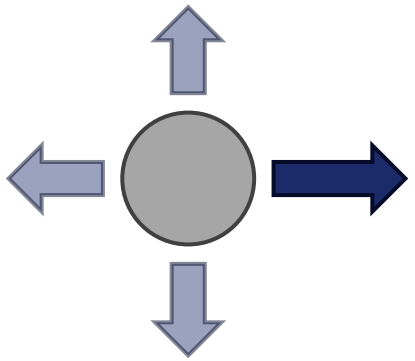
ε-greedy 방식에서 ε 값은 처음 100만 프레임 동안 1에서 0.1까지 선형적으로 감소, 이후 0.1로 고정

Replay memory는 가장 최근 100만 개 프레임을 저장하도록 설정



ε=1

무작위 행동 선택

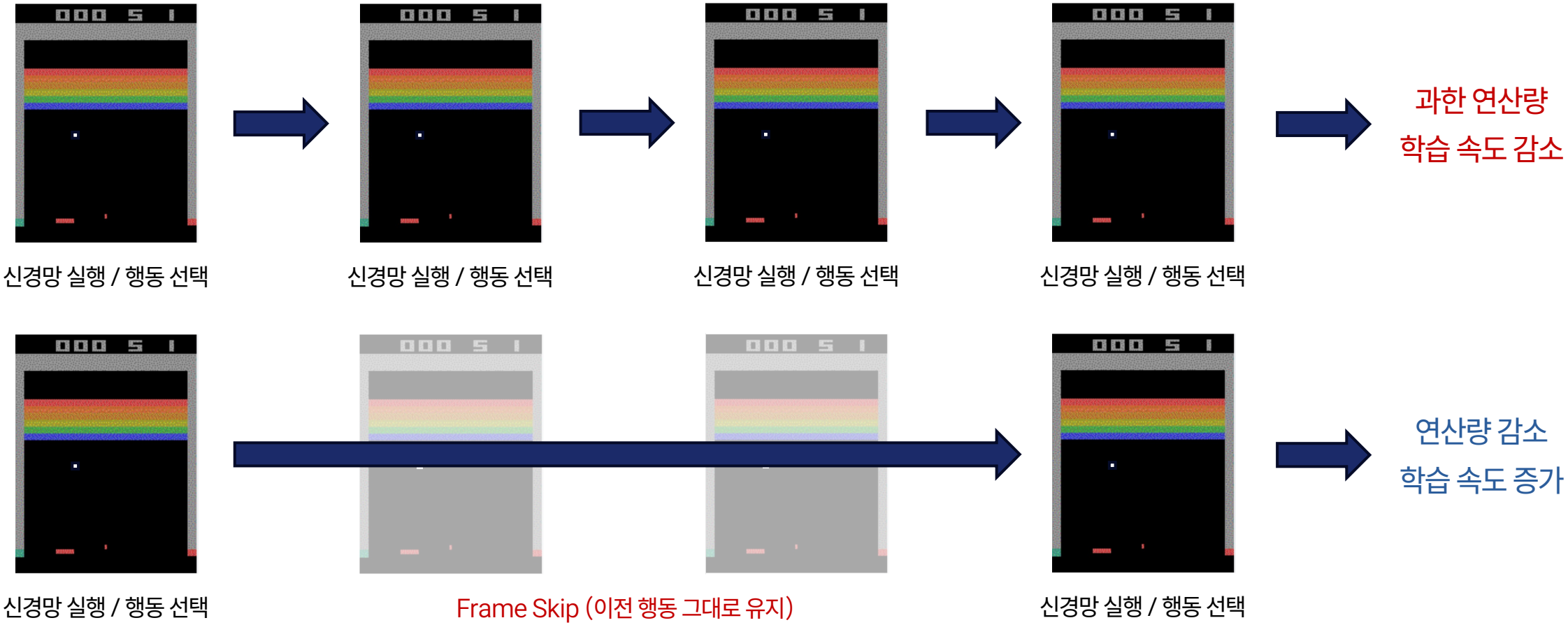


ε=0.1

90%로 최적 행동, 10%로 무작위 행동 선택

05 Experiments

이전 Atari 게임 플레이 연구들에 따라, Frame-skipping 기법을 사용해 연산량을 줄이고 데이터 효율을 증가시킴



05 Experiments

일반적인 지도학습과 달리, 강화학습은 학습이 잘 되고 있는지를 정확하게 평가하기 어려움

따라서 강화학습은 보통 한 에피소드에서 얻은 총 보상의 평균으로 성능을 평가함

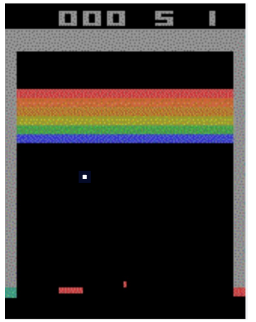
→ Dog → True

→ Dog → False

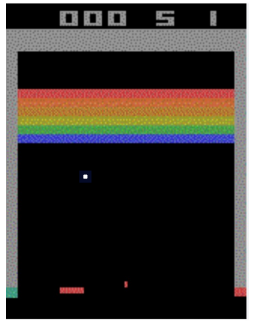
Accuracy = 0.99

(99 True / 1 False)

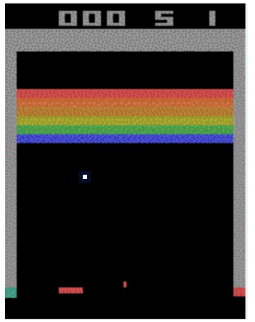
지도학습의 경우, 정답의 비율을 확인함으로써 모델을 평가하기 쉬움



첫번째 판 : 100점



두번째 판 : 300점



세번째 판 : 260점

평균 220점

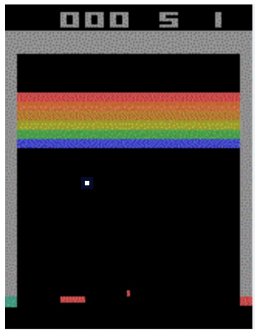
정답이 없음, '얼마나 잘했는지'로 평가해야 함

= 여러 판을 테스트해서 점수를 평균내자

05 Experiments

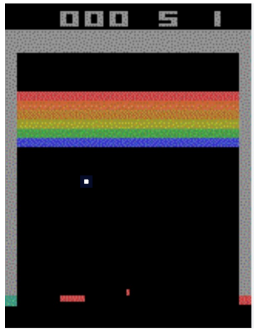
그러나 강화학습은 학습이 조금만 달라져도 정책이 완전히 변할 수 있기 때문에

노이즈가 너무 크게 발생함



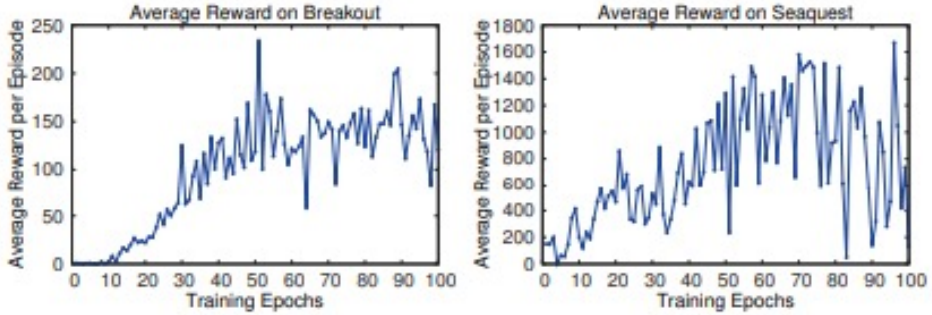
첫번째 학습 : 100점

두번째 학습: 1000점



세번째 학습 : 50점

정말 좋아지고 있는 것이 맞는가?



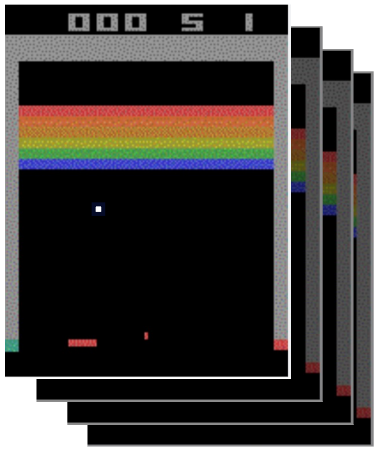
에피소드 보상의 평균이 학습마다 크게 변동

05 Experiments

해결 방법 : Q 값을 사용하자

└ 행동에 따라 받을 미래의 보상 예측값

본 실험에서는 :



첫번째 학습 (Epoch 1)

Q = 0.3

10번째 학습 (Epoch 10)

Q = 1.7

학습 이전에 랜덤 플레이를 통해 상태를 수집

수집된 상태 중 일부를 학습된 모델에서 테스트셋처럼 사용

05 Experiments

실제로 Q값의 평균을 사용한 결과 훨씬 더 부드럽고 안정적인 곡선이 나타남

또한 발산 문제가 일어나지 않고 안정적으로 학습이 가능함을 시사함

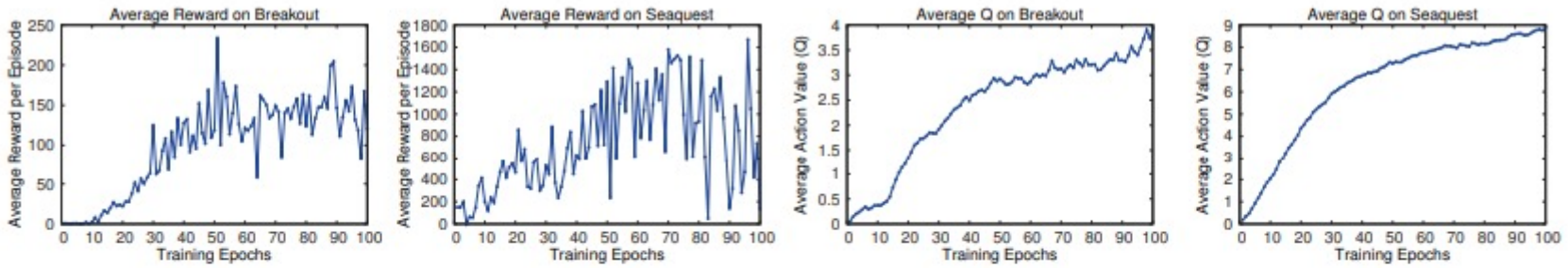
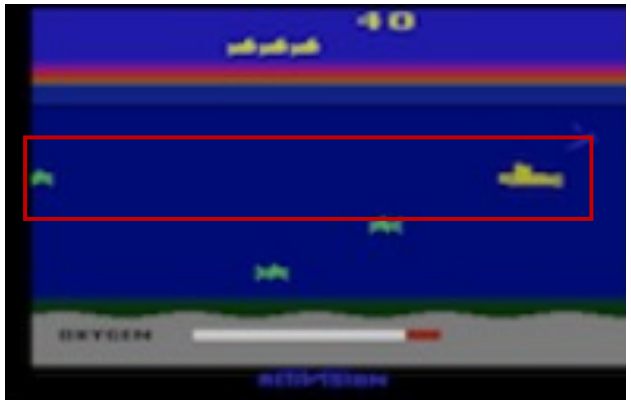


Figure 2: The two plots on the left show average reward per episode on Breakout and Seaquest respectively during training. The statistics were computed by running an ϵ-greedy policy with $\epsilon = 0.05$ for 10000 steps. The two plots on the right show the average maximum predicted action-value of a held out set of states on Breakout and Seaquest respectively. One epoch corresponds to 50000 minibatch weight updates or roughly 30 minutes of training time.

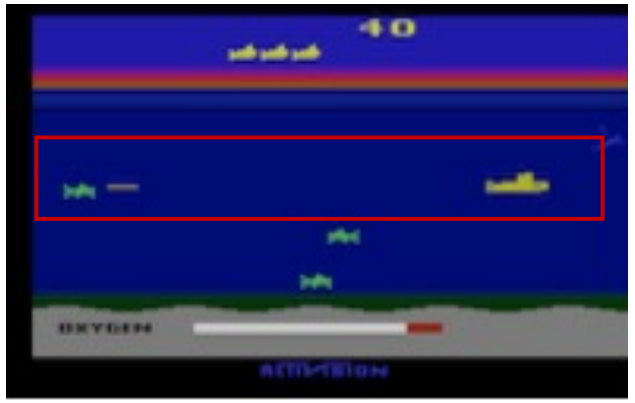
05 Experiments

상황에 따른 Q값의 변화를 통해 모델이 상황을 올바르게 이해하고 있음을 확인



A : 적 등장

점수 획득 기회 → 가치 상승

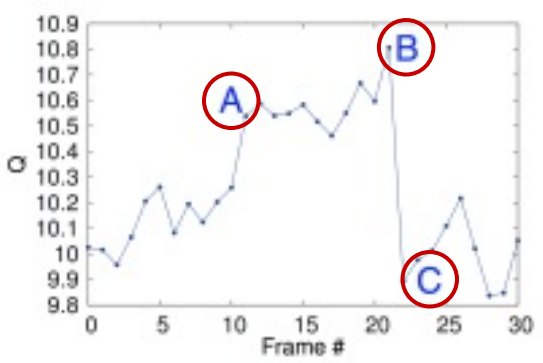


B : 공격 성공 직전

가치 최고값 도달

C : 처치 직후

상황 종료 → 가치 원위치



05 Experiments

비교군

- Random — 완전 무작위
- Sarsa
- Contingency

  → 다른 강화학습 알고리즘, 사전에 적, 배경과 같은 정보를 제공
- Human — 실제 인간의 플레이

- HNeat Best
- Hneat Pixel

  → 다른 방식(진화 알고리즘), 일반화가 약해 단일 최고 기록만으로 비교

05 Experiments

다른 강화학습 알고리즘과 달리 사전 지식이 전혀 없는 상황에서도 우수한 성능을 보였으며,

다른 접근법(유전 알고리즘)과 비교했을 때도 높은 성능과 강한 일반화 능력을 보임

	B. Rider	Breakout	Enduro	Pong	Q*bert	Seaquest	S. Invaders
Random	354	1.2	0	−20.4	157	110	179
Sarsa [3]	996	5.2	129	−19	614	665	271
Contingency [4]	1743	6	159	−17	960	723	268
DQN	**4092**	**168**	**470**	**20**	**1952**	**1705**	**581**
Human	7456	31	368	−3	18900	28010	3690
HNeat Best [8]	3616	52	106	19	1800	920	**1720**
HNeat Pixel [8]	1332	4	91	−16	1325	800	1145
DQN Best	**5184**	**225**	**661**	**21**	**4500**	**1740**	1075

Table 1: The upper table compares average total reward for various learning methods by running an ϵ-greedy policy with $\epsilon = 0.05$ for a fixed number of steps. The lower table reports results of the single best performing episode for HNeat and DQN. HNeat produces deterministic policies that always get the same score while DQN used an ϵ-greedy policy with $\epsilon = 0.05$.

06 Conclusion

- 사전 정보 없이 Raw pixel(화면)만으로도 게임을 학습
- 딥러닝과 강화학습을 안정적으로 학습시키는 방법을 제안
- 별다른 튜닝 없이 기존 방법보다 우수한 성능을 달성

NOVA

AI / Data Engineering Track

Thank you

숭실대학교 전기공학부 학술 소모임 NOVA
발표자 : 우상윤