Research Trend of Deep Reinforcement Learning

2023. 02. 24

발표자: 이영재

발표자 소개

- 이름: 이영재 (Young Jae Lee)
  - Data Mining & Quality Analytics Lab
  - Ph.D. Student (2019.03 ~ Present)
  - 지도 교수: 김성범 교수님
- 연구 분야
  - Deep Reinforcement Learning
  - Self/Semi-Supervised Learning
- 연락망
  - E-mail: jae601@korea.ac.kr

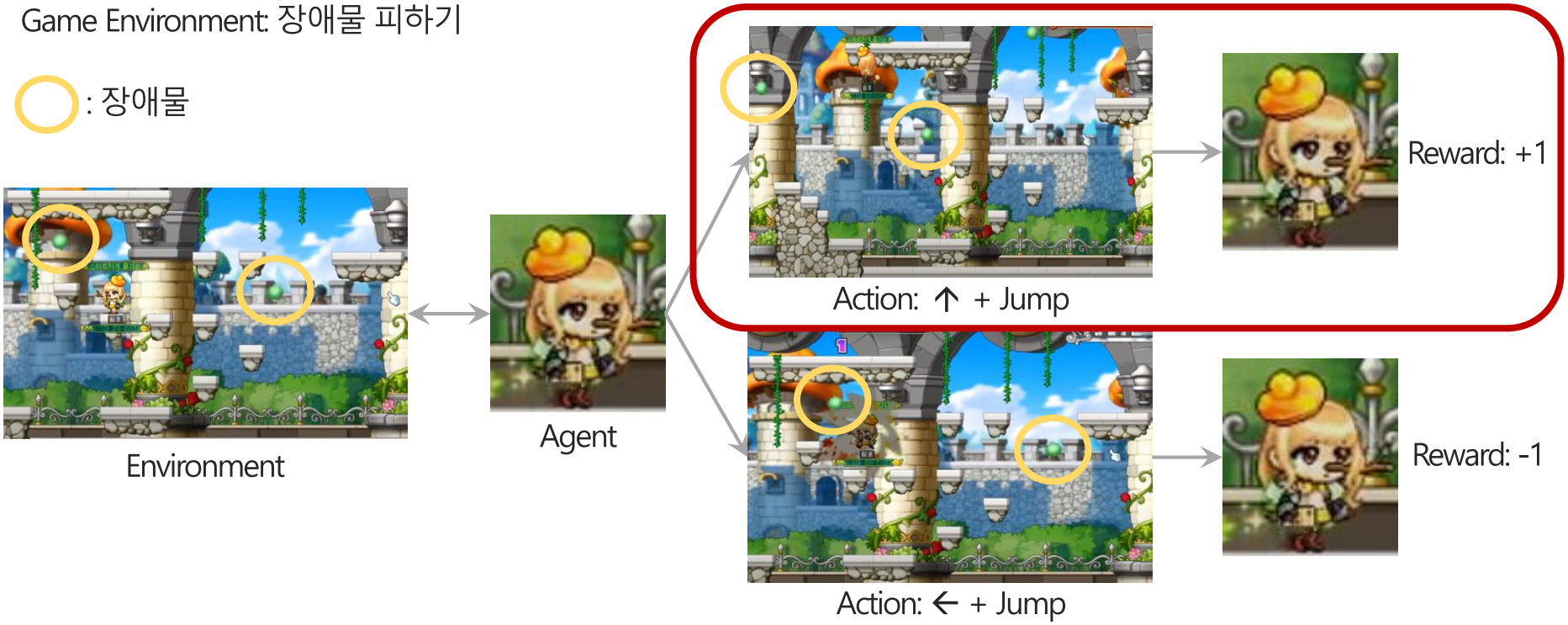
목차

- Introduction
- Deep Reinforcement Learning Research Trends
  - ✓ Representation Learning for Reinforcement Learning
  - ✓ Generalization in Reinforcement Learning
  - ✓ Offline Reinforcement Learning
  - ✓ Multi-Task Reinforcement Learning
  - ✓ Multi-Agent Reinforcement Learning
- Conclusion

Introduction

❖ Example of Reinforcement Learning

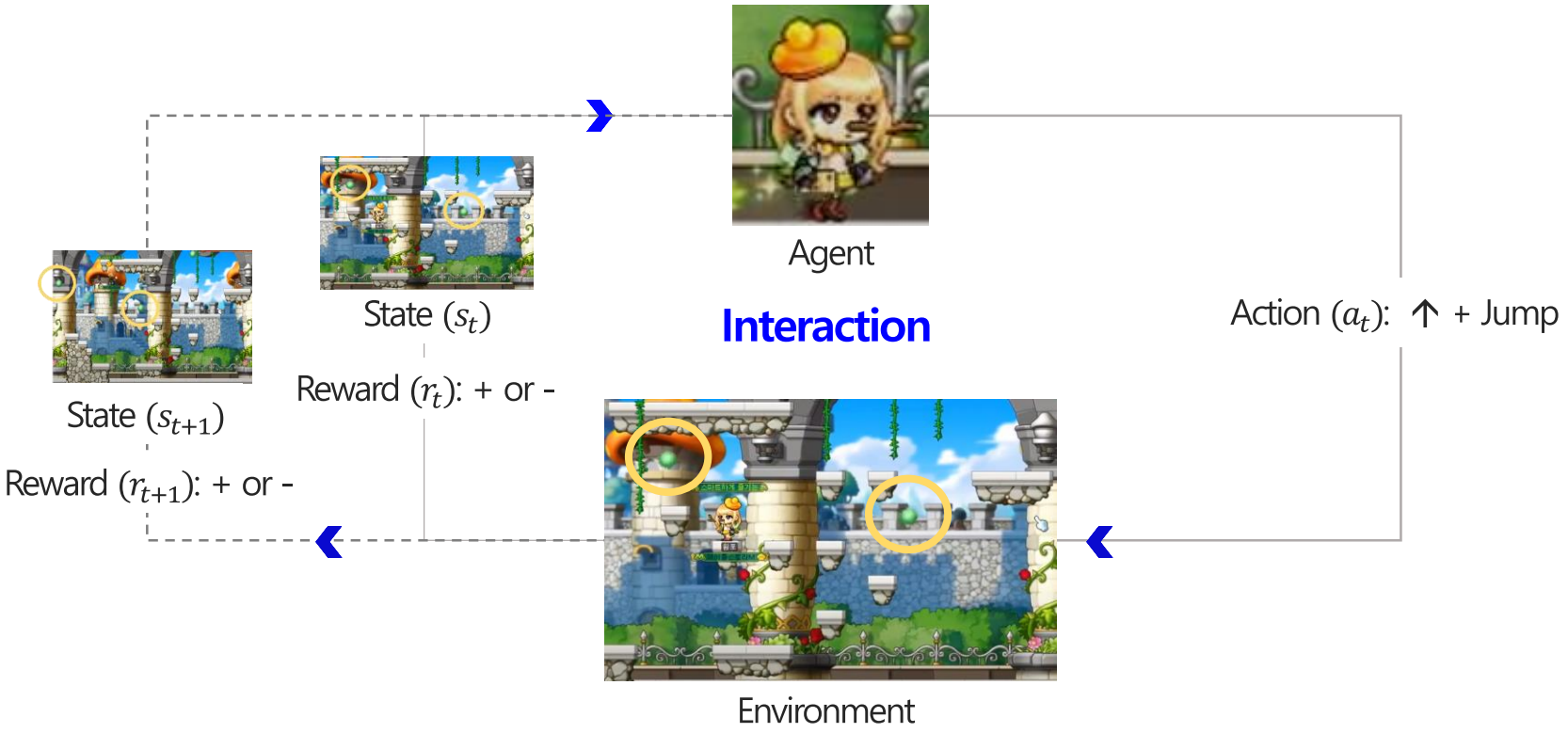
- 게임 환경: 장애물을 피하며 목적을 달성
- Agent 목표: 좌·우로 날아다니는 장애물을 피하며 목적지까지 도달하는 것



Introduction

❖ Definition of Reinforcement Learning

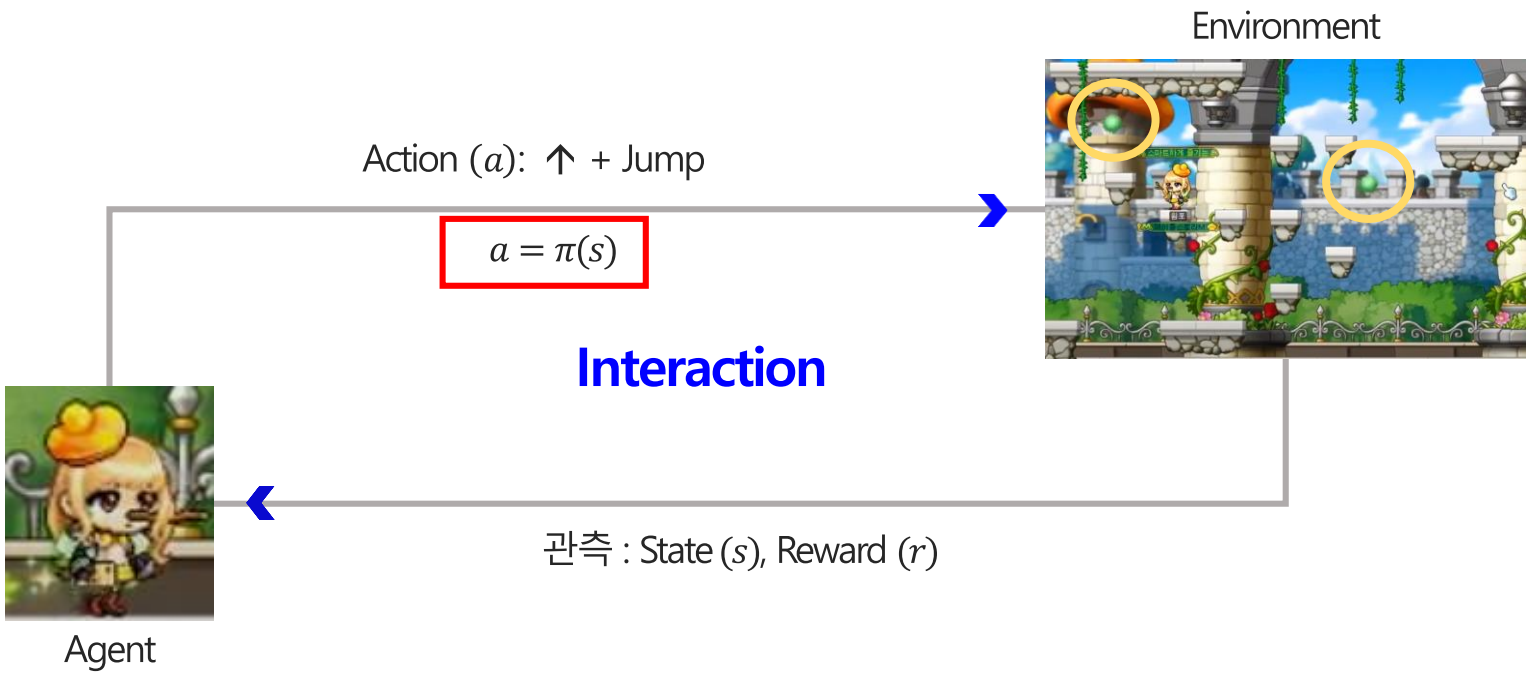
- 환경(Environment)과 상호작용(Interaction)하며 목표를 달성하는 에이전트(Agent)를 다루는 학습 방법



Introduction

❖ The Goal of Reinforcement Learning

- 미래에 받을 보상의 합을 최대화하는 정책(Policy) $\boldsymbol{\pi(a|s)}$를 찾는 것
- $\boldsymbol{\pi(a|s)}$는 에이전트의 행동 함수로써 State (s)에서 취할 Action (a)을 알려줌



Introduction

❖ Reinforcement Learning vs. Other Learning

- 공통점: Action (a)이나 레이블(y), 사전에 정의한 Task와 같은 것을 예측
- 차이점: **상호작용(Interaction)하는 환경(Environment)**의 존재 여부

	Reinforcement Learning	Supervised Learning	Contrastive Learning
Training Data	$S, A, R, S, A, R, S, A \ldots$	(X, Y)	(X)
Model	$\pi(a\|s)$	$\hat{\mathrm{Y}} = F(\mathrm{X})$	$Self\left(\hat{\mathrm{Y}} = F(\mathrm{X})\right)$
Objective	$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$	$\left(\mathrm{Y} - \hat{\mathrm{Y}}\right)^2$	$\mathcal{L}_i = -log \frac{exp\left(\frac{i \cdot i^+}{\tau}\right)}{exp\left(\frac{i \cdot i^+}{\tau}\right) + \sum_{k \notin \{i, i^+\}} exp\left(\frac{i \cdot k}{\tau}\right)}$

Introduction

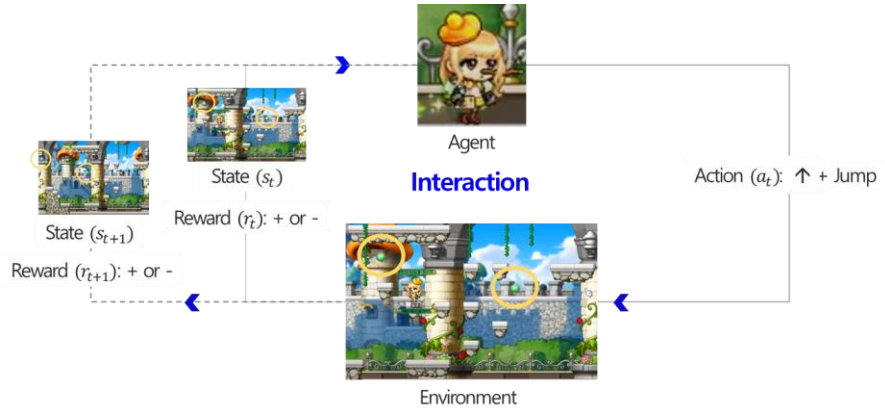
	Reinforcement Learning
Training Data	$S,A,R,S,A,R,S,A\ldots$
Model	$\pi(a\|s)$
Objective	$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

❖ Training Data

- Markov Decision Process (MDP)
  - ✓ 순차적인 의사결정 문제를 풀기 위한 프로세스
  - ✓ 강화학습에서 사용하는 모든 환경(Environment)은 MDP로 구성되어 있음
  - ✓ Markov Property: 미래는 과거 상태가 아닌 오직 현재 상태에만 의존

$$(\mathbf{P}[S_{t+1}|S_t] = \mathbf{P}[S_{t+1}|S_1, S_2, \ldots, S_t])$$

- 5가지 원소 $< S, A, P, R, \gamma >$
  - ✓ S –State의 집합
  - ✓ A – Action의 집합
  - ✓ P – 전이 확률 함수
  - ✓ R – Reward 함수
  - ✓ γ – Discount Factor (할인율)



- **Episode: $< s_0, a_0, r_1, s_1, a_1, r_2, s_2, a_2, r_3, s_3, \ldots, s_T >$, total time step: T**

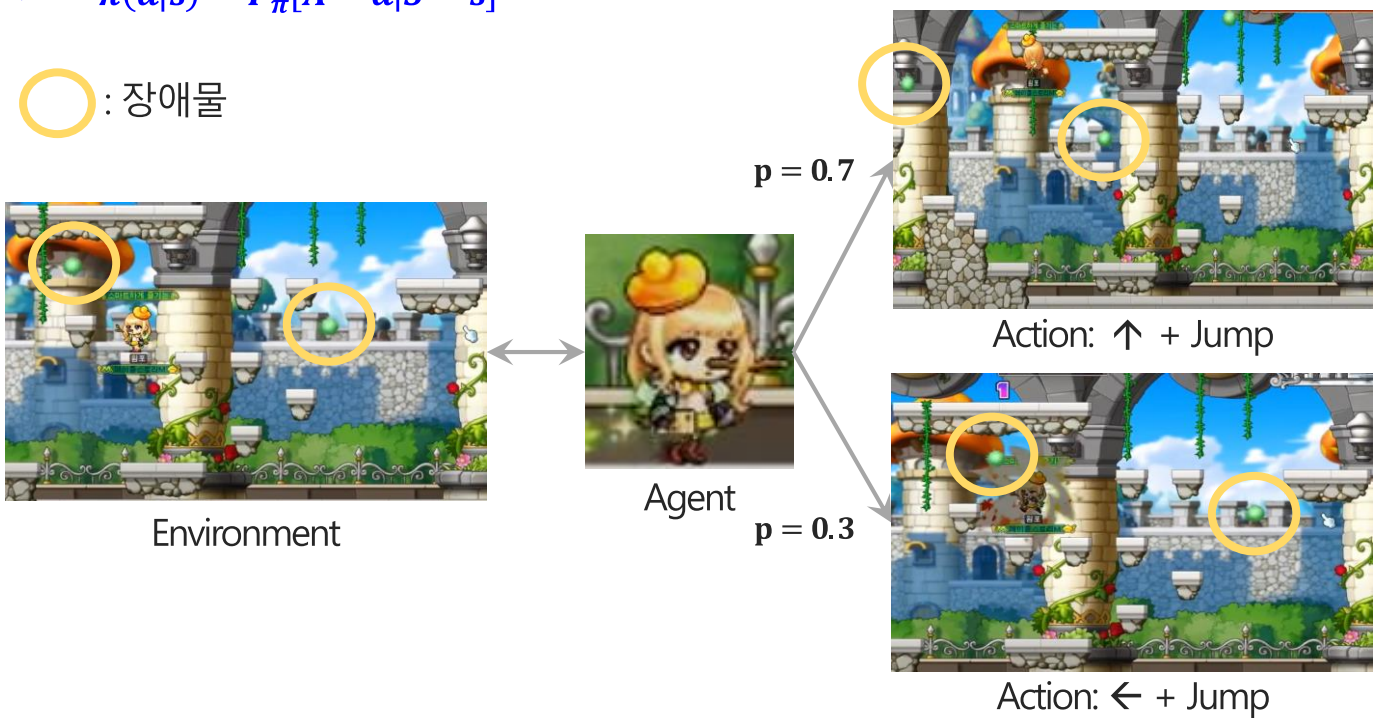
Introduction

	Reinforcement Learning
Training Data	$S,A,R,S,A,R,S,A\ldots$
Model	$\pi(a\|s)$
Objective	$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

❖ Model $\pi(a|s)$

- MDP로 구성된 환경(Environment)의 설명자 역할
- 정책(Policy)이라고 부르며 에이전트(Agent)의 행동 함수
- 주어진 환경의 State (s)에서 취할 Action (a)을 알려주는 역할

✓ $\boldsymbol{\pi(a|s) = P_\pi[A = a|S = s]}$



Introduction

	Reinforcement Learning
Training Data	$S,A,R,S,A,R,S,A\ldots$
Model	$\pi(a\|s)$
Objective	$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

❖ Bellman Equation

- Value Function의 반환값: $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

State-Value Function (상태 가치 함수)

✓ $v_\pi(s) = E_\pi[G_t|S_t = s]$

$= E_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots |S_t = s]$

$= E_\pi[R_{t+1} + \gamma(R_{t+2} + \gamma R_{t+3} + \cdots)|S_t = s]$

$= E_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s]$

$= E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s]$

State가 얼마나 좋은지 알려줌

Action-Value Function (행동 가치 함수)

✓ $q_\pi(s,a) = E_\pi[G_t|S_t = s, A_t = a]$

$= E_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots |S_t = s, A_t = a]$

$= E_\pi[R_{t+1} + \gamma(R_{t+2} + \gamma R_{t+3} + \cdots)|S_t = s, A_t = a]$

$= E_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s, A_t = a]$

$= E_\pi[R_{t+1} + \gamma q_\pi(s_{t+1}, a)|S_t = s, A_t = a]$

주어진 State에서 취한 Action이 얼마나 좋은지 알려줌

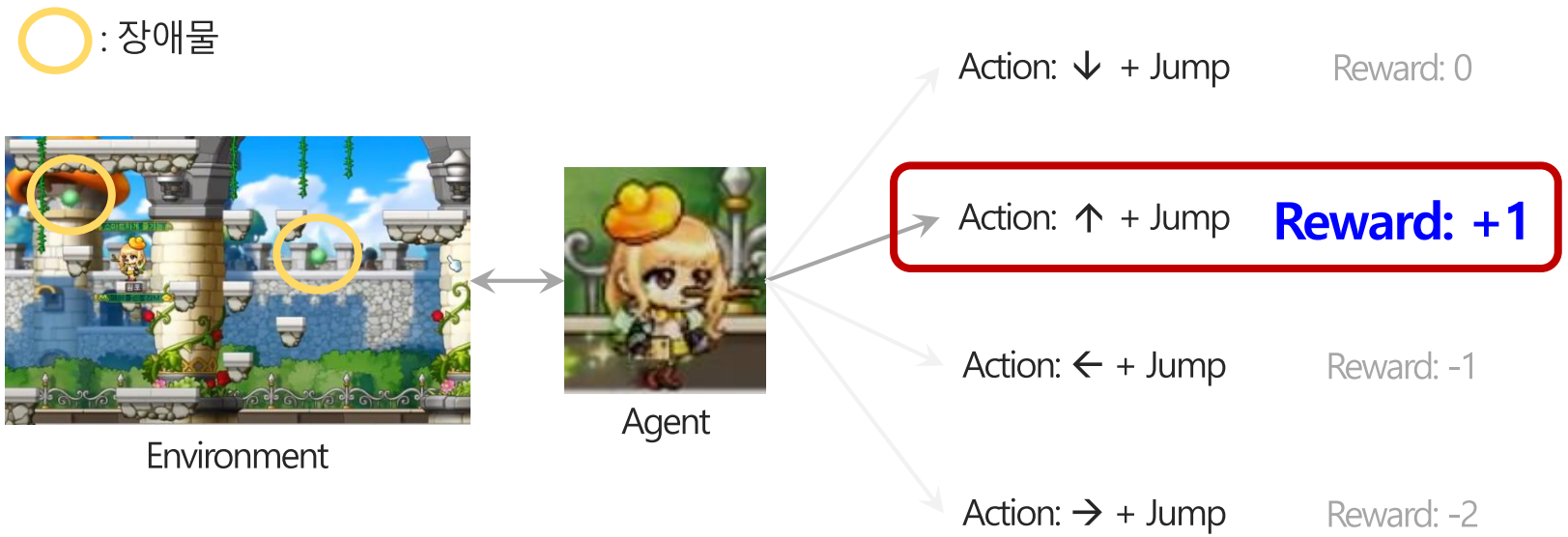
Introduction

	Reinforcement Learning
Training Data	$S,A,R,S,A,R,S,A\ldots$
Model	$\pi(a\|s)$
Objective	$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

❖ Q-Learning

- 가치(Value) 기반의 학습
- 에이전트가 특정 State (s)에서 자신이 취할 수 있는 Action (a)들 중 **이득이 되는 방향으로 학습**하는 방법

✓ $\mathbf{Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[R_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)]}$



Introduction

	Reinforcement Learning
Training Data	$S,A,R,S,A,R,S,A\ldots$
Model	$\pi(a\|s)$
Objective	$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

❖ Policy Gradient

- 정책(Policy) 기반의 학습
- 파라미터 θ에 대한 정책 $\pi(a|s;\theta)$를 직접 학습하는 방법

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

✓ $\boldsymbol{\theta' = \theta + \alpha \nabla_\theta J(\theta)}$, $\boldsymbol{\nabla_\theta J(\theta)}$ = Policy Gradient

차이점

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

$$\approx E_\tau[\nabla_\theta \sum_{t=0}^{T-1} log\pi_\theta(a_t|s_t) G_t]$$

$$where\ G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Action-Value Function (행동 가치 함수)

✓ $q_\pi(s,a) = E_\pi[G_t|S_t = s, A_t = a]$

$= E_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots|S_t = s, A_t = a]$

$= E_\pi[R_{t+1} + \gamma(R_{t+2} + \gamma R_{t+3} + \cdots)|S_t = s, A_t = a]$

$= E_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s, A_t = a]$

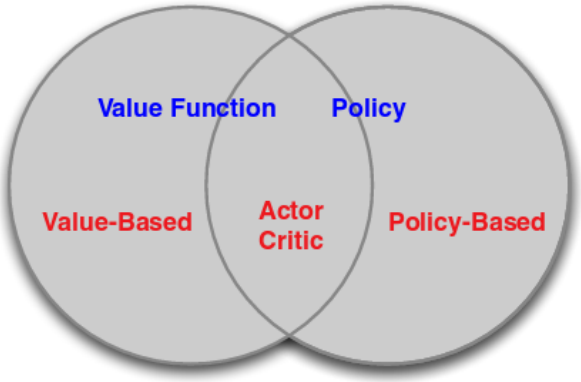
$= E_\pi[R_{t+1} + \gamma q_\pi(s_{t+1}, a)|S_t = s, A_t = a]$

Introduction

	Reinforcement Learning
Training Data	$S,A,R,S,A,R,S,A\ldots$
Model	$\pi(a\|s)$
Objective	$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

❖ Actor-Critic Methods

- 가치(Value)와 정책(Policy) 모두 고려한 학습



Policy Gradient

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

$$\approx E_\tau[\nabla_\theta \sum_{t=0}^{T-1} log\pi_\theta(a_t|s_t) G_t]$$

$$where\ G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Action-Value Function (행동 가치 함수)

$$E_\tau[\nabla_\theta \sum_{t=0}^{T-1} log\pi_\theta(a_t|s_t) G_t]$$

$$\approx \sum_{t=0}^{T-1} E_{s_o,a_0,\ldots,s_t,a_t}[\nabla_\theta log\pi_\theta(a_t|s_t)] E_{r_{t+1},s_{t+1},\ldots,s_T,r_T}[G_t]$$

$$= \sum_{t=0}^{T-1} E_{s_o,a_0,\ldots,s_t,a_t}[\nabla_\theta log\pi_\theta(a_t|s_t)] Q_{\pi_\theta}(s_t,a_t)$$

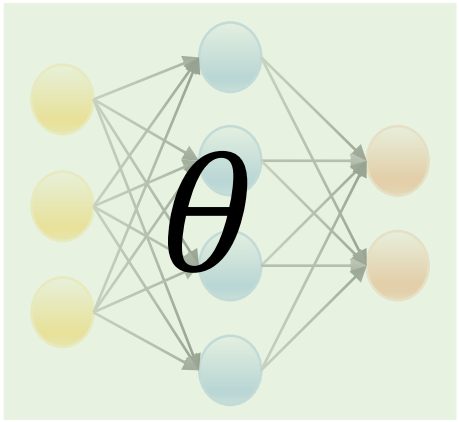
Introduction

	Reinforcement Learning
Training Data	$S,A,R,S,A,R,S,A\ldots$
Model	$\pi(a\|s)$
Objective	$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

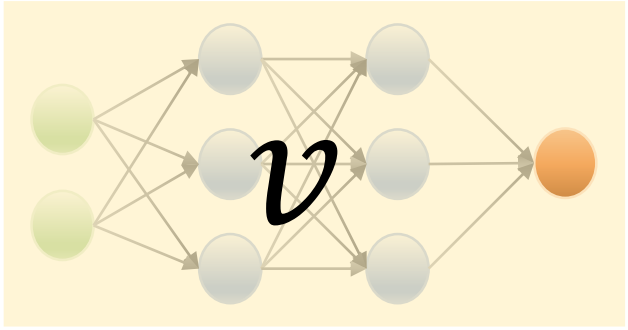
❖ Actor-Critic Methods

- 가치(Value)와 정책(Policy) 모두 고려한 학습

Actor



Critic



$$\theta \leftarrow \theta + \nabla_\theta log\pi_\theta(a_t|s_t)\left(r_{t+1} + \gamma Q_v(s_{t+1}, a_{t+1}) - Q_v(s_t, a_t)\right)$$

score

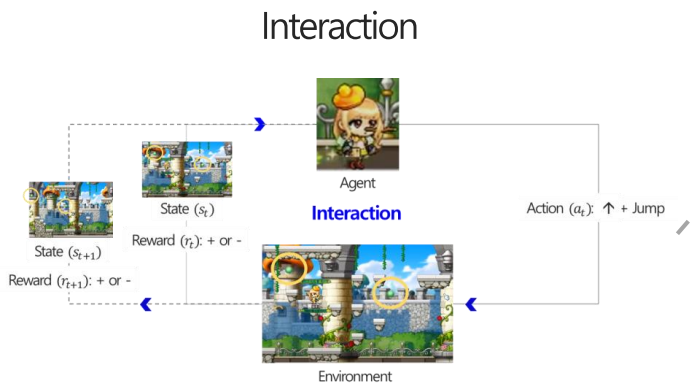
- ✓ Actor에서 Action (a)을 선택
- ✓ Critic은 선택된 Action (a)을 평가하는 역할
- ✓ Critic에서 파라미터 v를 업데이트
- ✓ Critic이 제안하는 방향으로 θ를 업데이트

Introduction

	Reinforcement Learning
Training Data	$S,A,R,S,A,R,S,A\ldots$
Model	$\pi(a\|s)$
Objective	$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

❖ Data Efficiency (=Sample Efficiency)

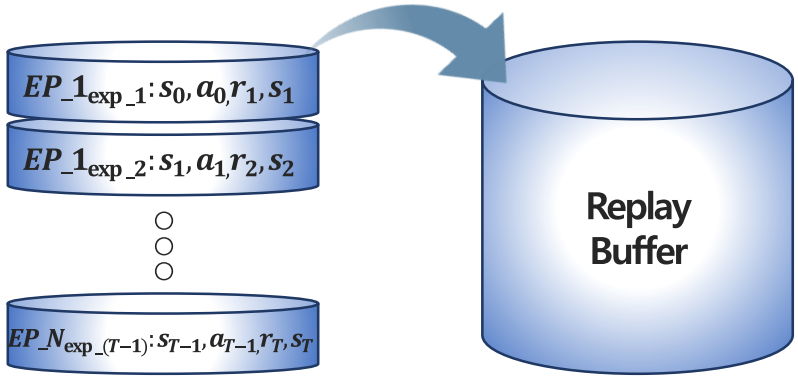
- Training Data (Remind)
  - ✓ **Episode: $< s_0, a_0, r_1, s_1, a_1, r_2, s_2, a_2, r_3, s_3, \ldots, s_T >$, total time step: T**
- Experience Replay (Replay Buffer, Replay Memory)
  - ✓ 각 Timestep별로 얻은 Experience $(s_t, a_t, r_{t+1}, s_{t+1})$들을 저장하는 공간



Episode_1: $< s_0, a_0, r_1, s_1, a_1, r_2, s_2, a_2, r_3, s_3, \ldots, s_T >$

Episode_2: $< s_0, a_0, r_1, s_1, a_1, r_2, s_2, a_2, r_3, s_3, \ldots, s_T >$

Episode_N: $< s_0, a_0, r_1, s_1, a_1, r_2, s_2, a_2, r_3, s_3, \ldots, s_T >$

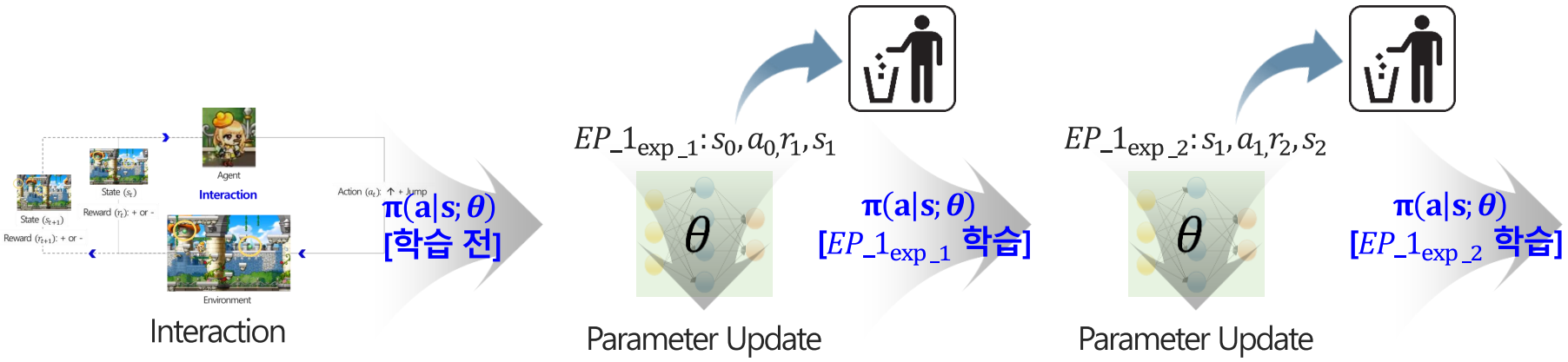


Introduction

	Reinforcement Learning
Training Data	$S,A,R,S,A,R,S,A\ldots$
Model	$\pi(a\|s)$
Objective	$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

❖ **On-Policy** vs. Off-Policy

- 정책(Policy)이 Experience를 얻음과 동시에 학습을 진행
  - ✓ Experience는 현재 정책(학습 전)으로부터 얻음
  - ✓ 단점 1: Experience를 가지고 정책을 업데이트하기 때문에 Sample 분포 자체가 현재 정책에 의존적(Sample Dependent)
  - ✓ 단점 2: 정책을 업데이트한 후, 이전 Experience는 업데이트 된 정책과 다르기 때문에 사용 불가능
  - ✓ 단점 3: 정책이 발산하거나 Local Optimal에 수렴할 가능성이 커짐

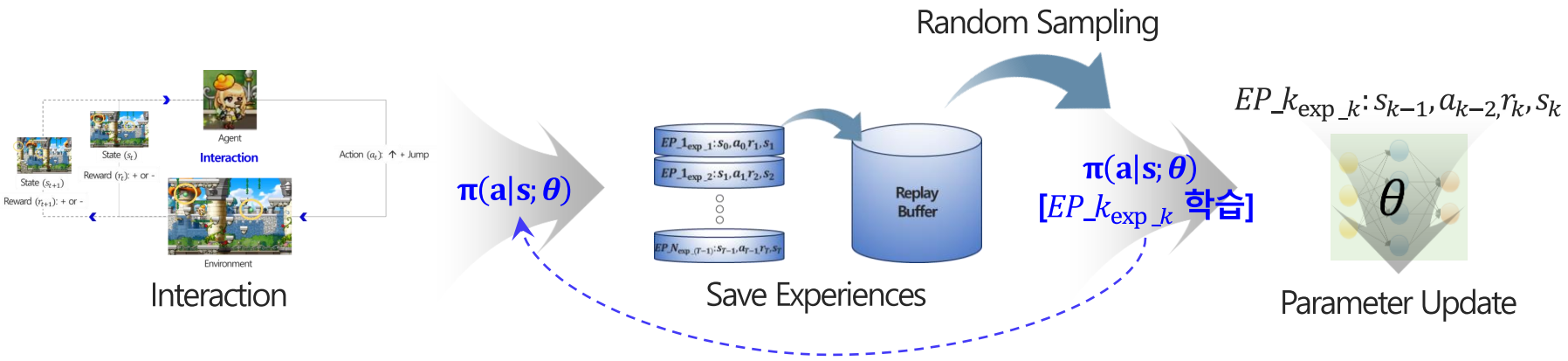


Introduction

	Reinforcement Learning
Training Data	$S,A,R,S,A,R,S,A\ldots$
Model	$\pi(a\|s)$
Objective	$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$

❖ On-Policy vs. **Off-Policy**

- Behavior 정책(Policy)과 Target 정책이 분리되어 진행
  - ✓ Behavior 정책은 State (s_t)에서 Action (a_t)를 취한 후 State ($\hat{s}_{t+1}$)를 예측하는 역할(Continue)
  - ✓ Target 정책은 예측한 State ($\hat{s}_{t+1}$)에서 Action ($\hat{a}_{t+1}$)을 예측하는 역할(Fixed Time)
  - ✓ 현재 학습하는 정책이 과거에 경험했던 Experience로도 학습에 사용 가능(**Experience Replay**)
  - ✓ Sample들을 사용하고 버리는 것과 달리 Random Sampling하여 Sample의 효율성을 높임
  - ✓ 정책이 Local Optimal에 수렴하거나 발산하는 경우를 방지



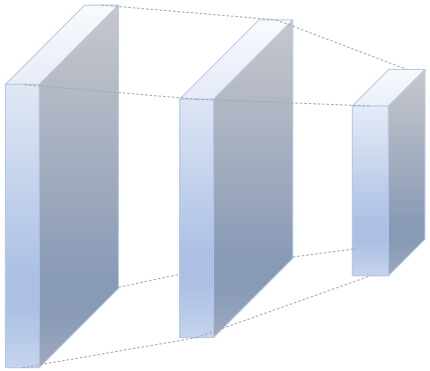
Introduction

❖ Deep Reinforcement Learning

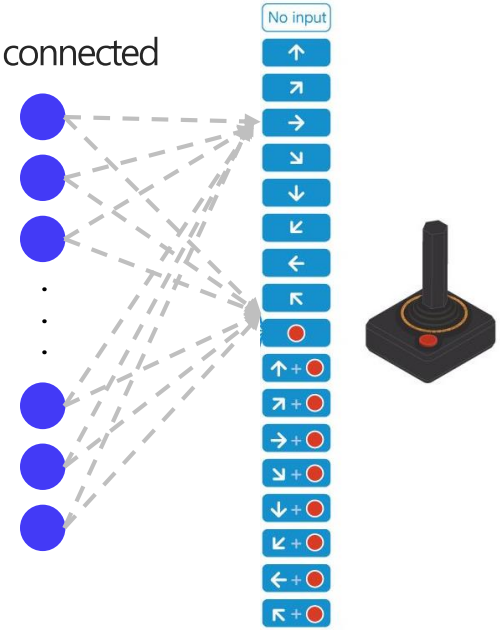
- 실제 환경은 매우 복잡한 환경을 띔
  - ✓ Ex) StarCraft II의 경우 state (s)와 action (a)의 조합이 억 단위 이상
- 이를 해결하기 위해 Deep Reinforcement Learning (DRL)이 탄생
  - ✓ Google DeepMind에서 연구한 Deep Q-Network (DQN)이 DRL의 시작이 됨
  - ✓ Deep Learning이 발전하면서 강화학습과 결합하여 수많은 연구를 수행할 수 있게 됨

StarCraft II

State 정보를 합성곱 연산



Fully connected



Deep Reinforcement Learning Research Trends

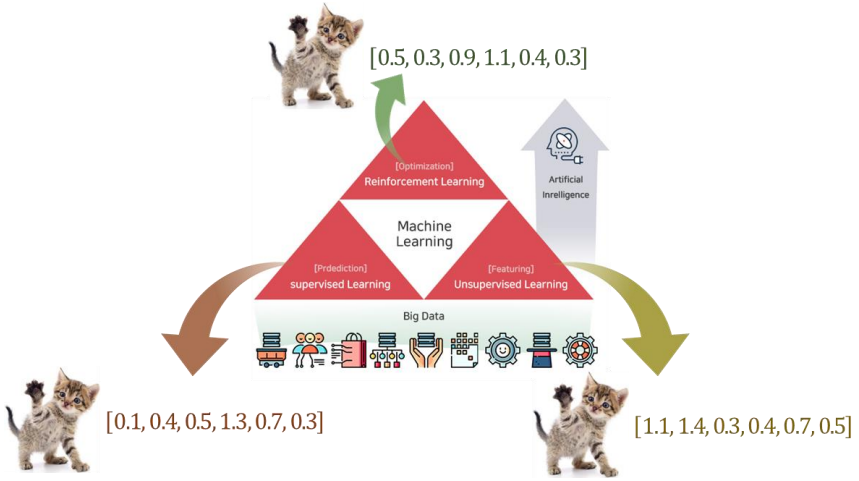
❖ Research Trends

1. 데이터 또는 샘플의 효율성을 향상시키기 위한 연구
2. 서로 다른 환경에서의 일반화 성능을 향상시키기 위한 연구
3. 에이전트와 환경과의 상호작용 없이 정책을 학습하기 위한 연구
4. 하나의 글로벌 정책 학습하여 여러 환경의 문제를 해결하기 위한 연구
5. 여러 에이전트가 협동하여 문제를 해결하기 위한 연구

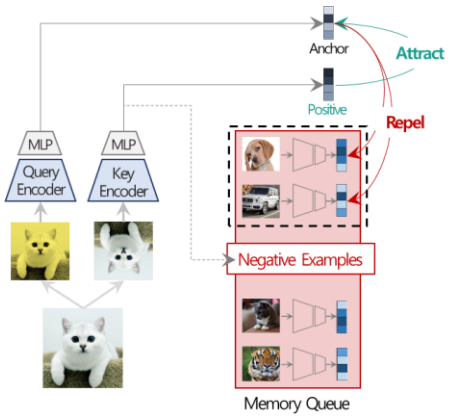
Deep Reinforcement Learning Research Trends

- Representation Learning for Reinforcement Learning
  - **목표:** 데이터 효율성을 향상시키기 위해 Self-Supervised Learning을 강화학습(RL)과 결합
  - Self-Supervised Learning (SSL)은 강화학습 환경의 상태에 대한 좋은 표현을 학습
    - ✓ Representation Learning: 이미지, 텍스트 등과 같은 데이터를 숫자 형태로 기술하도록 학습
    - ✓ Pretext Tasks, Contrastive Learning, Non-Contrastive Learning…

Representation Learning



Momentum Contrast (MoCo)



$$\mathcal{L}_i = -\log \frac{\exp\left(\frac{i \cdot i^+}{\tau}\right)}{\exp\left(\frac{i \cdot i^+}{\tau}\right) + \sum_{k \notin \{i, i^+\}} \exp\left(\frac{i \cdot k}{\tau}\right)}$$

i: Anchor
i^+: Positive
k: Negative Examples

Momentum Update
(Key Encoder)

$$\theta_{key} \leftarrow m\theta_{key} + (1-m)\theta_{query}$$

Deep Reinforcement Learning Research Trends

- Representation Learning for Reinforcement Learning
  1. CURL: Contrastive Unsupervised Representations for Reinforcement Learning (2020, ICML)

CURL: Contrastive Unsupervised Representations for Reinforcement Learning

Aravind Srinivas* [1] Michael Laskin* [1] Pieter Abbeel [1]

Abstract

We present CURL: Contrastive Unsupervised Representations for Reinforcement Learning. CURL extracts high-level features from raw pixels using contrastive learning and performs off-policy control on top of the extracted features. CURL outperforms prior pixel-based methods, both model-based and model-free, on complex tasks in the DeepMind Control Suite and Atari Games showing 1.9x and 1.2x performance gains at the 100K environment and interaction steps benchmarks respectively. On the DeepMind Control Suite, CURL is the first image-based algorithm to nearly match the sample-efficiency of methods that use state-based features. Our code is open-sourced and available at https://www.github.com/MishaLaskin/curl.

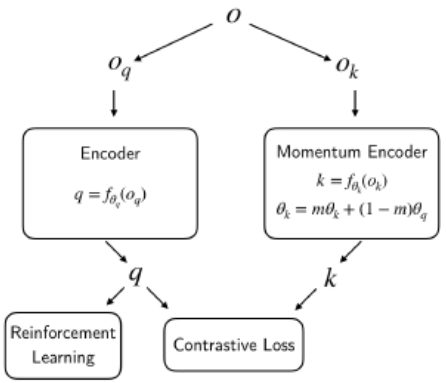


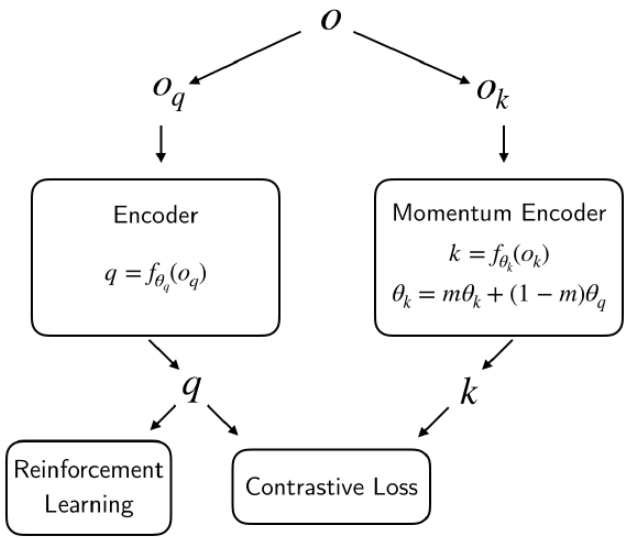
Figure 1. Contrastive Unsupervised Representations for Reinforcement Learning (CURL) combines instance contrastive learning and reinforcement learning. CURL trains a visual

Deep Reinforcement Learning Research Trends

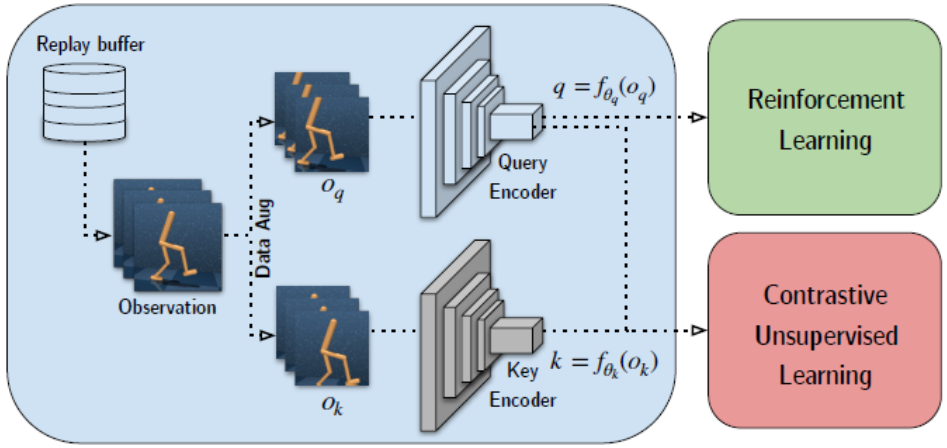
❖ Representation Learning for Reinforcement Learning

1. CURL: Contrastive Unsupervised Representations for Reinforcement Learning (2020, ICML)
   - ✓ 데이터 효율성을 향상시키기 위해 강화학습과 Self-Supervised Learning을 결합한 연구
   - ✓ 강화학습 방법론: 가치(Value) 기반 학습 방식인 Efficient Rainbow 사용(Model-Free, Off-Policy)
   - ✓ Self-Supervised Learning: MoCo 사용(Contrastive Learning)

MoCo



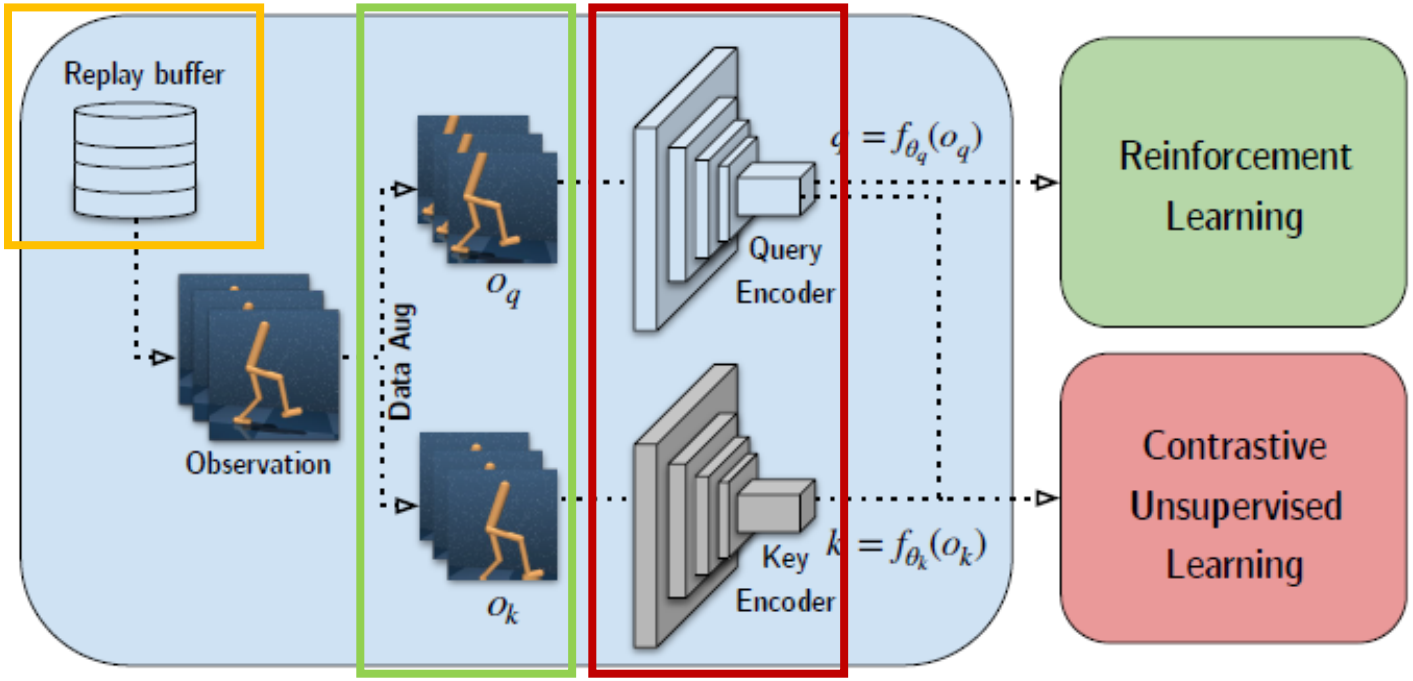
CURL



Deep Reinforcement Learning Research Trends

❖ Representation Learning for Reinforcement Learning

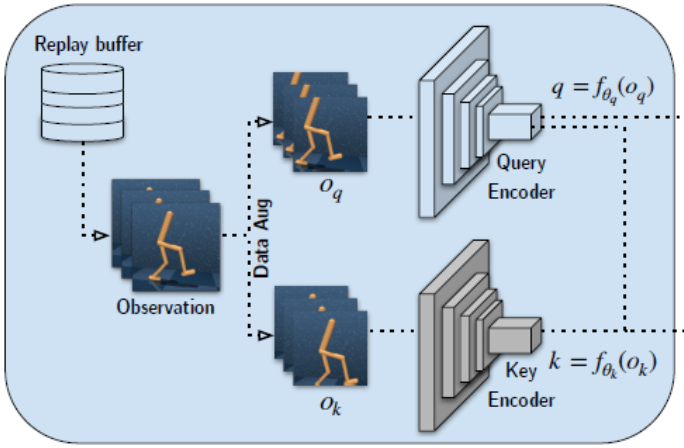
1. CURL: Contrastive Unsupervised Representations for Reinforcement Learning (2020, ICML)
   - ✓ Query Encoder와 Key Encoder로 구성
   - ✓ 각 Timestep별 Experience를 Replay Buffer에 저장
   - ✓ Mini Batch 사이즈만큼 샘플링하고 서로 다른 두개의 Data Augmentation 적용



Deep Reinforcement Learning Research Trends

❖ Representation Learning for Reinforcement Learning

1. CURL: Contrastive Unsupervised Representations for Reinforcement Learning (2020, ICML)
   - ✓ 동일 이미지로부터 나온 두 이미지는 **Positive Pair**, 이외에는 **Negative Examples**로 정의
   - ✓ Query Encoder에서 추출한 Feature Vector는 강화학습과 SSL에 사용
   - ✓ 강화학습으로부터 계산된 Loss와 SSL로부터 계산된 Loss에 λ를 곱한 Loss를 더하여 학습 수행
   - ✓ 강화학습과 SSL을 동시에 수행하는 **One-Stage 방식**(Not Transfer Learning)



Reinforcement Learning

Contrastive Unsupervised Learning

RL Loss Function

$$\lambda * \mathcal{L}_i = -log \frac{exp\left(\frac{i \cdot i^+}{\tau}\right)}{exp\left(\frac{i \cdot i^+}{\tau}\right) + \sum_{k \notin \{i, i^+\}} exp\left(\frac{i \cdot k}{\tau}\right)}$$

Momentum Update (Key Encoder)

$$\theta_{key} \leftarrow m\theta_{key} + (1-m)\theta_{query}$$

Deep Reinforcement Learning Research Trends

❖ Representation Learning for Reinforcement Learning

1. CURL: Contrastive Unsupervised Representations for Reinforcement Learning (2020, ICML)
   - ✓ DMControl과 26개의 Atari 2600 게임을 사용하여 성능 평가
   - ✓ DMControl은 100만, 500만 / Atari는 10만 Timestep으로 제한하여 데이터 효율성 향상 검증

DMControl

500K step scores	CURL	PlaNet	Dreamer	SAC+AE	SLACv1	Pixel SAC	State SAC
Finger, spin	**926 ± 45**	561 ± 284	796 ± 183	884 ± 128	673 ± 92	179 ± 166	923 ± 21
Cartpole, swingup	**841 ± 45**	475 ± 71	762 ± 27	735 ± 63	-	419 ± 40	848 ± 15
Reacher, easy	**929 ± 44**	210 ± 390	793 ± 164	627 ± 58	-	145 ± 30	923 ± 24
Cheetah, run	518 ± 28	305 ± 131	570 ± 253	550 ± 34	**640 ± 19**	197 ± 15	795 ± 30
Walker, walk	**902 ± 43**	351 ± 58	897 ± 49	847 ± 48	842 ± 51	42 ± 12	948 ± 54
Ball in cup, catch	**959 ± 27**	460 ± 380	879 ± 87	794± 58	852 ± 71	312± 63	974 ± 33
100K step scores							
Finger, spin	**767 ± 56**	136 ± 216	341 ± 70	740 ± 64	693 ± 141	179 ± 66	811±46
Cartpole, swingup	**582±146**	297±39	326±27	311±11	-	419±40	835±22
Reacher, easy	**538±233**	20±50	314±155	274±14	-	145±30	746±25
Cheetah, run	299 ±48	138±88	235± 137	267±24	**319±56**	197±15	616±18
Walker, walk	**403±24**	224±48	277±12	394±22	361±73	42±12	891±82
Ball in cup, catch	**769 ± 43**	0 ± 0	246 ± 174	391± 82	512 ± 110	312± 63	746±91

Atari 2600

Game	Human	Random	Rainbow	SimPLe	OTRainbow	Eff. Rainbow	CURL
Alien	7127.7	227.8	318.7	616.9	**824.7**	739.9	558.2
Amidar	1719.5	5.8	32.5	88.0	82.8	**188.6**	142.1
Assault	742.0	222.4	231	527.2	351.9	431.2	**600.6**
Asterix	8503.3	210.0	243.6	**1128.3**	628.5	470.8	734.5
Bank Heist	753.1	14.2	15.55	34.2	**182.1**	51.0	131.6
Battle Zone	37187.5	2360.0	2360.0	5184.4	4060.6	10124.6	**14870.0**
Boxing	12.1	0.1	-24.8	**9.1**	2.5	0.2	1.2
Breakout	30.5	1.7	1.2	**16.4**	9.84	1.9	4.9
Chopper Command	7387.8	811.0	120.0	**1246.9**	1033.33	861.8	1058.5
crazy_climber	35829.4	10780.5	2254.5	**62583.6**	21327.8	16185.3	12146.5
demon_attack	1971.0	152.1	163.6	208.1	711.8	508.0	**817.6**
freeway	29.6	0.0	0.0	20.3	25.0	**27.9**	26.7
frostbite	4334.7	65.2	60.2	254.7	231.6	866.8	**1181.3**
gopher	2412.5	257.6	431.2	771.0	**778.0**	349.5	669.3
hero	30826.4	1027.0	487	2656.6	6458.8	**6857.0**	6279.3
jamesbond	302.8	29.0	47.4	125.3	112.3	301.6	**471.0**
kangaroo	3035.0	52.0	0.0	323.1	605.4	779.3	**872.5**
krull	2665.5	1598.0	1468	**4539.9**	3277.9	2851.5	4229.6
kung_fu_master	22736.3	258.5	0.	**17257.2**	5722.2	14346.1	14307.8
ms_pacman	6951.6	307.3	67	**1480.0**	941.9	1204.1	1465.5
Pong	14.6	-20.7	-20.6	**12.8**	1.3	-19.3	-16.5
Private eye	69571.3	24.9	0	58.3	100.0	97.8	**218.4**
Qbert	13455.0	163.9	123.46	**1288.8**	509.3	1152.9	1042.4
Road_Runner	7845.0	11.5	1588.46	5640.6	2696.7	**9600.0**	5661.0
seaquest	42054.7	68.4	131.69	**683.3**	286.92	354.1	384.5
Up_n_Down	11693.2	533.4	504.6	**3350.3**	2847.6	2877.4	2955.2

Deep Reinforcement Learning Research Trends

- Representation Learning for Reinforcement Learning
  2. Data-Efficient Reinforcement Learning with Self-Predictive Representations (2021, ICLR)

Data-Efficient Reinforcement Learning with Self-Predictive Representations

Max Schwarzer* Mila, Université de Montréal

Ankesh Anand* Mila, Université de Montréal, Microsoft Research

Rishab Goel Mila

R Devon Hjelm Microsoft Research, Mila, Université de Montréal

Aaron Courville Mila, Université de Montréal, CIFAR Fellow

Philip Bachman Microsoft Research

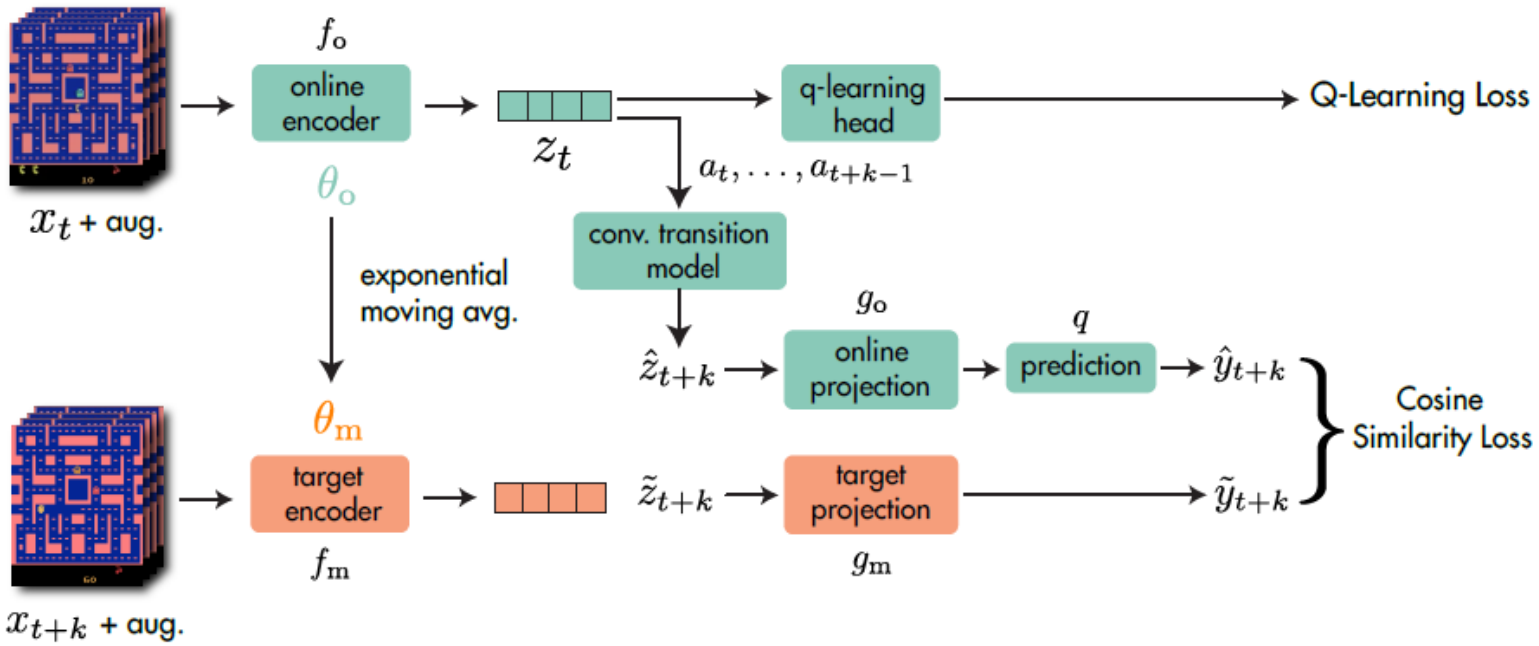
Abstract

While deep reinforcement learning excels at solving tasks where large amounts of data can be collected through virtually unlimited interaction with the environment, learning from limited interaction remains a key challenge. We posit that an agent can learn more efficiently if we augment reward maximization with self-supervised objectives based on structure in its visual input and sequential interaction with the environment. Our method, Self-Predictive Representations (SPR), trains an agent to predict its own latent state representations multiple steps into the future. We compute *target* representations for future states using an encoder which is an exponential moving average of the agent's parameters and we make predictions using a learned transition model. On its own, this future prediction objective outperforms prior methods for sample-efficient deep RL from pixels. We further improve performance by adding data augmentation to the future prediction loss, which forces the agent's representations to be consistent across multiple views of an observation. Our full self-supervised objective, which combines future prediction and data augmentation, achieves a median human-normalized score of 0.415 on Atari in a setting limited to 100k steps of environment interaction, which represents a 55% relative improvement over the previous state-of-the-art. Notably, even in this limited data regime, SPR exceeds expert human scores on 7 out of 26 games. The code associated with this work is available at https://github.com/mila-iqia/spr.

Deep Reinforcement Learning Research Trends

❖ Representation Learning for Reinforcement Learning

2. Data-Efficient Reinforcement Learning with Self-Predictive Representations (2021, ICLR)
   - ✓ 강화학습 방법론: 가치(Value) 기반 학습 방식인 Efficient Rainbow 사용(Model-Free, Off-Policy)
   - ✓ Self-Supervised Learning: BYOL 사용(Non-Contrastive Learning

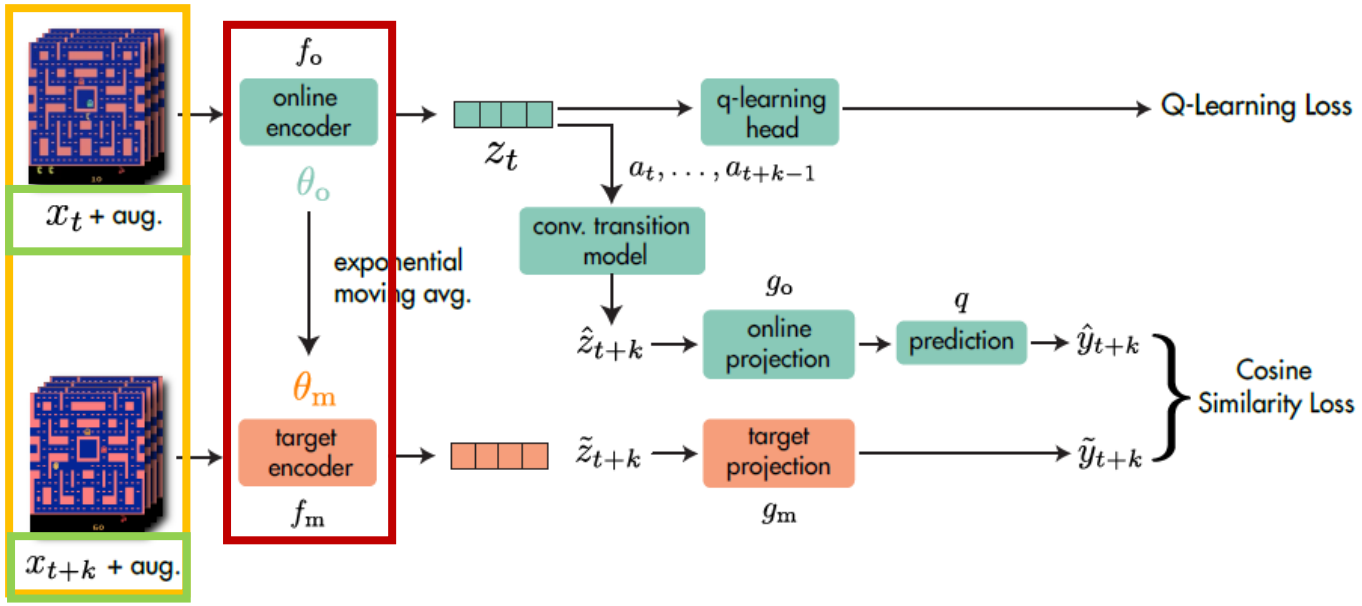


Deep Reinforcement Learning Research Trends

❖ Representation Learning for Reinforcement Learning

2. Data-Efficient Reinforcement Learning with Self-Predictive Representations (2021, ICLR)

- ✓ Online Encoder와 Target Encoder로 구성
- ✓ 각 Timestep별 Experience를 Replay Buffer에 저장 및 Mini Batch 사이즈만큼 샘플링
- ✓ 서로 다른 시점$(t, t+k)$의 이미지에 서로 다른 두개의 Data Augmentation 적용
- ✓ Data Augmentation으로부터 나온 두 이미지는 Positive Pair로 정의

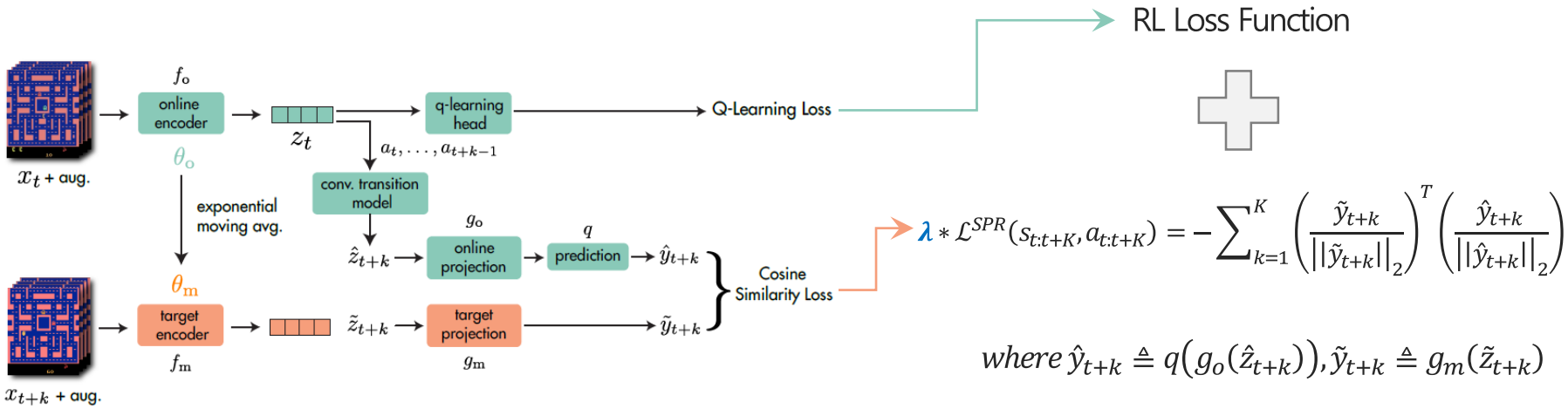


Deep Reinforcement Learning Research Trends

❖ Representation Learning for Reinforcement Learning

2. Data-Efficient Reinforcement Learning with Self-Predictive Representations (2021, ICLR)

- ✓ Online Encoder에서 추출한 Feature Vector (z_t)는 강화학습과 SSL에 사용
- ✓ Self-Predictive Representations (SPR) 기법을 제안하여 데이터 효율성 향상
- ✓ SPR은 transition model (h) 추가, 미래의 요약된 State ($\hat{z}_{t+k}$) 예측 및 k 반복하며 SSL Loss 계산
- ✓ $\hat{z}_{t+k} \triangleq h(\hat{z}_{t+k-1}, a_{t+k-1})$, a_t는 Mini Batch 내 존재하는 미래의 Action 정보



Deep Reinforcement Learning Research Trends

❖ Representation Learning for Reinforcement Learning

2. Data-Efficient Reinforcement Learning with Self-Predictive Representations (2021, ICLR)

✓ Atari 2600 26개 게임을 사용하여 성능 평가

✓ 10만 Timestep으로 제한하여 데이터 효율성 향상 검증

Game	Random	Human	SimPLe	DER	OTRainbow	CURL	DrQ	SPR (no Aug)	SPR
Alien	227.8	7127.7	616.9	739.9	824.7	558.2	771.2	**847.2**	801.5
Amidar	5.8	1719.5	88.0	**188.6**	82.8	142.1	102.8	142.7	176.3
Assault	222.4	742.0	527.2	431.2	351.9	600.6	452.4	**665.0**	571.0
Asterix	210.0	8503.3	**1128.3**	470.8	628.5	734.5	603.5	820.2	977.8
Bank Heist	14.2	753.1	34.2	51.0	182.1	131.6	168.9	**425.6**	380.9
BattleZone	2360.0	37187.5	5184.4	10124.6	4060.6	14870.0	12954.0	10738.0	**16651.0**
Boxing	0.1	12.1	9.1	0.2	2.5	1.2	6.0	12.7	**35.8**
Breakout	1.7	30.5	16.4	1.9	9.8	4.9	16.1	12.9	**17.1**
ChopperCommand	811.0	7387.8	**1246.9**	861.8	1033.3	1058.5	780.3	667.3	974.8
Crazy Climber	10780.5	35829.4	**62583.6**	16185.3	21327.8	12146.5	20516.5	43391.0	42923.6
Demon Attack	152.1	1971.0	208.1	508.0	711.8	817.6	**1113.4**	370.1	545.2
Freeway	0.0	29.6	20.3	**27.9**	25.0	26.7	9.8	16.1	24.4
Frostbite	65.2	4334.7	254.7	866.8	231.6	1181.3	331.1	1657.4	**1821.5**
Gopher	257.6	2412.5	771.0	349.5	**778.0**	669.3	636.3	774.5	715.2
Hero	1027.0	30826.4	2656.6	6857.0	6458.8	6279.3	3736.3	5707.4	**7019.2**
Jamesbond	29.0	302.8	125.3	301.6	112.3	**471.0**	236.0	367.2	365.4
Kangaroo	52.0	3035.0	323.1	779.3	605.4	872.5	940.6	1359.5	**3276.4**
Krull	1598.0	2665.5	**4539.9**	2851.5	3277.9	4229.6	4018.1	3123.1	3688.9
Kung Fu Master	258.5	22736.3	**17257.2**	14346.1	5722.2	14307.8	9111.0	15469.7	13192.7
Ms Pacman	307.3	6951.6	**1480.0**	1204.1	941.9	1465.5	960.5	1247.7	1313.2
Pong	-20.7	14.6	**12.8**	-19.3	1.3	-16.5	-8.5	-16.0	-5.9
Private Eye	24.9	69571.3	58.3	97.8	100.0	**218.4**	-13.6	52.6	124.0
Qbert	163.9	13455.0	**1288.8**	1152.9	509.3	1042.4	854.4	606.6	669.1
Road Runner	11.5	7845.0	5640.6	9600.0	2696.7	5661.0	8895.1	10511.0	**14220.5**
Seaquest	68.4	42054.7	**683.3**	354.1	286.9	384.5	301.2	580.8	583.1
Up N Down	533.4	11693.2	3350.3	2877.4	2847.6	2955.2	3180.8	6604.6	**28138.5**
Mean Human-Norm'd	0.000	1.000	0.443	0.285	0.264	0.381	0.357	0.463	**0.704**
Median Human-Norm'd	0.000	1.000	0.144	0.161	0.204	0.175	0.268	0.307	**0.415**
Mean DQN@50M-Norm'd	0.000	23.382	0.232	0.239	0.197	0.325	0.171	0.336	**0.510**
Median DQN@50M-Norm'd	0.000	0.994	0.118	0.142	0.103	0.142	0.131	0.225	**0.361**
# Superhuman	0	N/A	2	2	1	2	2	5	**7**

Deep Reinforcement Learning Research Trends

- Representation Learning for Reinforcement Learning
  3. Pretraining Reward-Free Representations for Data-Efficient Reinforcement Learning (2021, ICLR)

Self-Supervision for Reinforcement Learning Workshop - ICLR 2021

PRETRAINING REWARD-FREE REPRESENTATIONS FOR DATA-EFFICIENT REINFORCEMENT LEARNING

Max Schwarzer[*1,2], **Nitarshan Rajkumar**[*1,2], **Michael Noukhovitch**[1,2], **Ankesh Anand**[1,2]
Laurent Charlin[1,3], **Devon Hjelm**[1,4], **Philip Bachman**[1,4], **Aaron Courville**[1,2]
[1]Mila, [2]Université de Montréal, [3]HEC Montréal, [4]Microsoft Research

ABSTRACT

Data efficiency poses a major challenge for deep reinforcement learning. We approach this issue from the perspective of self-supervised representation learning, leveraging reward-free exploratory data to pretrain encoder networks. We employ a novel combination of latent dynamics modelling and goal-reaching objectives, which exploit the inherent structure of data in reinforcement learning. We demonstrate that our method scales well with network capacity and pretraining data. When evaluated on the Atari 100k data-efficiency benchmark, our approach significantly outperforms previous methods combining unsupervised pretraining with task-specific finetuning, and approaches human-level performance.

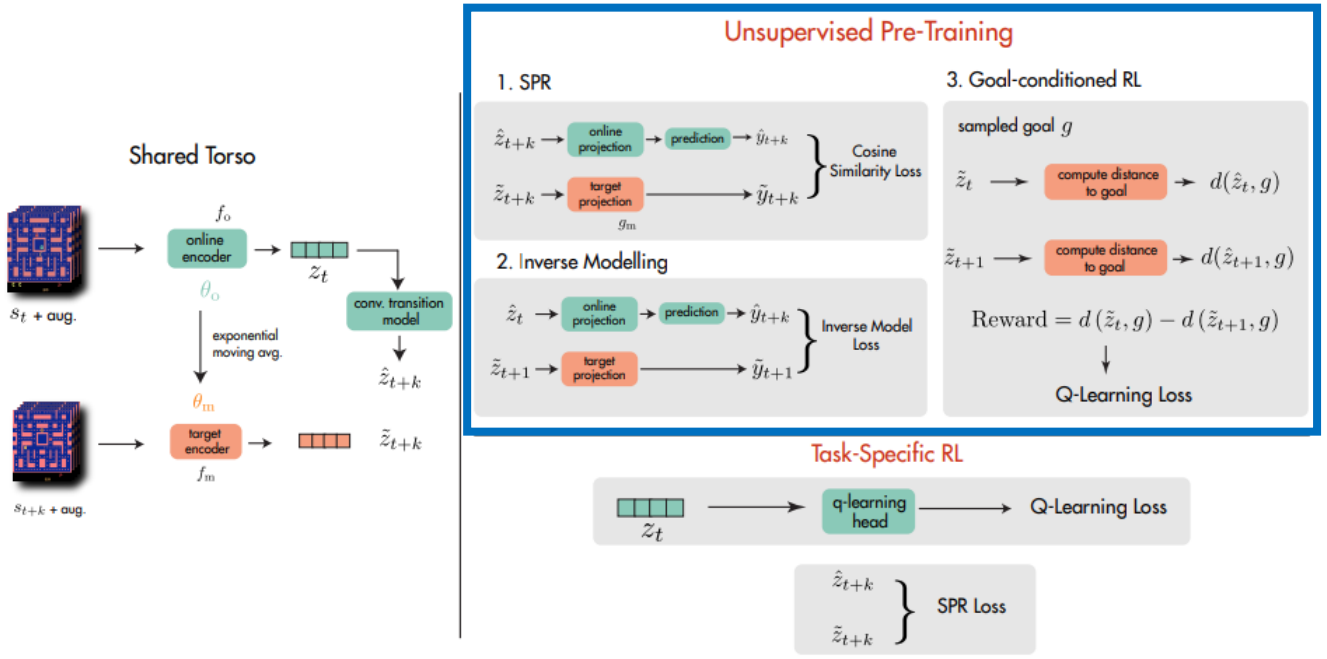
Deep Reinforcement Learning Research Trends

❖ Representation Learning for Reinforcement Learning

3. Pretraining Reward-Free Representations for Data-Efficient Reinforcement Learning (2021, ICLR)

✓ CURL, SPR과 달리 비지도 사전 학습 단계를 수행하는 **Two-Stage 방식**의 방법론

✓ 3개의 구성요소(SPR, Inverse Modeling, Goal-Conditioned RL)를 사용하여 사전 학습 수행

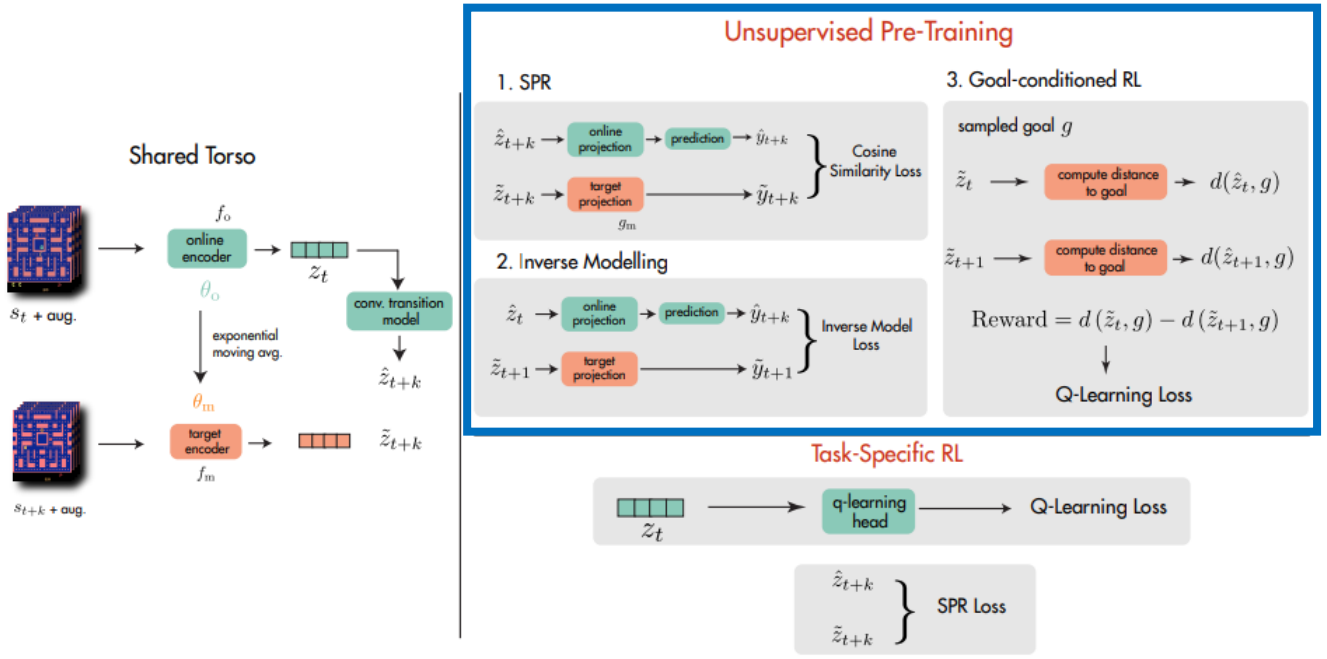


Deep Reinforcement Learning Research Trends

❖ Representation Learning for Reinforcement Learning

3. Pretraining Reward-Free Representations for Data-Efficient Reinforcement Learning (2021, ICLR)

- ✓ SPR: BYOL 구조를 사용 / 미래의 요약된 State ($\hat{z}_{t+k}$) 예측 및 k 반복하며 SSL Loss 계산
- ✓ Inverse Modeling: 요약된 현재 State ($\hat{z}_t$)와 Next State ($\hat{z}_{t+1}$)로 현재 Action ($\hat{a}_t$) 예측
- ✓ Goal-Conditioned RL: Goal (g)과 Goal-Conditioned Reward를 정의 및 학습



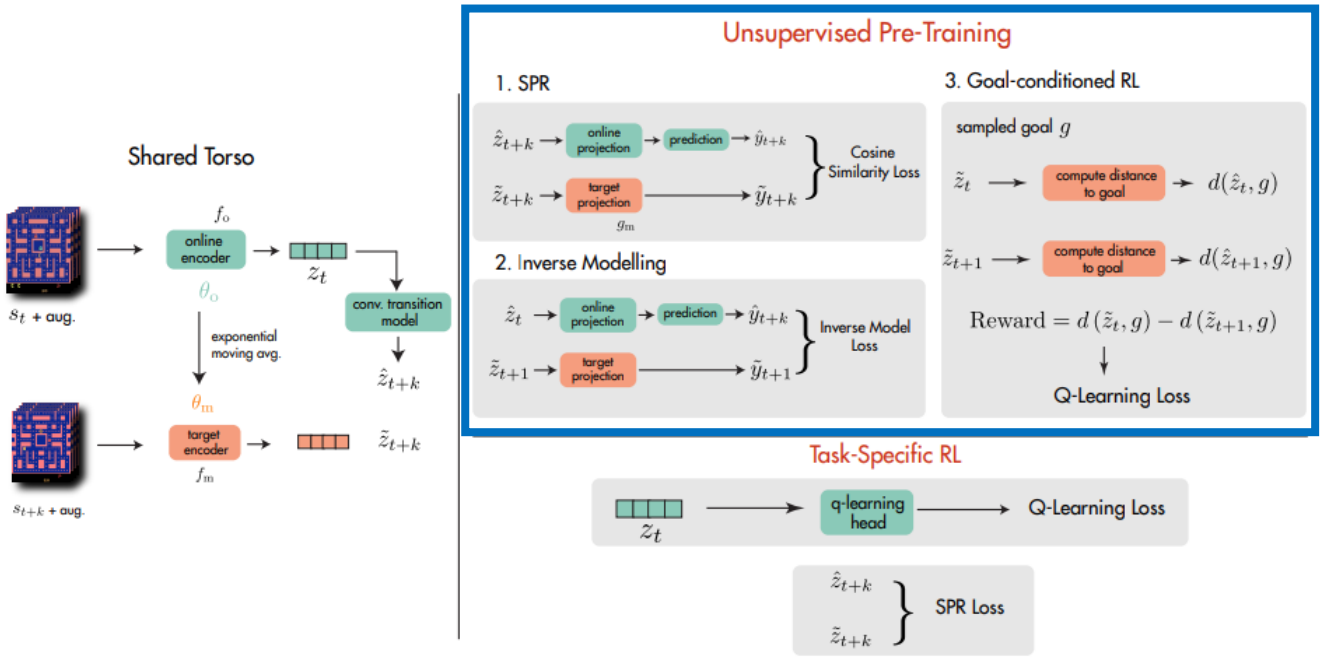
Deep Reinforcement Learning Research Trends

❖ Representation Learning for Reinforcement Learning

3. Pretraining Reward-Free Representations for Data-Efficient Reinforcement Learning (2021, ICLR)

✓ State (s_t)에 대한 Goal (g)는 가까운 미래 시점으로부터 Uniform하게 샘플링한 State의 타겟 표현

✓ Goal-Conditioned Reward는 요약된 State ($\tilde{z}_t, \tilde{z}_{t+1}$)와 Goal ($g$) 간의 거리로 정의



$$R(\tilde{z}_t, \tilde{z}_{t+1}, g) = d(\tilde{z}_t, g) - d(\tilde{z}_{t+1}, g)$$

$$d(\tilde{z}_t, g) = \exp\left(2\frac{\tilde{z}_t \cdot g}{||\tilde{z}_t||_2 \cdot ||g||_2} - 2\right)$$

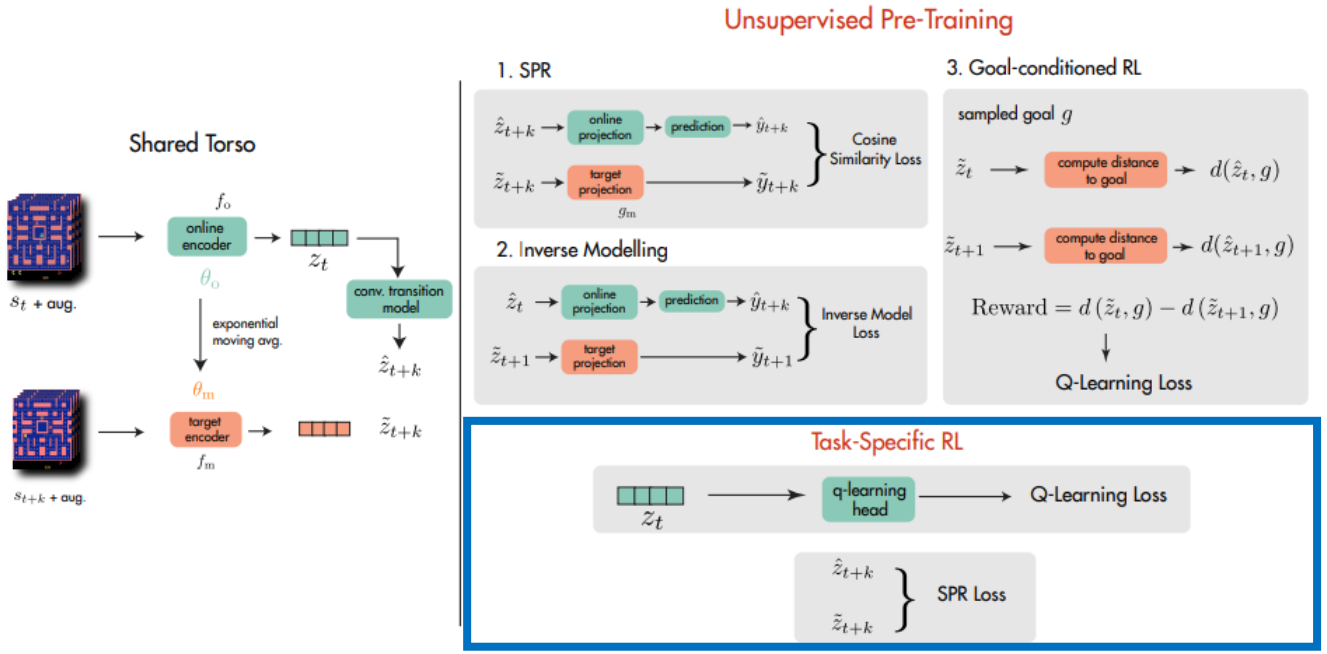
Deep Reinforcement Learning Research Trends

❖ Representation Learning for Reinforcement Learning

3. Pretraining Reward-Free Representations for Data-Efficient Reinforcement Learning (2021, ICLR)

✓ 사전 학습 후 Online Encoder와 Target Encoder를 사용하여 Fine-Tuning을 진행

✓ Fine-Tuning은 SPR과 강화학습 Loss를 가중합하여 학습 진행



Deep Reinforcement Learning Research Trends

❖ Representation Learning for Reinforcement Learning

3. Pretraining Reward-Free Representations for Data-Efficient Reinforcement Learning (2021, ICLR)

✓ 10만 Timestep으로 제한된 Atari 2600 26개 게임을 사용하여 데이터 효율성 향상 검증

✓ 제안하는 SGI 사전 학습에 Large Size ResNet Convolution을 사용한 SGI-C/L이 가장 우수

	Random	Human	SPR	ATC	SGI-R	SGI-E	SGI-W	SGI-C/S	SGI-C	SGI-C/L
Alien	227.8	7127.7	801.5	699.0	1034.5	857.6	1043.8	1070.5	1101.7	**1184.0**
Amidar	5.8	1719.5	176.3	95.4	154.8	166.8	**206.7**	185.9	168.2	171.2
Assault	222.4	742.0	571.0	509.8	446.6	583.1	759.5	632.4	905.1	**1326.5**
Asterix	210.0	8503.3	977.8	454.1	754.6	953.6	**1539.1**	651.8	835.6	567.2
Bank Heist	14.2	753.1	380.9	534.9	397.4	514.8	426.3	547.4	**608.4**	567.8
Battle Zone	2360.0	37187.5	**16651.0**	13683.8	4439.0	16417.0	7103.0	12107.0	13170.0	14462.0
Boxing	0.1	12.1	35.8	16.8	57.7	33.6	50.2	40.0	36.9	**73.9**
Breakout	1.7	30.5	17.1	16.9	23.4	17.8	35.4	23.8	42.8	**251.9**
Chopper Command	811.0	7387.8	974.8	870.8	784.7	1136.2	1040.1	1042.7	**1404.0**	1037.9
Crazy Climber	10780.5	35829.4	42923.6	74215.5	50561.2	76356.3	81057.4	75542.1	88561.2	**94602.2**
Demon Attack	152.1	1971.0	545.2	524.6	2198.7	357.5	1408.5	1135.5	968.1	**5634.8**
Freeway	0.0	29.6	24.4	5.7	2.1	15.1	26.5	12.5	**30.0**	28.6
Frostbite	65.2	4334.7	**1821.5**	222.6	349.3	981.4	247.7	861.1	741.3	927.8
Gopher	257.6	2412.5	715.2	946.2	1033.9	964.9	1846.0	1172.4	1660.4	**2035.8**
Hero	1027.0	30826.4	7019.2	6119.4	7875.2	6863.7	7503.9	7090.4	7474.0	**9975.9**
Jamesbond	29.0	302.8	365.4	272.6	263.9	383.8	**425.1**	413.2	366.4	394.8
Kangaroo	52.0	3035.0	**3276.4**	603.1	923.8	1588.9	598.6	1236.8	2172.8	1887.5
Krull	1598.0	2665.5	3688.9	4494.7	5672.6	4070.7	5583.2	**6161.3**	5734.0	5862.6
Kung Fu Master	258.5	22736.3	13192.7	11648.2	13349.2	11802.1	14199.7	16781.8	16137.8	**17340.7**
Ms Pacman	307.3	6951.6	1313.2	848.9	411.0	1278.3	1970.8	1519.5	1520.0	**2218.0**
Pong	-20.7	14.6	-5.9	-13.5	-3.9	4.2	4.7	**9.7**	7.6	7.7
Private Eye	24.9	69571.3	**124.0**	95.0	95.3	100.0	100.0	84.7	90.0	83.8
Qbert	163.9	13455.0	669.1	572.2	595.0	717.6	**855.6**	804.7	709.8	702.6
Road Runner	11.5	7845.0	14220.5	7989.3	5476.0	9195.2	18011.9	12083.5	**18370.2**	18306.8
Seaquest	68.4	42054.7	583.1	415.7	735.3	615.2	656.1	728.2	728.4	**1979.3**
Up N Down	533.4	11693.2	28138.5	84361.2	67968.1	63612.9	**84551.4**	42165.6	79228.8	46083.3
Median HNS	0.000	1.000	0.415	0.204	0.326	0.456	0.589	0.423	0.679	**0.755**
Mean HNS	0.000	1.000	0.704	0.780	0.888	0.838	1.144	0.914	1.149	**1.590**
#Games > Human	0	0	7	5	5	6	8	6	**9**	**9**
#Games > 0	0	26	**26**	**26**	25	**26**	**26**	**26**	**26**	**26**

Deep Reinforcement Learning Research Trends

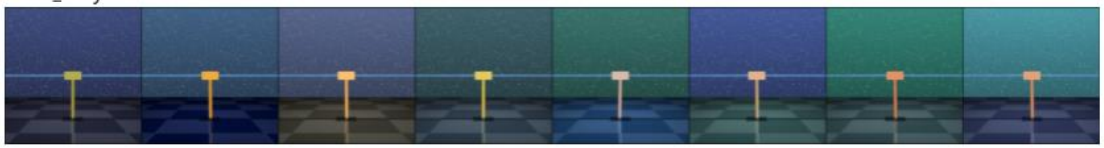
❖ Generalization in Reinforcement Learning

- **목표:** 데이터 증강 기법, 표현 학습 등을 통해 데이터 효율성 및 일반화 성능 향상
- 일반화 성능은 학습 환경과 다른 테스트 환경을 통해 검증

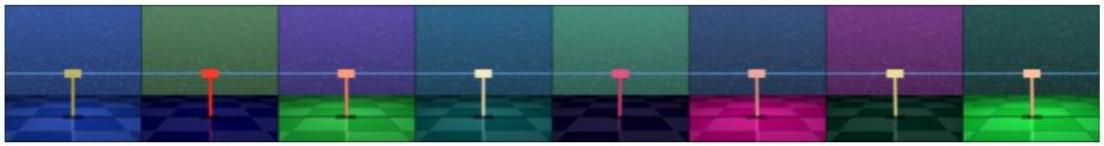
Training Environment

Test Environment

color_easy



color_hard



video_easy

video_hard

Deep Reinforcement Learning Research Trends

- Generalization in Reinforcement Learning
  1. Reinforcement Learning with Augmented Data (2020, NeurIPS)

Reinforcement Learning with Augmented Data

Michael Laskin* UC Berkeley

Kimin Lee* UC Berkeley

Adam Stooke UC Berkeley

Lerrel Pinto New York University

Pieter Abbeel UC Berkeley

Aravind Srinivas UC Berkeley

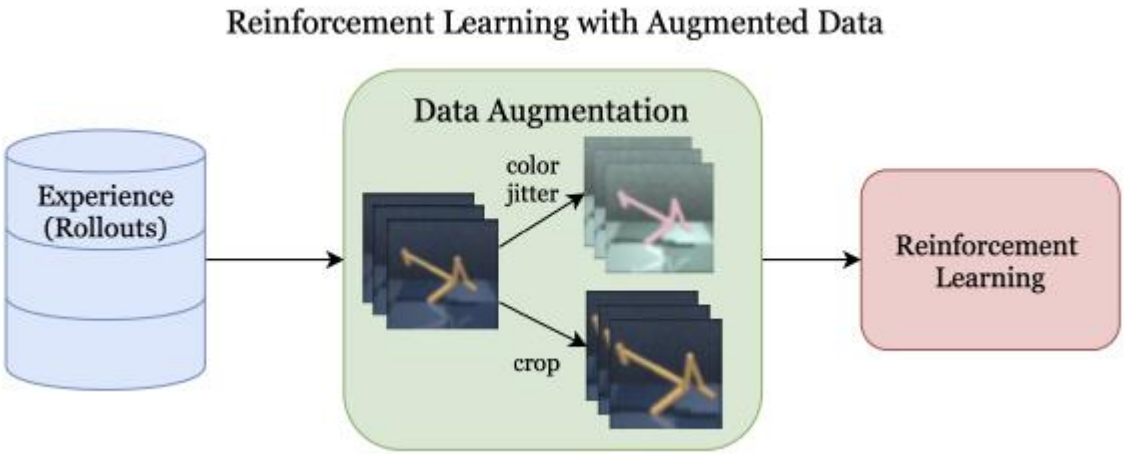
Abstract

Learning from visual observations is a fundamental yet challenging problem in Reinforcement Learning (RL). Although algorithmic advances combined with convolutional neural networks have proved to be a recipe for success, current methods are still lacking on two fronts: (a) data-efficiency of learning and (b) generalization to new environments. To this end, we present Reinforcement Learning with Augmented Data (RAD), a simple plug-and-play module that can enhance most RL algorithms. We perform the first extensive study of general data augmentations for RL on both pixel-based and state-based inputs, and introduce two new data augmentations - *random translate* and *random amplitude scale*. We show that augmentations such as random translate, crop, color jitter, patch cutout, random convolutions, and amplitude scale can enable simple RL algorithms to outperform complex state-of-the-art methods across common benchmarks. RAD sets a new state-of-the-art in terms of data-efficiency and final performance on the DeepMind Control Suite benchmark for pixel-based control as well as OpenAI Gym benchmark for state-based control. We further demonstrate that RAD significantly improves test-time generalization over existing methods on several OpenAI ProcGen benchmarks. Our RAD module and training code are available at https://www.github.com/MishaLaskin/rad.

Deep Reinforcement Learning Research Trends

❖ Generalization in Reinforcement Learning

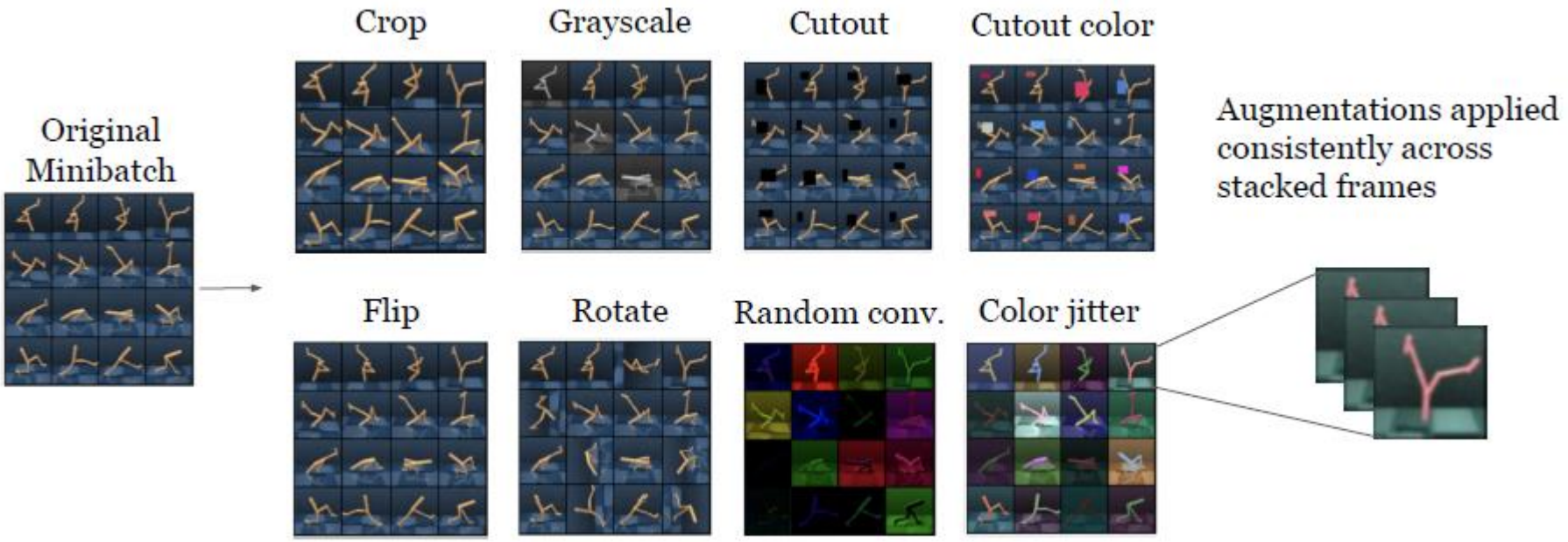
1. Reinforcement Learning with Augmented Data (2020, NeurIPS)
   - ✓ 강화학습을 위해 사용되는 데이터(Experience)에 데이터 증강 기법을 적용
   - ✓ 오직 상태 데이터에 데이터 증강 기법을 적용한 것으로 어떠한 강화학습 알고리즘과 결합 가능



Deep Reinforcement Learning Research Trends

❖ Generalization in Reinforcement Learning

1. Reinforcement Learning with Augmented Data (2020, NeurIPS)
   - ✓ Crop, Grayscale, Cutout, Cutout Color, Flip, Rotate, Random Conv, Color Jitter를 사용
   - ✓ Random Conv: 임의의 파라미터로 구성된 합성곱 레이어를 통해 데이터를 변환



Deep Reinforcement Learning Research Trends

❖ Generalization in Reinforcement Learning

1. Reinforcement Learning with Augmented Data (2020, NeurIPS)

✓ 10만, 50만 Timestpe으로 제한된 DMControl 6개 환경을 사용하여 데이터 효율성 향상 검증

500K STEP SCORES	RAD	CURL	PLANET	DREAMER	SAC+AE	SLACV1	PIXEL SAC	STATE SAC
FINGER, SPIN	**962** ± 16	926 ± 45	561 ± 284	796 ± 183	884 ± 128	673 ± 92	192 ± 166	923 ± 211
CARTPOLE, SWING	**847** ± 26	845 ± 45	475 ± 71	762 ± 27	735 ± 63	-	419 ± 40	848 ± 15
REACHER, EASY	**952** ± 25	929 ± 44	210 ± 44	793 ± 164	627 ± 58	-	145 ± 30	923 ± 24
CHEETAH, RUN	611 ± 51	518 ± 28	305 ± 131	**732** ± 103	550 ± 34	640 ± 19	197 ± 15	795 ± 30
WALKER, WALK	**918** ± 16	902 ± 43	351 ± 58	897 ± 49	847 ± 48	842 ± 51	42 ± 12	948 ± 54
CUP, CATCH	**960** ± 12	959 ± 27	460 ± 380	879 ± 87	794 ± 58	852 ± 71	312 ± 63	974 ± 33
100K STEP SCORES								
FINGER, SPIN	**831** ± 30	767 ± 56	136 ± 216	341 ± 70	740 ± 64	693 ± 141	224 ± 101	811 ± 46
CARTPOLE, SWING	**813** ± 32	582 ± 146	297 ± 39	326 ± 27	311 ± 11	-	200 ± 72	835 ± 22
REACHER, EASY	510 ± 76	**538** ± 233	20 ± 50	314 ± 155	274 ± 14	-	136 ± 15	746 ± 25
CHEETAH, RUN	**387** ± 82	299 ± 48	138 ± 88	238 ± 76	267 ± 24	319 ± 56	130 ± 12	616 ± 18
WALKER, WALK	**429** ± 58	403 ± 24	224 ± 48	277 ± 12	394 ± 22	361 ± 73	127 ± 24	891 ± 82
CUP, CATCH	495 ± 168	**769** ± 43	0 ± 0	246 ± 174	391 ± 82	512 ± 110	97 ± 27	746 ± 91

Deep Reinforcement Learning Research Trends

❖ Generalization in Reinforcement Learning

1. Reinforcement Learning with Augmented Data (2020, NeurIPS)
   - ✓ Random Crop 증강 기법이 효과적임을 실험적으로 확인

Task: walker, walk

Scores at 500k environment steps
for permutations of 2 data augmentations

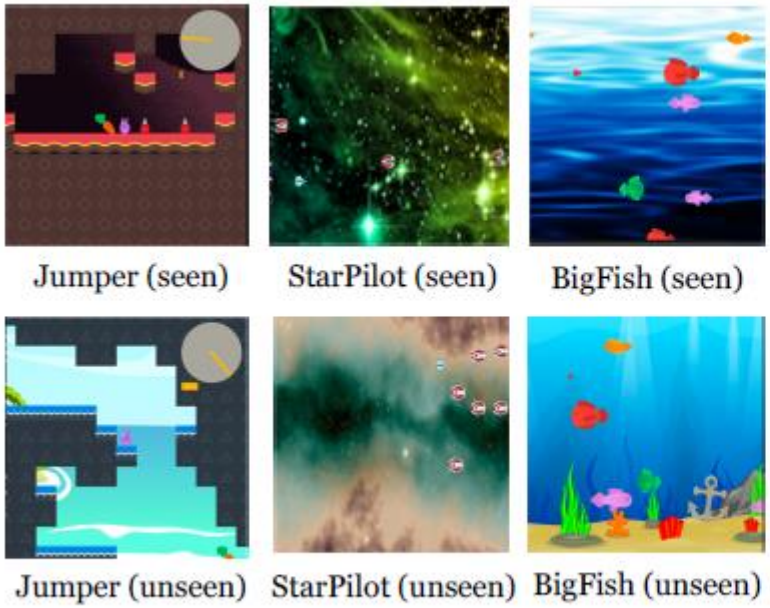
	crop	grayscale	rotate	cutout	color jitter	flip
crop	920	849	635	855	797	650
grayscale	856	175	349	187	214	231
rotate	604	403	268	293	392	394
cutout	722	206	376	215	31	284
color jitter	831	227	420	407	194	265
flip	828	210	391	244	264	223

Score: 900, 800, 700, 600, 500, 400, 300, 200, 100

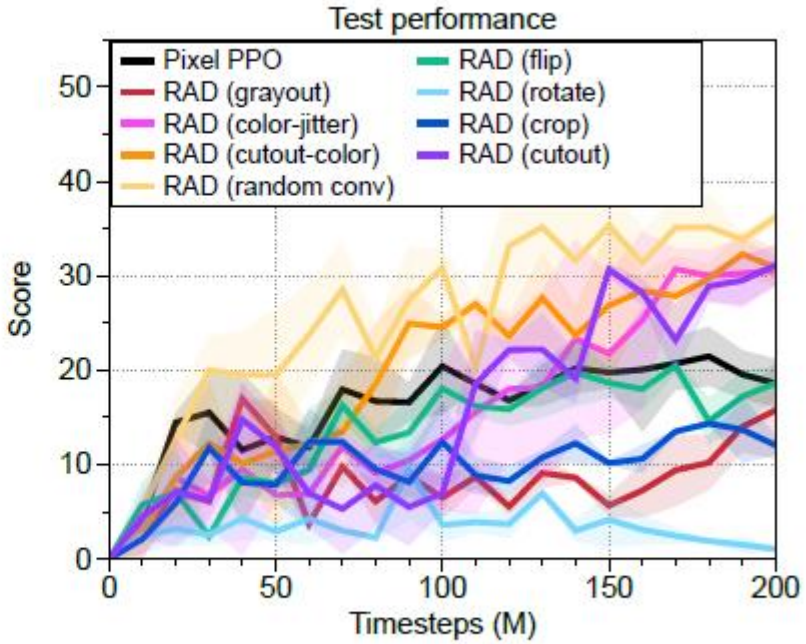
Deep Reinforcement Learning Research Trends

❖ Generalization in Reinforcement Learning

1. Reinforcement Learning with Augmented Data (2020, NeurIPS)
   - ✓ ProcGen 환경을 사용하여 학습 환경(Seen)과 다른 테스트 환경(Unseen)을 통해 일반화 성능 검증
   - ✓ 학습 환경 상태에 데이터 증강 기법만 적용하더라도 일반화 성능이 향상되는 것을 확인



(a) ProcGen



(b) Test performance on modified CoinRun

Deep Reinforcement Learning Research Trends

- Generalization in Reinforcement Learning

  2. Generalization in Reinforcement Learning by Soft Data Augmentation (2021, ICRA)

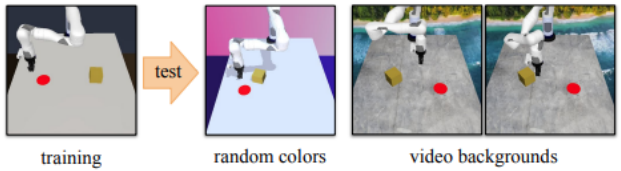
2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

Generalization in Reinforcement Learning by Soft Data Augmentation

Nicklas Hansen*†, Xiaolong Wang*

***Abstract*— Extensive efforts have been made to improve the generalization ability of Reinforcement Learning (RL) methods via domain randomization and data augmentation. However, as more factors of variation are introduced during training, optimization becomes increasingly challenging, and empirically may result in lower sample efficiency and unstable training. Instead of learning policies directly from augmented data, we propose SOft Data Augmentation (SODA), a method that decouples augmentation from policy learning. Specifically, SODA imposes a soft constraint on the encoder that aims to maximize the mutual information between latent representations of augmented and non-augmented data, while the RL optimization process uses strictly non-augmented data. Empirical evaluations are performed on diverse tasks from DeepMind Control suite as well as a robotic manipulation task, and we find SODA to significantly advance sample efficiency, generalization, and stability in training over state-of-the-art vision-based RL methods.[1]**

(a) Environments for DeepMind Control tasks. We consider 5 challenging continuous control tasks from this benchmark.

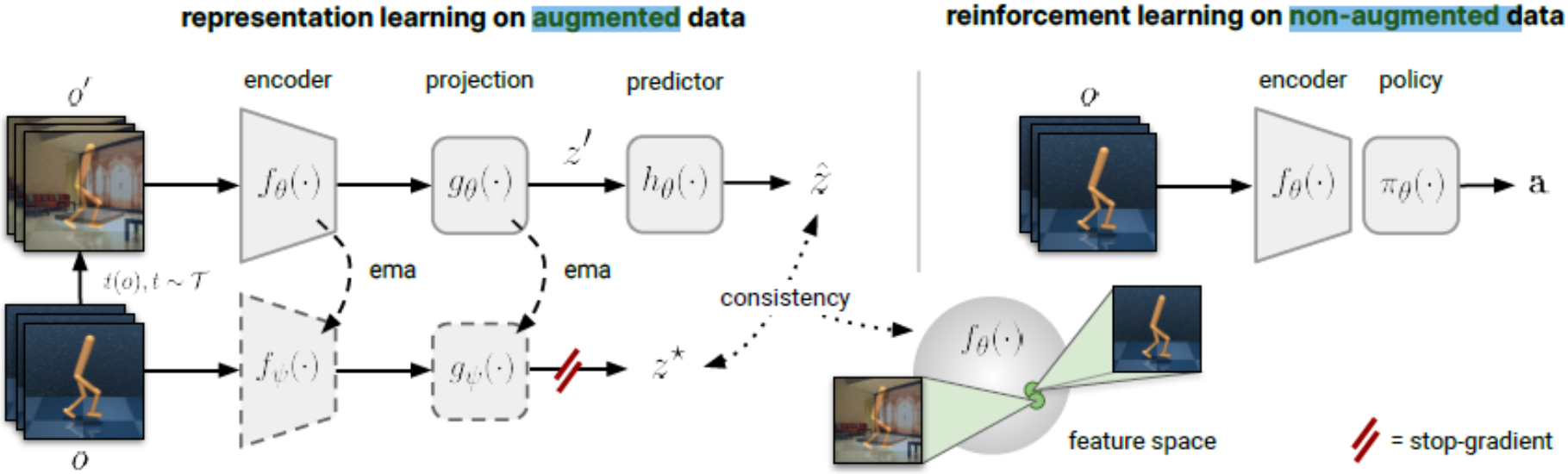


(b) Environments for robotic manipulation. The task is to push the yellow cube to the location of the red disc.

Deep Reinforcement Learning Research Trends

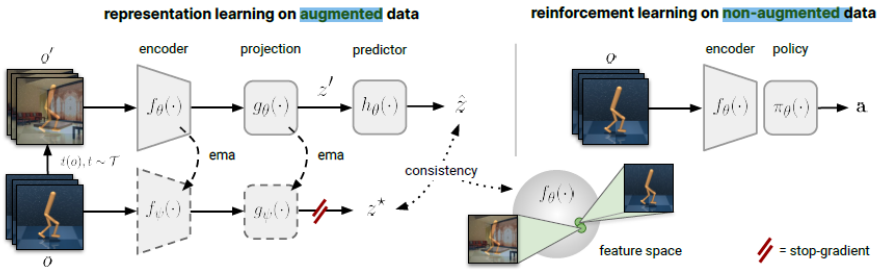
❖ Generalization in Reinforcement Learning

2. Generalization in Reinforcement Learning by Soft Data Augmentation (2021, ICRA)
   - ✓ 데이터 증강 기법을 적용한 상태로부터 정책(Policy)을 직접 학습하는 것이 아닌 표현 학습과 강화학습을 분리하여 학습하는 프로세스 제안
   - ✓ **이유:** 정책을 직접 학습할 때 심하게 변형된 상태가 오히려 악영향을 끼침
   - ✓ 증강 데이터는 BYOL을 적용하여 데이터에 대한 적절한 표현 학습할 때만 사용
   - ✓ 비증강 데이터는 강화학습 정책을 학습할 때만 사용



Deep Reinforcement Learning Research Trends

- ❖ Generalization in Reinforcement Learning
  - 2. Generalization in Reinforcement Learning by Soft Data Augmentation (2021, ICRA)
    - ✓ 비증강 데이터를 사용하여 ω만큼 강화학습 진행
    - ✓ ω이후 샘플된 데이터에 증강 기법을 적용하여 표현 학습 진행
    - ✓ 강화학습과 표현 학습에서 사용되는 Encoder는 Shared Network임



Algorithm 1 Soft Data Augmentation (SODA)

θ, ψ: randomly initialized network parameters
ω: RL updates per iteration
τ: momentum coefficient

for every iteration **do**
  for update $= 1, 2, ..., \omega$ **do**
    Sample batch of transitions $\nu \sim \mathcal{B}$
    Optimize $\mathcal{L}_{RL}(\nu)$ wrt θ
  Sample batch of observations $o \sim \mathcal{B}$
  Augment observations $o' = t(o),\ t \sim \mathcal{T}$
  Compute online predictions $\hat{z} = h_\theta(g_\theta(f_\theta(o')))$
  Compute target projections $z^\star = g_\psi(f_\psi(o))$
  Optimize $\mathcal{L}_{SODA}(\hat{z}, z^\star)$ wrt θ
  Update $\psi \leftarrow (1-\tau)\psi + \tau\theta$

Deep Reinforcement Learning Research Trends

(b) Random overlay data augmentation.

❖ Generalization in Reinforcement Learning

2. Generalization in Reinforcement Learning by Soft Data Augmentation (2021, ICRA)
   - ✓ DMControl 5개 환경에 일반화 성능 검증
   - ✓ 테스트 환경: 학습 환경의 배경을 변경한 환경과 색을 변경한 환경을 사용

Training Environment

video_hard

video backgrounds

color_hard

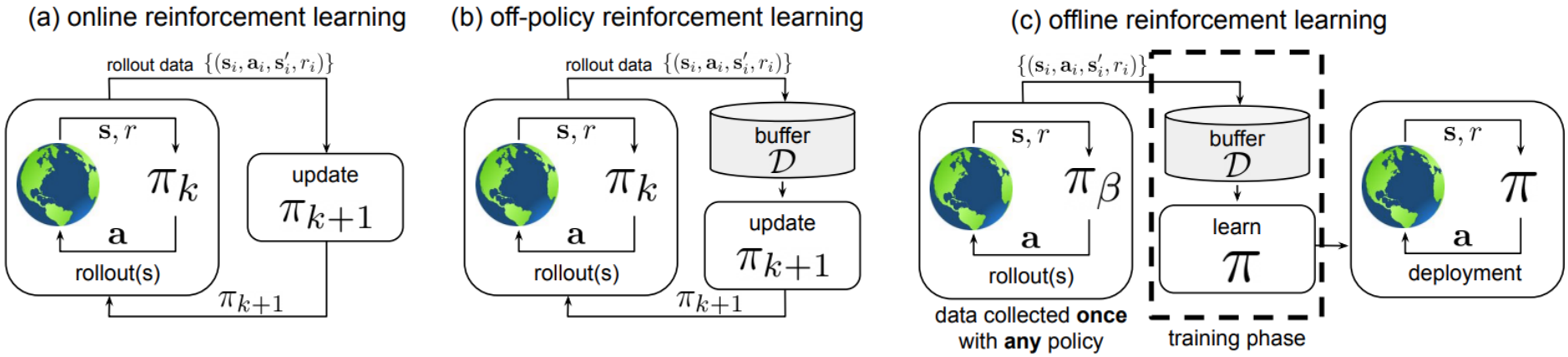
random colors

DMControl-GB (generalization)	CURL [19]	RAD [9]	PAD [42]	SAC (DR)	SAC (conv)	SODA (conv)	SAC (overlay)	SODA (overlay)	CURL [19]	RAD [9]	PAD [42]	SODA (overlay)
walker, walk	556 ±133	606 ±63	717 ±79	520 ±107	169 ±124	635 ±48	718 ±47	**768 ±38**	445 ±99	400 ±61	468 ±47	**692 ±68**
walker, stand	852 ±75	745 ±146	935 ±20	839 ±58	435 ±100	903 ±56	**960 ±2**	955 ±13	662 ±54	644 ±88	797 ±46	**893 ±12**
cartpole, swingup	404 ±67	373 ±72	521 ±76	524 ±184	176 ±62	474 ±143	718 ±30	**758 ±62**	454 ±110	590 ±53	630 ±63	**805 ±28**
ball_in_cup, catch	316 ±119	481 ±26	436 ±55	232 ±135	249 ±190	539 ±111	713 ±125	**875 ±56**	231 ±92	541 ±29	563 ±50	**949 ±19**
finger, spin	502 ±19	400 ±64	691 ±80	402 ±208	355 ±88	363 ±185	607 ±68	**695 ±97**	691 ±12	667 ±154	**803 ±72**	793 ±128

Deep Reinforcement Learning Research Trends

❖ Offline Reinforcement Learning

- **목표:** 추가적인 데이터 수집없이(상호작용 X) 기존에 수집된 데이터 내에서 정책(Policy)을 효율적으로 학습할 수 있도록 함
  - ✓ 강화학습을 실제 세계에 적용하기 위해서는 환경과 상호작용하여 얻는 Online 데이터가 필요
  - ✓ 상호작용할 수 있는 환경 부재, Online 데이터 수집 시 비용 및 시간 소비가 큼



Deep Reinforcement Learning Research Trends

- Offline Reinforcement Learning
  1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)

Conservative Q-Learning for Offline Reinforcement Learning

Aviral Kumar[1], **Aurick Zhou**[1], **George Tucker**[2], **Sergey Levine**[1,2]
[1]UC Berkeley, [2]Google Research, Brain Team
aviralk@berkeley.edu

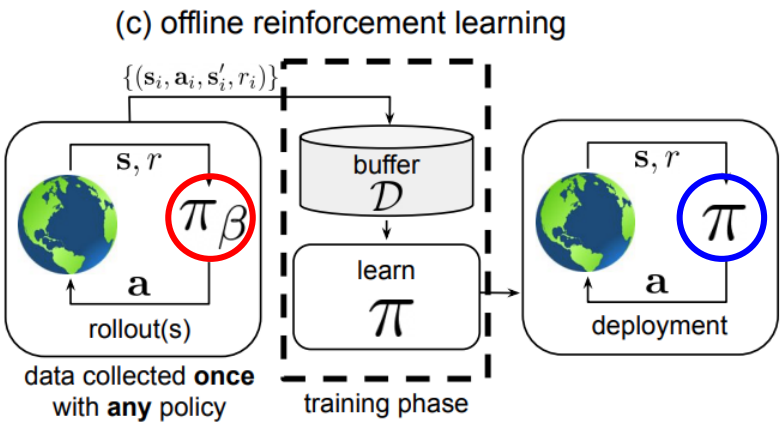
Abstract

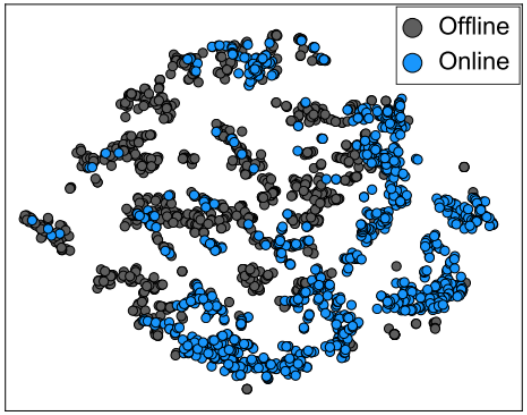
Effectively leveraging large, previously collected datasets in reinforcement learning (RL) is a key challenge for large-scale real-world applications. Offline RL algorithms promise to learn effective policies from previously-collected, static datasets without further interaction. However, in practice, offline RL presents a major challenge, and standard off-policy RL methods can fail due to overestimation of values induced by the distributional shift between the dataset and the learned policy, especially when training on complex and multi-modal data distributions. In this paper, we propose *conservative Q-learning (CQL)*, which aims to address these limitations by learning a conservative Q-function such that the expected value of a policy under this Q-function lower-bounds its true value. We theoretically show that CQL produces a lower bound on the value of the current policy and that it can be incorporated into a policy learning procedure with theoretical improvement guarantees. In practice, CQL augments the standard Bellman error objective with a simple Q-value regularizer which is straightforward to implement on top of existing deep Q-learning and actor-critic implementations. On both discrete and continuous control domains, we show that CQL substantially outperforms existing offline RL methods, often learning policies that attain 2-5 times higher final return, especially when learning from complex and multi-modal data distributions.

Deep Reinforcement Learning Research Trends

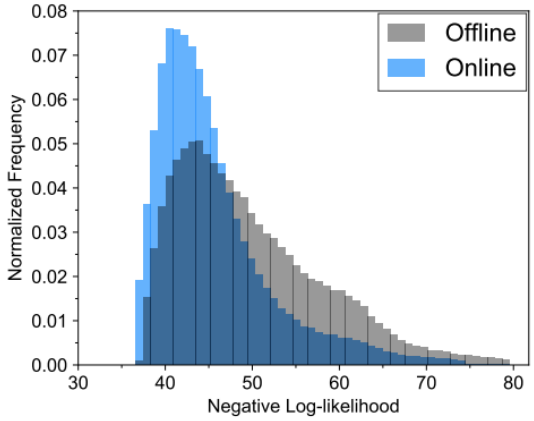
❖ Offline Reinforcement Learning

1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)
   - ✓ 환경과 상호작용 없이 기존에 수집된 데이터로 정책을 학습할 수 있는 Offline RL 등장
   - ✓ Q-Value의 과대 평가(Overestimation)가 발생하는 Offline RL 한계 존재
   - ✓ 과대 평가의 대표적인 원인은 **Distribution Shift (Out of Distribution)**
   - ✓ Offline 상에서 데이터를 수집하는 π_β와 Online 상에서 학습하는 π 간 분포 차이 존재





(a) t-SNE visualization

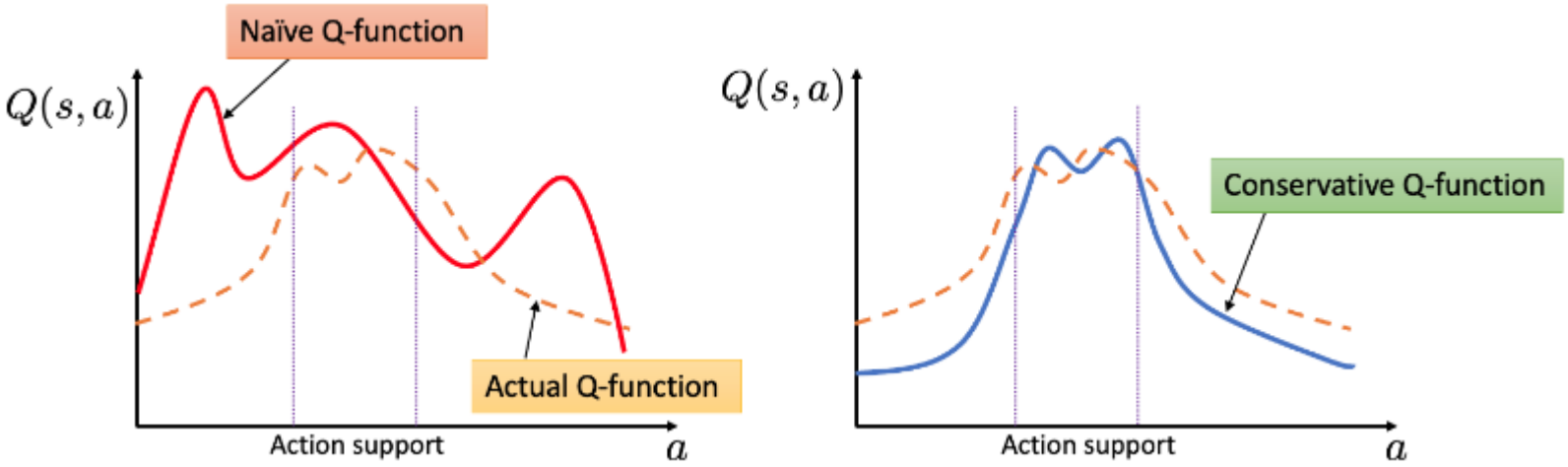


(b) Negative log-likelihood

Deep Reinforcement Learning Research Trends

❖ Offline Reinforcement Learning

1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)
   - ✓ 과대 평가 문제를 해결하기 위해 Conservative Q-Learning (CQL) 알고리즘 제안
   - ✓ CQL은 실제 Q-Value (Actual Q-Function)보다 정책의 Q-Value (Estimated Q-Function) 값이 낮아지도록 유도
   - ✓ 즉, CQL은 Q-Value의 Lower-Bound를 잘 찾는 것이 핵심



Deep Reinforcement Learning Research Trends

❖ Offline Reinforcement Learning

1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)

- ✓ $\pi_\beta(a|s)$에 따라 생성된 데이터셋 D에 대한 $V^\pi(s)$ (State s에 대한 기대값)를 추정하는 것이 목표
- ✓ **Solution:** 표준 벨만 목적식과 함께 Unseen Action들의 Q-Value를 최소화함으로써 Lower-Bound Q-Function 학습 → Unseen Action의 Q-Value에 대한 Regularization
- ✓ 1) Training the Q-Function for Lower-Bound Q-Value

$$\hat{Q}^{k+1} \leftarrow \arg\min_Q \alpha\, \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right] + \frac{1}{2}\,\mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}\left[\left(Q(\mathbf{s},\mathbf{a}) - \hat{\mathcal{B}}^\pi \hat{Q}^k(\mathbf{s},\mathbf{a})\right)^2\right], \quad (1)$$

Minimize Q-Value

Standard Bellman Error Objective

- 데이터셋 D에 없는 Action도 포함하는 Q-Function값을 최소화하는 방향으로 계산하는 것이 핵심
- α는 Tradeoff 계수로 값이 크면 State, Action에 대해 $\hat{Q}^\pi(s,a) \leq Q^\pi(s,a)$ 만족(Theorem 3.1 증명)

Theorem 3.1. *For any $\mu(\mathbf{a}|\mathbf{s})$ with $\operatorname{supp}\mu \subset \operatorname{supp}\hat{\pi}_\beta$, with probability $\geq 1-\delta$, $\hat{Q}^\pi$ (the Q-function obtained by iterating Equation (1)) satisifies:*

$$\forall \mathbf{s}\in\mathcal{D},\mathbf{a},\ \hat{Q}^\pi(s,a) \leq Q^\pi(\mathbf{s},\mathbf{a}) - \alpha\left[(I-\gamma P^\pi)^{-1}\frac{\mu}{\hat{\pi}_\beta}\right](\mathbf{s},\mathbf{a}) + \left[(I-\gamma P^\pi)^{-1}\frac{C_{r,T,\delta}R_{\max}}{(1-\gamma)\sqrt{|\mathcal{D}|}}\right](\mathbf{s},\mathbf{a}).$$

Thus, if α is sufficiently large, then $\hat{Q}^\pi(\mathbf{s},\mathbf{a}) \leq Q^\pi(\mathbf{s},\mathbf{a}), \forall \mathbf{s}\in\mathcal{D},\mathbf{a}$. When $\hat{\mathcal{B}}^\pi = \mathcal{B}^\pi$, any $\alpha > 0$ guarantees $\hat{Q}^\pi(\mathbf{s},\mathbf{a}) \leq Q^\pi(\mathbf{s},\mathbf{a}), \forall \mathbf{s}\in\mathcal{D},\mathbf{a}\in\mathcal{A}$.

Deep Reinforcement Learning Research Trends

❖ Offline Reinforcement Learning

1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)

✓ $\pi_\beta(a|s)$에 따라 생성된 데이터셋 D에 대한 $V^\pi(s)$ (State s에 대한 기대값)를 추정하는 것이 목표

✓ **Solution:** 표준 벨만 목적식과 함께 Unseen Action들의 Q-Value를 최소화함으로써 Lower-Bound Q-Function 학습 → Unseen Action의 Q-Value에 대한 Regularization

✓ 2) Focus on $V^\pi(s)$ Instead of $Q^\pi(s,a)$ → Lower-Bound $\hat{Q}^\pi(s,a) \leq Q^\pi(s,a)$ 개선

$$\hat{Q}^{k+1} \leftarrow \arg\min_Q \; \alpha \cdot \left(\mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right] - \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right]\right) + \frac{1}{2}\,\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[\left(Q(\mathbf{s},\mathbf{a}) - \hat{\mathcal{B}}^\pi\hat{Q}^k(\mathbf{s},\mathbf{a})\right)^2\right]. \quad (2)$$

- 파란색 박스를 최소화하는 Q-Value를 찾는 것이 핵심! But, 왼쪽 기대값은 Dataset D에 없는 Action에 대해서도 Q-Function을 계산하기 때문에 과대 평가 경향 존재 → 빨간 글씨 기대값이 커져야 전체를 최소화할 수 있으므로 Q-Value를 최대화하는 Term이 됨
- 따라서, $\mu(a|s) = \hat{\pi}(a|s)$일 때, $\hat{V}^\pi(s) \leq V^\pi(s)$ 보장할 수 있음(Theorem 3.2 증명)

Theorem 3.2 (Equation 2 results in a tighter lower bound). *The value of the policy under the Q-function from Equation 2, $\hat{V}^\pi(\mathbf{s}) = \mathbb{E}_{\pi(\mathbf{a}|\mathbf{s})}[\hat{Q}^\pi(\mathbf{s},\mathbf{a})]$, lower-bounds the true value of the policy obtained via exact policy evaluation, $V^\pi(\mathbf{s}) = \mathbb{E}_{\pi(\mathbf{a}|\mathbf{s})}[Q^\pi(\mathbf{s},\mathbf{a})]$, when $\mu = \pi$, according to:*

$$\forall \mathbf{s}\in\mathcal{D},\; \hat{V}^\pi(\mathbf{s}) \leq V^\pi(\mathbf{s}) - \alpha\left[(I-\gamma P^\pi)^{-1}\,\mathbb{E}_\pi\left[\frac{\pi}{\hat{\pi}_\beta} - 1\right]\right](\mathbf{s}) + \left[(I-\gamma P^\pi)^{-1}\frac{C_{r,T,\delta}R_{\max}}{(1-\gamma)\sqrt{|\mathcal{D}|}}\right](\mathbf{s}).$$

Thus, if $\alpha > \frac{C_{r,T}R_{\max}}{1-\gamma}\cdot\max_{\mathbf{s}\in\mathcal{D}}\frac{1}{|\mathcal{D}(\mathbf{s})|}\cdot\left[\sum_{\mathbf{a}}\pi(\mathbf{a}|\mathbf{s})\left(\frac{\pi(\mathbf{a}|\mathbf{s})}{\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})}-1\right)\right]^{-1}$, $\forall \mathbf{s}\in\mathcal{D}$, $\hat{V}^\pi(\mathbf{s}) \leq V^\pi(\mathbf{s})$, with probability $\geq 1-\delta$. When $\hat{\mathcal{B}}^\pi = \mathcal{B}^\pi$, then any $\alpha > 0$ guarantees $\hat{V}^\pi(\mathbf{s}) \leq V^\pi(\mathbf{s}), \forall \mathbf{s}\in\mathcal{D}$.

Deep Reinforcement Learning Research Trends

❖ Offline Reinforcement Learning

1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)

- ✓ $\pi_\beta(a|s)$에 따라 생성된 데이터셋 D에 대한 $V^\pi(s)$ (State s에 대한 기대값)를 추정하는 것이 목표
- ✓ **Solution:** 표준 벨만 목적식과 함께 Unseen Action들의 Q-Value를 최소화함으로써 Lower-Bound Q-Function 학습 → Unseen Action의 Q-Value에 대한 Regularization
- ✓ 3) Conservative Q-Function for Policy Improvement

$$\min_Q \max_\mu \alpha \left(\mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right] - \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right]\right)$$

Maximize Q-Value

$$+\frac{1}{2}\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[\left(Q(\mathbf{s},\mathbf{a}) - \hat{\mathcal{B}}^{\pi_k}\hat{Q}^k(\mathbf{s},\mathbf{a})\right)^2\right] + \mathcal{R}(\mu) \quad (\mathrm{CQL}(\mathcal{R})). \quad (3)$$

- Policy Evaluation과 Improvement를 한번에 진행(Policy Optimization 연산 효율성)
- Policy Improvement는 현재 Q-Function 값을 최대화하는 Policy를 근사화하는 $\mu(a|s)$를 선택(Max μ)
- $\mathcal{R}(\mu)$는 regularizer → 종류에 따라 CQL 특성이 변화(CQL Variants)
- CQL은 모든 State에 대해 $\hat{V}^\pi(s) \leq V^\pi(s)$임을 증명(Theorem 3.3)

Theorem 3.3 (CQL learns lower-bounded Q-values). *Let* $\pi_{\hat{Q}^k}(\mathbf{a}|\mathbf{s}) \propto \exp(\hat{Q}^k(\mathbf{s},\mathbf{a}))$ *and assume that* $D_{\mathrm{TV}}(\hat{\pi}^{k+1}, \pi_{\hat{Q}^k}) \leq \varepsilon$ *(i.e.,* $\hat{\pi}^{k+1}$ *changes slowly w.r.t to* $\hat{Q}^k$*). Then, the policy value under* $\hat{Q}^k$*, lower-bounds the actual policy value,* $\hat{V}^{k+1}(\mathbf{s}) \leq V^{k+1}(\mathbf{s})$ $\forall \mathbf{s}$ *if*

$$\mathbb{E}_{\pi_{\hat{Q}^k}(\mathbf{a}|\mathbf{s})}\left[\frac{\pi_{\hat{Q}^k}(\mathbf{a}|\mathbf{s})}{\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})} - 1\right] \geq \max_{\mathbf{a}\,s.t.\,\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})>0}\left(\frac{\pi_{\hat{Q}^k}(\mathbf{a}|\mathbf{s})}{\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})}\right)\cdot\varepsilon.$$

Deep Reinforcement Learning Research Trends

❖ Offline Reinforcement Learning

1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)

✓ $\pi_\beta(a|s)$에 따라 생성된 데이터셋 D에 대한 $V^\pi(s)$ (State s에 대한 기대값)를 추정하는 것이 목표

✓ **Solution:** 표준 벨만 목적식과 함께 Unseen Action들의 Q-Value를 최소화함으로써 Lower-Bound Q-Function 학습 → Unseen Action의 Q-Value에 대한 Regularization

✓ 3) Conservative Q-Function for Policy Improvement

Theorem 3.3 (CQL learns lower-bounded Q-values). *Let* $\pi_{\hat{Q}^k}(\mathbf{a}|\mathbf{s}) \propto \exp(\hat{Q}^k(\mathbf{s},\mathbf{a}))$ *and assume that* $D_{\mathrm{TV}}(\hat{\pi}^{k+1}, \pi_{\hat{Q}^k}) \leq \varepsilon$ *(i.e.,* $\hat{\pi}^{k+1}$ *changes slowly w.r.t to* $\hat{Q}^k$*). Then, the policy value under* $\hat{Q}^k$*, lower-bounds the actual policy value,* $\hat{V}^{k+1}(\mathbf{s}) \leq V^{k+1}(\mathbf{s}) \forall \mathbf{s}$ *if*

$$\mathbb{E}_{\pi_{\hat{Q}^k}(\mathbf{a}|\mathbf{s})}\left[\frac{\pi_{\hat{Q}^k}(\mathbf{a}|\mathbf{s})}{\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})} - 1\right] \geq \max_{\mathbf{a}\ s.t.\ \hat{\pi}_\beta(\mathbf{a}|\mathbf{s})>0}\left(\frac{\pi_{\hat{Q}^k}(\mathbf{a}|\mathbf{s})}{\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})}\right) \cdot \varepsilon.$$

- D_{TV}는 Total Variation Divergence를 의미($\delta(P,Q) = \sup_{A\in F}|P(A) - Q(A)|$)
- Learned Policy $(\hat{\pi}^{k+1})$와 Optimal Policy $(\pi_{\hat{Q}^k})$가 다를 경우 ϵ은 두 Policy 차이로 발생하는 과대 평가의 최대값이 됨
- Lower-Bound를 얻으려면 과소 평가 값이 커야 하며 이는 ϵ 작을 경우, 즉 Policy가 천천히 변경되어야 함

Deep Reinforcement Learning Research Trends

❖ Offline Reinforcement Learning

1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)
   - ✓ $\pi_\beta(a|s)$에 따라 생성된 데이터셋 D에 대한 $V^\pi(s)$ (State s에 대한 기대값)를 추정하는 것이 목표
   - ✓ **Solution:** 표준 벨만 목적식과 함께 Unseen Action들의 Q-Value를 최소화함으로써 Lower-Bound Q-Function 학습 → Unseen Action의 Q-Value에 대한 Regularization
   - ✓ 4) CQL Gap Expanding

Theorem 3.4 (CQL is gap-expanding). *At any iteration k, CQL expands the difference in expected Q-values under the behavior policy $\pi_\beta(\mathbf{a}|\mathbf{s})$ and μ_k, such that for large enough values of α_k, we have that* $\forall \mathbf{s}$, $\mathbb{E}_{\pi_\beta(\mathbf{a}|\mathbf{s})}[\hat{Q}^k(\mathbf{s},\mathbf{a})] - \mathbb{E}_{\mu_k(\mathbf{a}|\mathbf{s})}[\hat{Q}^k(\mathbf{s},\mathbf{a})] > \mathbb{E}_{\pi_\beta(\mathbf{a}|\mathbf{s})}[Q^k(\mathbf{s},\mathbf{a})] - \mathbb{E}_{\mu_k(\mathbf{a}|\mathbf{s})}[Q^k(\mathbf{s},\mathbf{a})]$.

- Dataset에 있는 Action (In Distribution)의 Q-Value와 Dataset에 없는 Action (Out of Distribution)의 Q-Value 차이 기준으로 예측한 Q-Function 간의 차이가 실제 Q-Function 간 차이보다 큰 경우를 의미
- Policy $\pi^k(a|s)$가 Data Distribution $\hat{\pi}_\beta(a|s)$에 더 가까워지도록 하는 것을 의미
- CQL Update는 Distribution Shift (Out of Distribution Action) 문제를 방지할 수 있게 됨

Deep Reinforcement Learning Research Trends

❖ Offline Reinforcement Learning

1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)

✓ $\pi_\beta(a|s)$에 따라 생성된 데이터셋 D에 대한 $V^\pi(s)$ (State s에 대한 기대값)를 추정하는 것이 목표

✓ **Solution:** 표준 벨만 목적식과 함께 Unseen Action들의 Q-Value를 최소화함으로써 Lower-Bound Q-Function 학습 → Unseen Action의 Q-Value에 대한 Regularization

✓ 5) CQL Variants $\mathcal{R}(\mu)$

$$\min_Q \max_\mu \alpha \left(\mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\mu(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right] - \mathbb{E}_{\mathbf{s}\sim\mathcal{D},\mathbf{a}\sim\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right]\right) + \frac{1}{2}\,\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[\left(Q(\mathbf{s},\mathbf{a}) - \hat{\mathcal{B}}^{\pi_k}\hat{Q}^k(\mathbf{s},\mathbf{a})\right)^2\right] + \mathcal{R}(\mu) \quad (\mathrm{CQL}(\mathcal{R})). \quad (3)$$

- **CQL(H):** $\mathcal{R}(\mu)$를 Fox H-Function으로 설정하고 풀이한 결과

$$\min_Q \alpha\mathbb{E}_{\mathbf{s}\sim\mathcal{D}}\left[\log\sum_{\mathbf{a}}\exp(Q(\mathbf{s},\mathbf{a})) - \mathbb{E}_{\mathbf{a}\sim\hat{\pi}_\beta(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right]\right] + \frac{1}{2}\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[\left(Q - \hat{\mathcal{B}}^{\pi_k}\hat{Q}^k\right)^2\right]. \quad (4)$$

- **CQL(ρ):** $\mathcal{R}(\mu)$를 KL-Divergence로 설정하고 풀이한 결과

$$\min_Q \alpha\mathbb{E}_{\mathbf{s}\sim d^{\pi_\beta}(\mathbf{s})}\left[\mathbb{E}_{\mathbf{a}\sim\rho(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\frac{\exp(Q(\mathbf{s},\mathbf{a}))}{Z}\right] - \mathbb{E}_{\mathbf{a}\sim\pi_\beta(\mathbf{a}|\mathbf{s})}\left[Q(\mathbf{s},\mathbf{a})\right]\right] + \frac{1}{2}\mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'\sim\mathcal{D}}\left[\left(Q - \mathcal{B}^{\pi_k}\hat{Q}^k\right)^2\right]. \quad (7)$$

https://en.wikipedia.org/wiki/Fox_H-function

Deep Reinforcement Learning Research Trends

❖ Offline Reinforcement Learning

1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)

✓ $\pi_\beta(a|s)$에 따라 생성된 데이터셋 D에 대한 $V^\pi(s)$ (State s에 대한 기대값)를 추정하는 것이 목표

✓ **Solution:** 표준 벨만 목적식과 함께 Unseen Action들의 Q-Value를 최소화함으로써 Lower-Bound Q-Function 학습 → Unseen Action의 Q-Value에 대한 Regularization

✓ 6) Safe Policy Improvement Guarantees

Theorem 3.5. *Let $\hat{Q}^\pi$ be the fixed point of Equation 2, then $\pi^*(\mathbf{a}|\mathbf{s}) := \arg\max_\pi \mathbb{E}_{\mathbf{s}\sim\rho(\mathbf{s})}[\hat{V}^\pi(\mathbf{s})]$ is equivalently obtained by solving:*

$$\pi^*(\mathbf{a}|\mathbf{s}) \leftarrow \arg\max_\pi \; J(\pi, \hat{M}) - \alpha \frac{1}{1-\gamma} \mathbb{E}_{\mathbf{s}\sim d^\pi_{\hat{M}}(\mathbf{s})} \left[D_{CQL}(\pi, \hat{\pi}_\beta)(\mathbf{s}) \right], \tag{5}$$

where $D_{CQL}(\pi, \pi_\beta)(\mathbf{s}) := \sum_{\mathbf{a}} \pi(\mathbf{a}|\mathbf{s}) \cdot \left(\frac{\pi(\mathbf{a}|\mathbf{s})}{\pi_\beta(\mathbf{a}|\mathbf{s})} - 1 \right)$.

- Theorem 3.5는 CQL의 최적화 과정을 보임

- $D_{CQL}(s)$에 의존하는 패널티를 통해 Learned Policy π와 Behavior Policy $\hat{\pi}_\beta$가 같아지도록 학습하게 됨 (차이가 없는 것을 보장)

$$\begin{aligned}
D_{\text{CQL}}(\mathbf{s}) &:= \sum_{\mathbf{a}} \pi(\mathbf{a}|\mathbf{s}) \left[\frac{\mu(\mathbf{a}|\mathbf{s})}{\pi_\beta(\mathbf{a}|\mathbf{s})} - 1 \right] \\
&= \sum_{\mathbf{a}} (\pi(\mathbf{a}|\mathbf{s}) - \pi_\beta(\mathbf{a}|\mathbf{s}) + \pi_\beta(\mathbf{a}|\mathbf{s})) \left[\frac{\mu(\mathbf{a}|\mathbf{s})}{\pi_\beta(\mathbf{a}|\mathbf{s})} - 1 \right] \\
&= \sum_{\mathbf{a}} (\pi(\mathbf{a}|\mathbf{s}) - \pi_\beta(\mathbf{a}|\mathbf{s})) \left[\frac{\pi(\mathbf{a}|\mathbf{s}) - \pi_\beta(\mathbf{a}|\mathbf{s})}{\pi_\beta(\mathbf{a}|\mathbf{s}} \right] + \sum_{\mathbf{a}} \pi_\beta(\mathbf{a}|\mathbf{s}) \left[\frac{\mu(\mathbf{a}|\mathbf{s})}{\pi_\beta(\mathbf{a}|\mathbf{s})} - 1 \right] \\
&= \sum_{\mathbf{a}} \underbrace{\left[\frac{(\pi(\mathbf{a}|\mathbf{s}) - \pi_\beta(\mathbf{a}|\mathbf{s}))^2}{\pi_\beta(\mathbf{a}|\mathbf{s})} \right]}_{\geq 0} + 0 \quad \text{since,} \sum_{\mathbf{a}} \pi(\mathbf{a}|\mathbf{s}) = \sum_{\mathbf{a}} \pi_\beta(\mathbf{a}|\mathbf{s}) = 1.
\end{aligned}$$

Note that the marked term, is positive since both the numerator and denominator are positive, and this implies that $D_{\text{CQL}}(\mathbf{s}) \geq 0$. Also, note that $D_{\text{CQL}}(\mathbf{s}) = 0$, iff $\pi(\mathbf{a}|\mathbf{s}) = \pi_\beta(\mathbf{a}|\mathbf{s})$. This implies that each value iterate incurs some underestimation, $\hat{V}^{k+1}(\mathbf{s}) \leq \mathcal{B}^\pi \hat{V}^k(\mathbf{s})$.

Deep Reinforcement Learning Research Trends

❖ Offline Reinforcement Learning

1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)

✓ $\pi_\beta(a|s)$에 따라 생성된 데이터셋 D에 대한 $V^\pi(s)$ (State s에 대한 기대값)를 추정하는 것이 목표

✓ **Solution:** 표준 벨만 목적식과 함께 Unseen Action들의 Q-Value를 최소화함으로써 Lower-Bound Q-Function 학습 → Unseen Action의 Q-Value에 대한 Regularization

✓ 7) CQL Pseudo Code

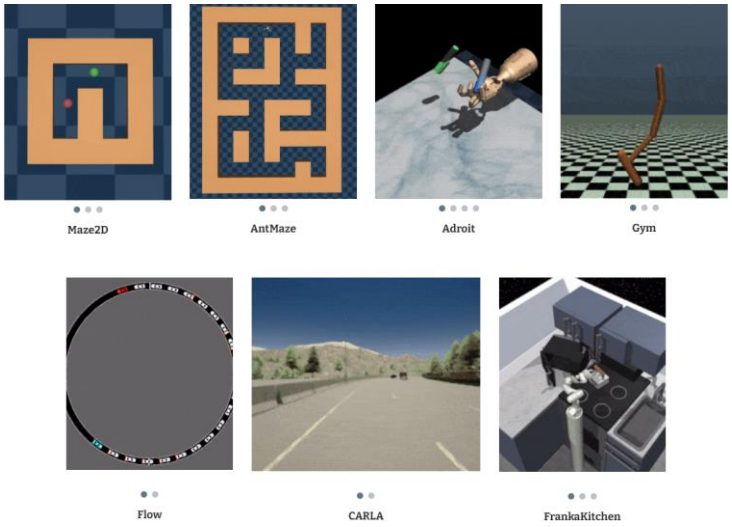
Algorithm 1 Conservative Q-Learning (both variants)

1: Initialize Q-function, Q_θ, and optionally a policy, π_ϕ.
2: **for** step t in $\{1, \ldots, \mathrm{N}\}$ **do**
3: Train the Q-function using G_Q gradient steps on objective from Equation 4
$\theta_t := \theta_{t-1} - \eta_Q \nabla_\theta \mathrm{CQL}(\mathcal{R})(\theta)$
(Use $\mathcal{B}^*$ for Q-learning, $\mathcal{B}^{\pi_{\phi_t}}$ for actor-critic)
4: (only with actor-critic) Improve policy π_ϕ via G_π gradient steps on ϕ with SAC-style entropy regularization:
$\phi_t := \phi_{t-1} + \eta_\pi \mathbb{E}_{\mathbf{s}\sim\mathcal{D}, \mathbf{a}\sim\pi_\phi(\cdot|\mathbf{s})}[Q_\theta(\mathbf{s}, \mathbf{a}) - \log \pi_\phi(\mathbf{a}|\mathbf{s})]$
5: **end for**

Deep Reinforcement Learning Research Trends

❖ Offline Reinforcement Learning

1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)
   - ✓ D4RL 벤치마크를 사용
   - ✓ 이전 Offline RL 방법론(BEAR, BRAC, SAC, BC)과 비교
   - ✓ 이전 방법론과 유사하거나 좋은 성능을 보였으며 난이도가 섞인 환경에서도 우수한 성능 증명



Task Name	SAC	BC	BEAR	BRAC-p	BRAC-v	CQL($\mathcal{H}$)
halfcheetah-random	30.5	2.1	25.5	23.5	28.1	**35.4**
hopper-random	**11.3**	9.8	9.5	**11.1**	**12.0**	**10.8**
walker2d-random	4.1	1.6	**6.7**	0.8	0.5	**7.0**
halfcheetah-medium	-4.3	36.1	38.6	**44.0**	**45.5**	**44.4**
walker2d-medium	0.9	6.6	33.2	72.7	**81.3**	79.2
hopper-medium	0.8	29.0	47.6	31.2	32.3	**58.0**
halfcheetah-expert	-1.9	**107.0**	**108.2**	3.8	-1.1	104.8
hopper-expert	0.7	**109.0**	**110.3**	6.6	3.7	**109.9**
walker2d-expert	-0.3	125.7	106.1	-0.2	-0.0	**153.9**
halfcheetah-medium-expert	1.8	35.8	51.7	43.8	45.3	**62.4**
walker2d-medium-expert	1.9	11.3	10.8	-0.3	0.9	**98.7**
hopper-medium-expert	1.6	**111.9**	4.0	1.1	0.8	**111.0**
halfcheetah-random-expert	53.0	1.3	24.6	30.2	2.2	**92.5**
walker2d-random-expert	0.8	0.7	1.9	0.2	2.7	**91.1**
hopper-random-expert	5.6	10.1	10.1	5.8	11.1	**110.5**
halfcheetah-mixed	-2.4	38.4	36.2	**45.6**	**45.9**	**46.2**
hopper-mixed	3.5	11.8	25.3	0.7	0.8	**48.6**
walker2d-mixed	1.9	11.3	10.8	-0.3	0.9	**26.7**

Deep Reinforcement Learning Research Trends

❖ Offline Reinforcement Learning

1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)
   - ✓ 특히, AntMaze, Adroit, Kitchen 환경에서 CQL만 유일하게 0이 아닌 점수를 얻음

Domain	Task Name	BC	SAC	BEAR	BRAC-p	BRAC-v	CQL($\mathcal{H}$)	CQL(ρ)
AntMaze	antmaze-umaze	65.0	0.0	**73.0**	50.0	70.0	**74.0**	**73.5**
	antmaze-umaze-diverse	55.0	0.0	61.0	40.0	70.0	**84.0**	61.0
	antmaze-medium-play	0.0	0.0	0.0	0.0	0.0	**61.2**	4.6
	antmaze-medium-diverse	0.0	0.0	8.0	0.0	0.0	**53.7**	5.1
	antmaze-large-play	0.0	0.0	0.0	0.0	0.0	**15.8**	3.2
	antmaze-large-diverse	0.0	0.0	0.0	0.0	0.0	**14.9**	2.3
Adroit	pen-human	34.4	6.3	-1.0	8.1	0.6	37.5	**55.8**
	hammer-human	1.5	0.5	0.3	0.3	0.2	**4.4**	2.1
	door-human	0.5	3.9	-0.3	-0.3	-0.3	**9.9**	**9.1**
	relocate-human	0.0	0.0	-0.3	-0.3	-0.3	0.20	**0.35**
	pen-cloned	**56.9**	23.5	26.5	1.6	-2.5	39.2	40.3
	hammer-cloned	0.8	0.2	0.3	0.3	0.3	2.1	**5.7**
	door-cloned	-0.1	0.0	-0.1	-0.1	-0.1	0.4	**3.5**
	relocate-cloned	**-0.1**	-0.2	-0.3	-0.3	-0.3	**-0.1**	**-0.1**
Kitchen	kitchen-complete	33.8	15.0	0.0	0.0	0.0	**43.8**	31.3
	kitchen-partial	33.8	0.0	13.1	0.0	0.0	**49.8**	**50.1**
	kitchen-undirected	47.5	2.5	47.2	0.0	0.0	**51.0**	**52.4**

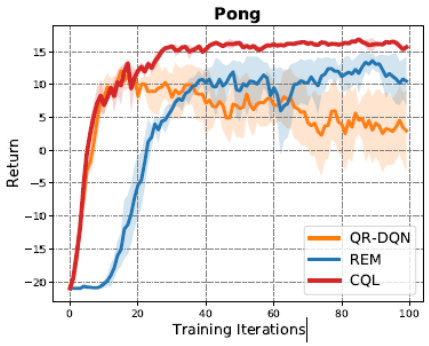
Deep Reinforcement Learning Research Trends

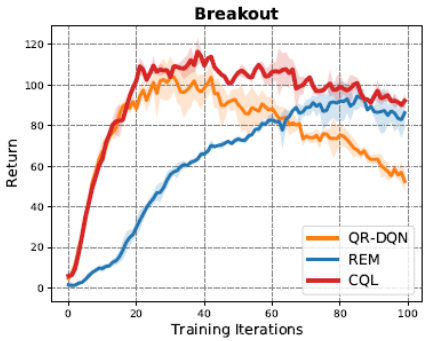
❖ Offline Reinforcement Learning

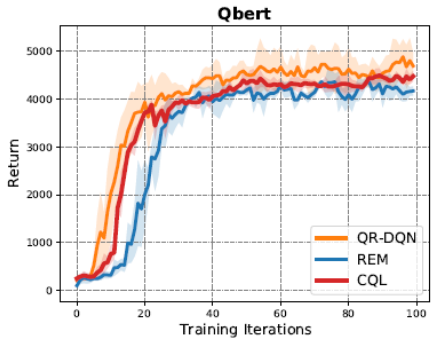
1. Conservative Q-Learning for Offline Reinforcement Learning (2020, NeurIPS)

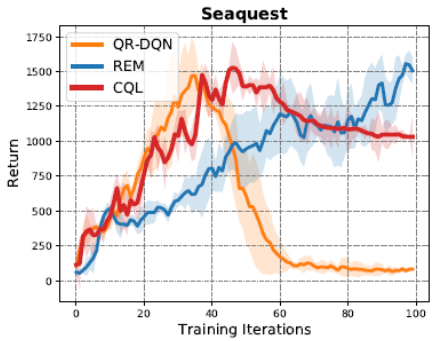
✓ Atari 2600 게임에서도 CQL의 우수함을 증명

Task Name	QR-DQN	REM	CQL($\mathcal{H}$)
Pong (1%)	-13.8	-6.9	**19.3**
Breakout	7.9	11.0	**61.1**
Q*bert	383.6	343.4	**14012.0**
Seaquest	672.9	499.8	**779.4**
Asterix	166.3	386.5	**592.4**
Pong (10%)	15.1	8.9	**18.5**
Breakout	151.2	86.7	**269.3**
Q*bert	7091.3	8624.3	**13855.6**
Seaquest	2984.8	**3936.6**	3674.1
Asterix	**189.2**	75.1	156.3









Deep Reinforcement Learning Research Trends

❖ Multi-Task Reinforcement Learning

- **목표:** 데이터 효율성, 학습 안정성 향상을 위해 Multi-Task 학습으로 서로 다른 환경에도 적용할 수 있는 글로벌 정책을 학습하도록 함
  - ✓ 단일 테스크는 정해진 환경에 알맞은 정책만을 학습하므로 데이터 비효율적임
  - ✓ 멀티 테스크 학습 시 다른 테스크의 Gradient가 부정적으로 간섭하여 불안정한 학습을 야기

Deep Reinforcement Learning Research Trends

- Multi-Task Reinforcement Learning
  1. Distral: Robust Multitask Reinforcement Learning (2017, NeurIPS)

Distral: Robust Multitask Reinforcement Learning

Yee Whye Teh, Victor Bapst, Wojciech Marian Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, Razvan Pascanu
DeepMind
London, UK

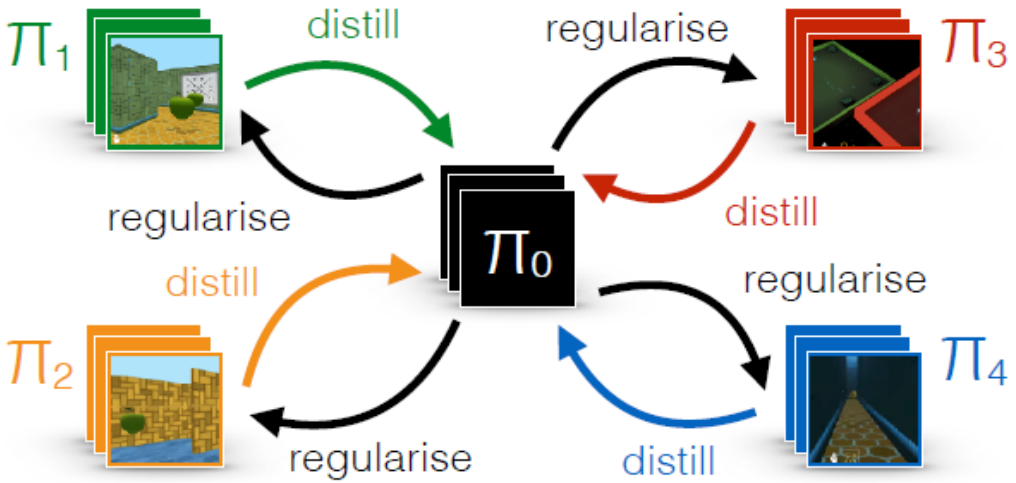
Abstract

Most deep reinforcement learning algorithms are data inefficient in complex and rich environments, limiting their applicability to many scenarios. One direction for improving data efficiency is multitask learning with shared neural network parameters, where efficiency may be improved through transfer across related tasks. In practice, however, this is not usually observed, because gradients from different tasks can interfere negatively, making learning unstable and sometimes even less data efficient. Another issue is the different reward schemes between tasks, which can easily lead to one task dominating the learning of a shared model. We propose a new approach for joint training of multiple tasks, which we refer to as Distral (distill & transfer learning). Instead of sharing parameters between the different workers, we propose to share a "distilled" policy that captures common behaviour across tasks. Each worker is trained to solve its own task while constrained to stay close to the shared policy, while the shared policy is trained by distillation to be the centroid of all task policies. Both aspects of the learning process are derived by optimizing a joint objective function. We show that our approach supports efficient transfer on complex 3D environments, outperforming several related methods. Moreover, the proposed learning process is more robust to hyperparameter settings and more stable—attributes that are critical in deep reinforcement learning.

Deep Reinforcement Learning Research Trends

❖ Multi-Task Reinforcement Learning

1. Distral: Robust Multitask Reinforcement Learning (2017, NeurIPS)
   - ✓ π_o: Distilled Policy (Shared Policy)
   - ✓ π_i: Task-Specific Policy
   - ✓ 한 시나리오의 π_1에서 얻은 Action 또는 Representation을 π_o로 Distill한 다음 다른 시나리오 π_i로 전송되는 방식



Deep Reinforcement Learning Research Trends

❖ Multi-Task Reinforcement Learning

1. Distral: Robust Multitask Reinforcement Learning (2017, NeurIPS)
   - ✓ π_o와 π_i는 테스크에 대한 Common Action을 포착하도록 함
   - ✓ KL Divergence로 각 Policy π_i를 Distilled Policy π_o로 정규화 함(파란색 박스)
   - ✓ 강화학습의 Exploration을 위해 Entropy Maximization Term 추가(빨간색 박스)
   - ✓ 수식 (1)을 최대화하는 방향으로 학습 진행

$$J(\pi_0, \{\pi_i\}_{i=1}^n) = \sum_i \mathbb{E}_{\pi_i}\left[\sum_{t\geq 0}\gamma^t R_i(a_t,s_t) - c_{\mathrm{KL}}\gamma^t \log\frac{\pi_i(a_t|s_t)}{\pi_0(a_t|s_t)} - c_{\mathrm{Ent}}\gamma^t \log \pi_i(a_t|s_t)\right]$$
$$= \sum_i \mathbb{E}_{\pi_i}\left[\sum_{t\geq 0}\gamma^t R_i(a_t,s_t) + \frac{\gamma^t\alpha}{\beta}\log\pi_0(a_t|s_t) - \frac{\gamma^t}{\beta}\log\pi_i(a_t|s_t)\right] \quad (1)$$

Deep Reinforcement Learning Research Trends

❖ Multi-Task Reinforcement Learning

1. Distral: Robust Multitask Reinforcement Learning (2017, NeurIPS)

✓ 제안하는 방법은 Q-Learning과 Policy Gradient method에 모두 사용 가능

Q-Learning

$$V_i(s_t) = \frac{1}{\beta} \log \sum_{a_t} \pi_0^{\alpha}(a_t|s_t) \exp\left[\beta Q_i(a_t, s_t)\right] \tag{2}$$

$$Q_i(a_t, s_t) = R_i(a_t, s_t) + \gamma \sum_{s_t} p_i(s_{t+1}|s_t, a_t) V_i(s_{t+1}) \tag{3}$$

Policy Gradient

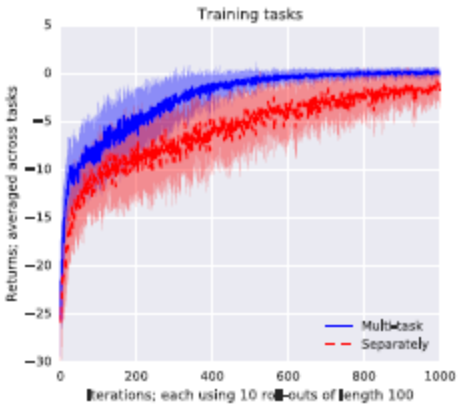
$$\nabla_{\theta_0} J = \sum_i \mathbb{E}_{\hat{\pi}_i} \left[\sum_{t \geq 1} \nabla_{\theta_0} \log \hat{\pi}_i(a_t|s_t) \left(\sum_{u \geq t} \gamma^u (R_i^{\text{reg}}(a_u, s_u)) \right) \right] \tag{10}$$

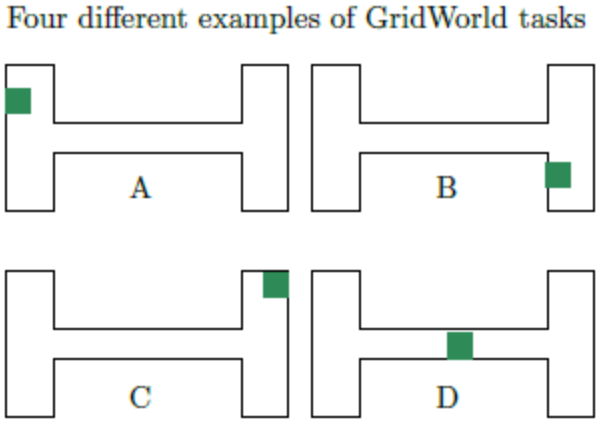
$$+ \frac{\alpha}{\beta} \sum_i \mathbb{E}_{\hat{\pi}_i} \left[\sum_{t \geq 1} \gamma^t \sum_{a'_t} (\hat{\pi}_i(a'_t|s_t) - \hat{\pi}_0(a'_t|s_t)) \nabla_{\theta_0} h_{\theta_0}(a'_t|s_t) \right]$$

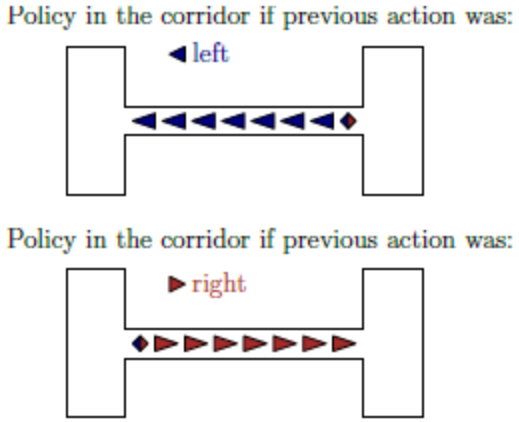
Deep Reinforcement Learning Research Trends

❖ Multi-Task Reinforcement Learning

1. Distral: Robust Multitask Reinforcement Learning (2017, NeurIPS)
   - ✓ Two Room Grid World에 Distral과 싱글 테스크 강화학습 적용하여 비교
   - ✓ 플랏 기준으로 Multi-Task를 수행하는 Distral이 빠르게 수렴하며 안정적인 학습을 보임





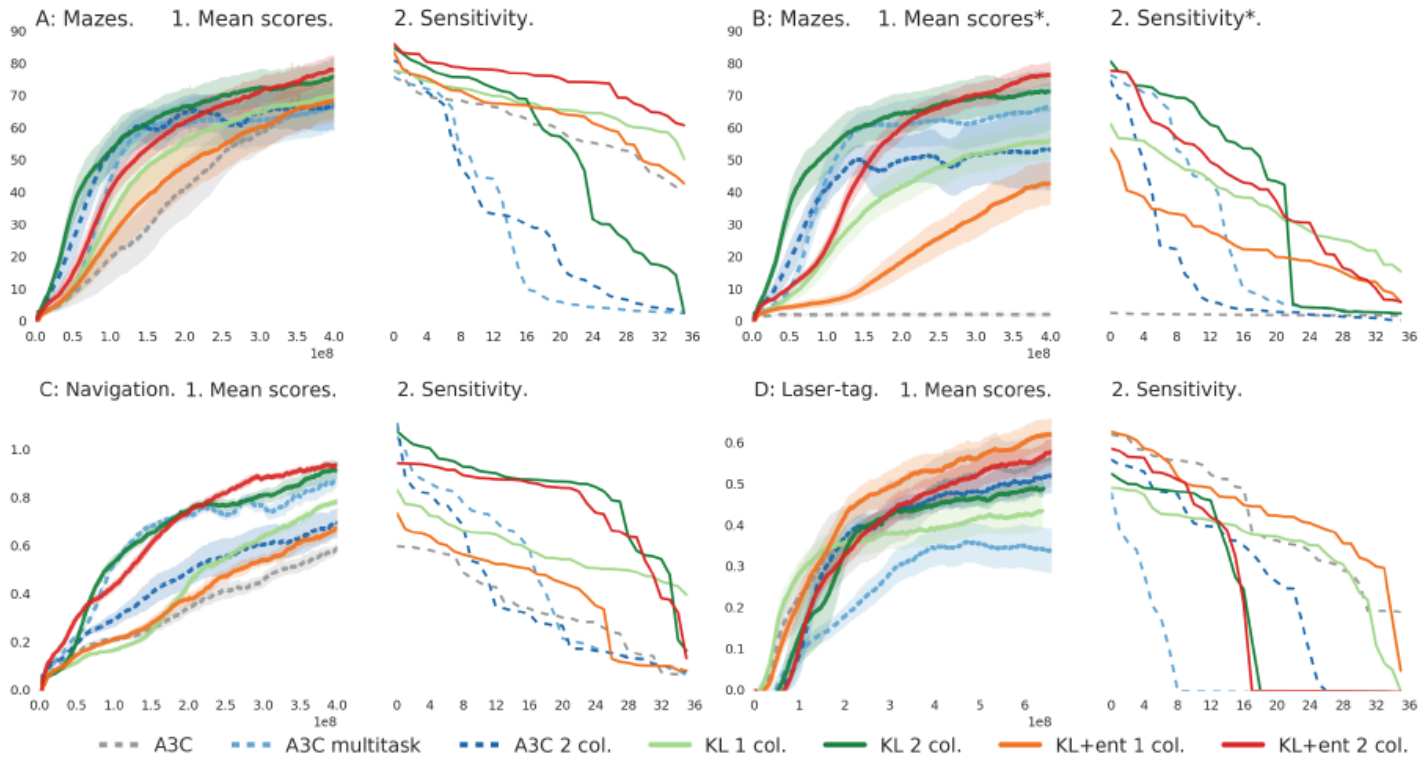


Deep Reinforcement Learning Research Trends

❖ Multi-Task Reinforcement Learning

1. Distral: Robust Multitask Reinforcement Learning (2017, NeurIPS)

   ✓ 3D 환경 Mazes, Navigation, Laser-tag에서도 싱글 테스크 학습보다 빠르게 수렴하며 안정적인 학습을 보여줌



Deep Reinforcement Learning Research Trends

❖ Multi-Agent Reinforcement Learning

- **목표:** 주어진 환경에서 두 개 이상의 에이전트가 협업 또는 경쟁을 통해 높은 보상을 얻을 수 있는 정책(Policy)을 학습하는 것
  - ✓ 협업 (Cooperation): 다수의 에이전트가 협력하여 공동의 목표를 달성할 수 있는 방법
  - ✓ 경쟁 (Competition): 다수의 에이전트가 경쟁하며 학습하는 방법

협업 (Cooperation)

경쟁 (Competition)

Deep Reinforcement Learning Research Trends

- Multi-Agent Reinforcement Learning
  1. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning (2018, ICML)

QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning

Tabish Rashid [*1] Mikayel Samvelyan [*2] Christian Schroeder de Witt [1]
Gregory Farquhar [1] Jakob Foerster [1] Shimon Whiteson [1]

Abstract

In many real-world settings, a team of agents must coordinate their behaviour while acting in a decentralised way. At the same time, it is often possible to train the agents in a centralised fashion in a simulated or laboratory setting, where global state information is available and communication constraints are lifted. Learning joint action-values conditioned on extra state information is an attractive way to exploit centralised learning, but the best strategy for then extracting decentralised policies is unclear. Our solution is QMIX, a novel value-based method that can train decentralised policies in a centralised end-to-end fashion. QMIX employs a network that estimates joint action-values as a complex non-linear combination of per-agent values that condition only on local observations. We structurally enforce that the joint-action value is monotonic in the per-agent values, which allows tractable maximisation of the joint action-value in off-policy learning, and guarantees consistency between the centralised and decentralised policies. We evaluate QMIX on a challenging set of StarCraft II micromanagement tasks, and show that QMIX significantly outperforms existing value-based multi-agent reinforcement learning methods.

(a) 5 Marines map (b) 2 Stalkers & 3 Zealots map

Figure 1. Decentralised unit micromanagement in StarCraft II, where each learning agent controls an individual unit. The goal is to coordinate behaviour across agents to defeat all enemy units.

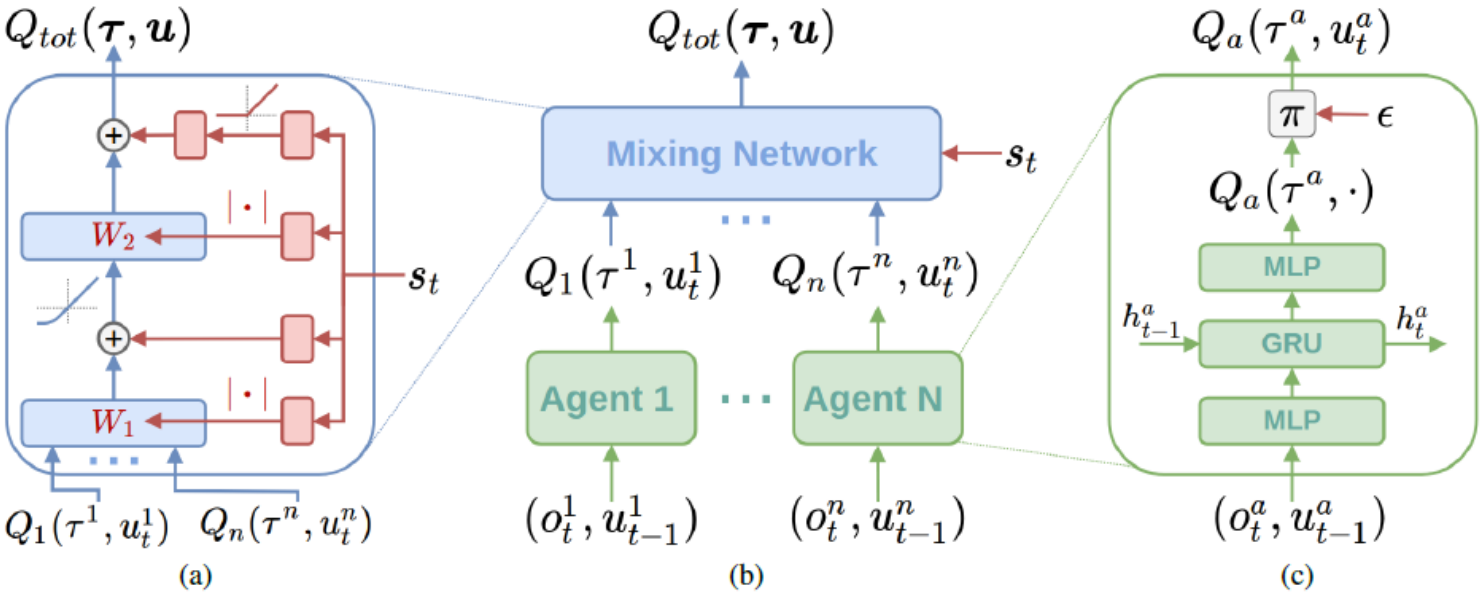
In many such settings, partial observability and/or communication constraints necessitate the learning of *decentralised policies*, which condition only on the local action-observation history of each agent. Decentralised policies also naturally attenuate the problem that joint action spaces grow exponentially with the number of agents, often rendering the application of traditional single-agent RL methods impractical.

Fortunately, decentralised policies can often be learned in a centralised fashion in a simulated or laboratory setting. This often grants access to additional state information, otherwise hidden from agents, and removes inter-agent communication constraints. The paradigm of *centralised training with decentralised execution* (Oliehoek et al., 2008; Kraemer

Deep Reinforcement Learning Research Trends

❖ Multi-Agent Reinforcement Learning

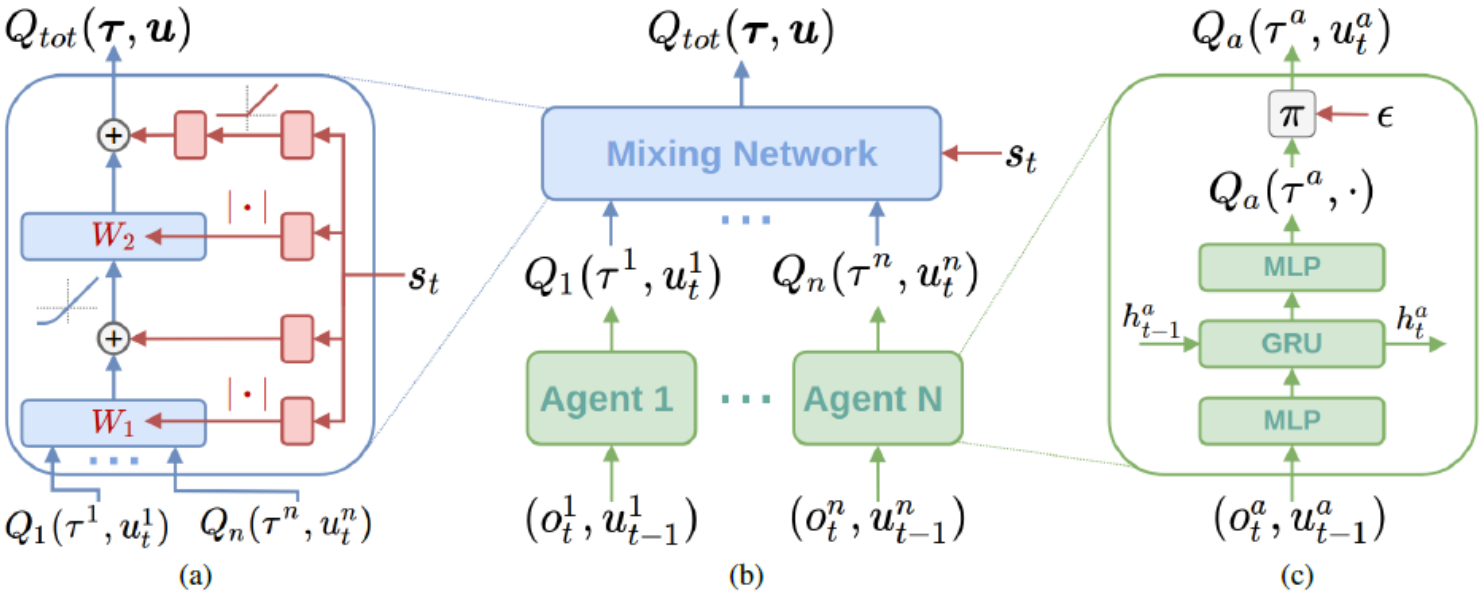
1. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning (2018, ICML)
   - ✓ QMIX 알고리즘의 전반적인 아키텍처
   - ✓ Q-Value 기반 알고리즘으로 Centralized Training Decentralized Execution (CTDE) 패러다임을 따름
   - ✓ 각 에이전트는 학습 시 공동 Q-Value 학습/실행 시 자신의 관측 정보에만 의존, 독립적으로 실행



Deep Reinforcement Learning Research Trends

❖ Multi-Agent Reinforcement Learning

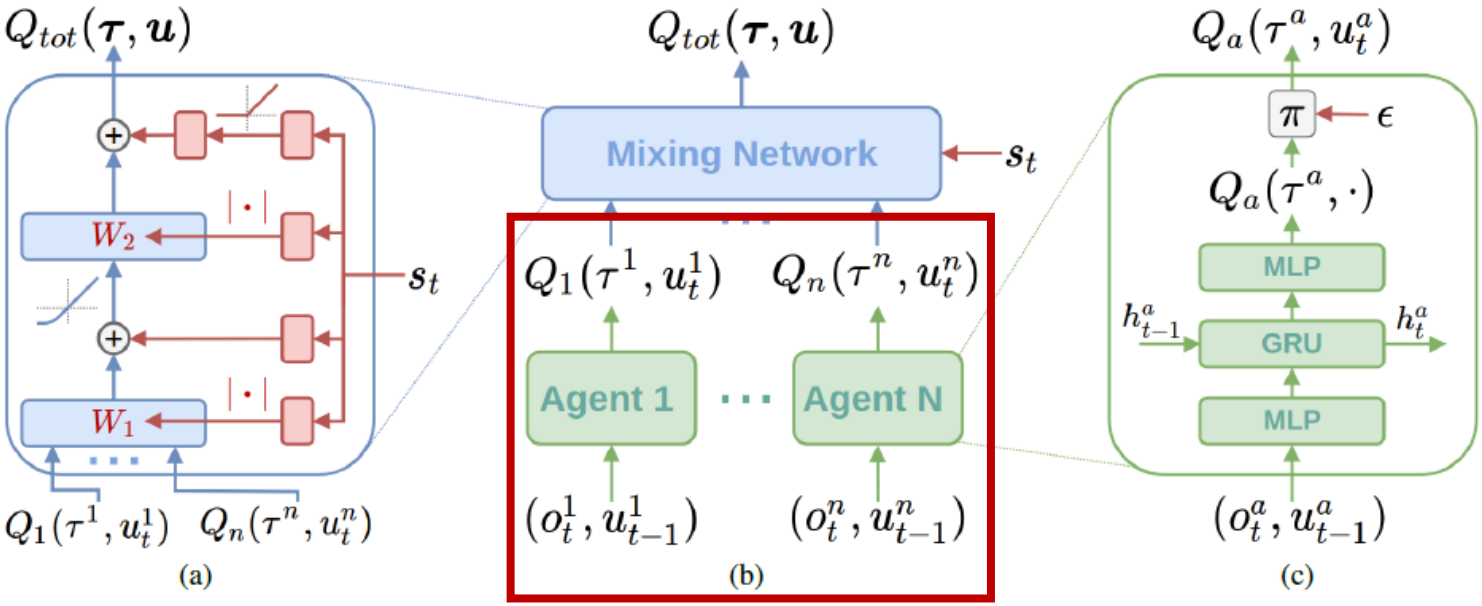
1. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning (2018, ICML)
   - ✓ 멀티 에이전트 강화학습 적용 시 제약조건
   - ✓ 1) 모든 에이전트는 부분 관측 정보(Partial Observation) 이용/서로 공유하지 않음
   - ✓ 2) 보상은 각 에이전트의 개개인이 아닌 팀 보상으로 주어짐



Deep Reinforcement Learning Research Trends

❖ Multi-Agent Reinforcement Learning

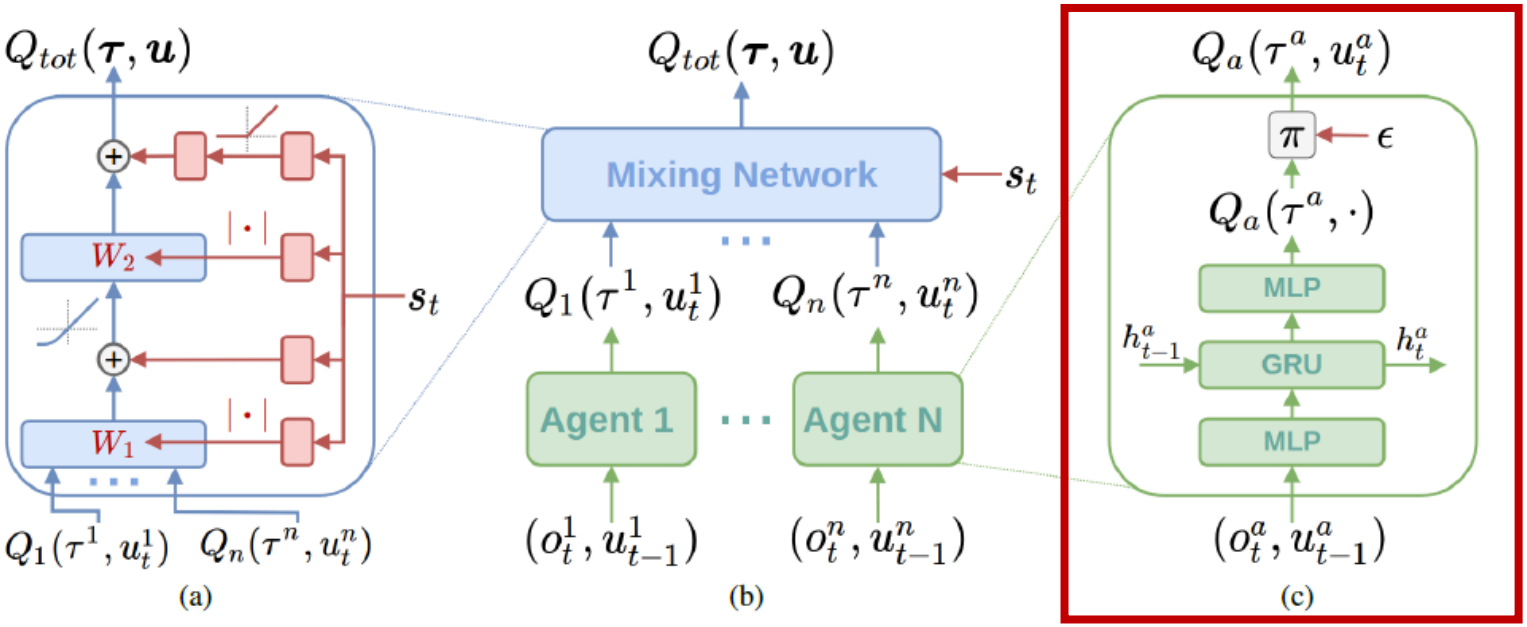
1. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning (2018, ICML)
   - ✓ (b)는 N명의 에이전트를 표현
   - ✓ (b)에서 에이전트별로 부분 관측 정보 o_t^n으로부터 $Q_n(\tau^n, u_t^n)$ 보상 정보를 출력



Deep Reinforcement Learning Research Trends

❖ Multi-Agent Reinforcement Learning

1. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning (2018, ICML)
   - ✓ 에이전트 간 협동(의사소통)을 위해 (c)와 같이 **순환 신경망(RNNs)**을 사용하여 정보 공유
   - ✓ 순환 신경망은 Gated Recurrent Unit (GRU) 셀을 사용

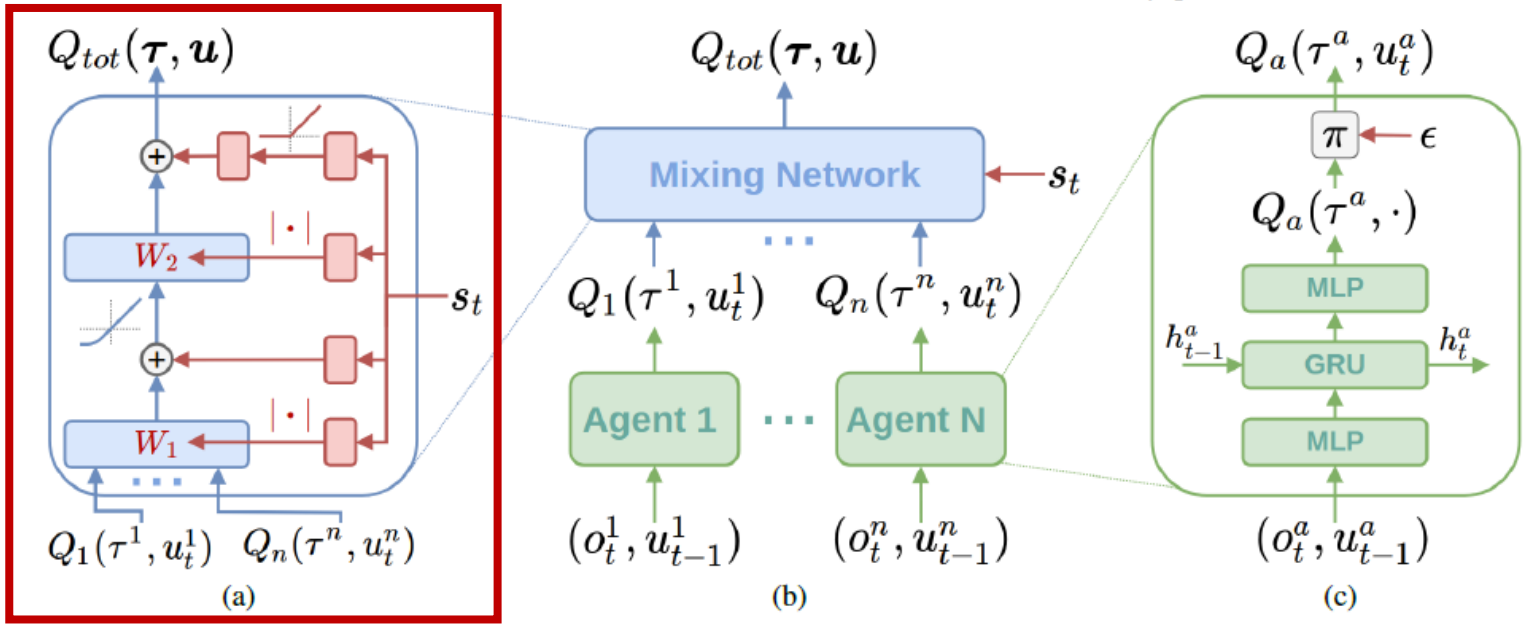


Deep Reinforcement Learning Research Trends

❖ Multi-Agent Reinforcement Learning

1. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning (2018, ICML)
   - ✓ 팀 보상으로 주어지는 제약조건을 만족하기 위해 (a) Mixing Network 적용
   - ✓ Mixing Network는 에이전트별 $Q_n(\tau^n, u_t^n)$와 환경 전체 상태 정보 s_t를 입력 받음
   - ✓ 최종 산출된 $Q_{tot}(\tau, u)$로 Centralized Training 수행

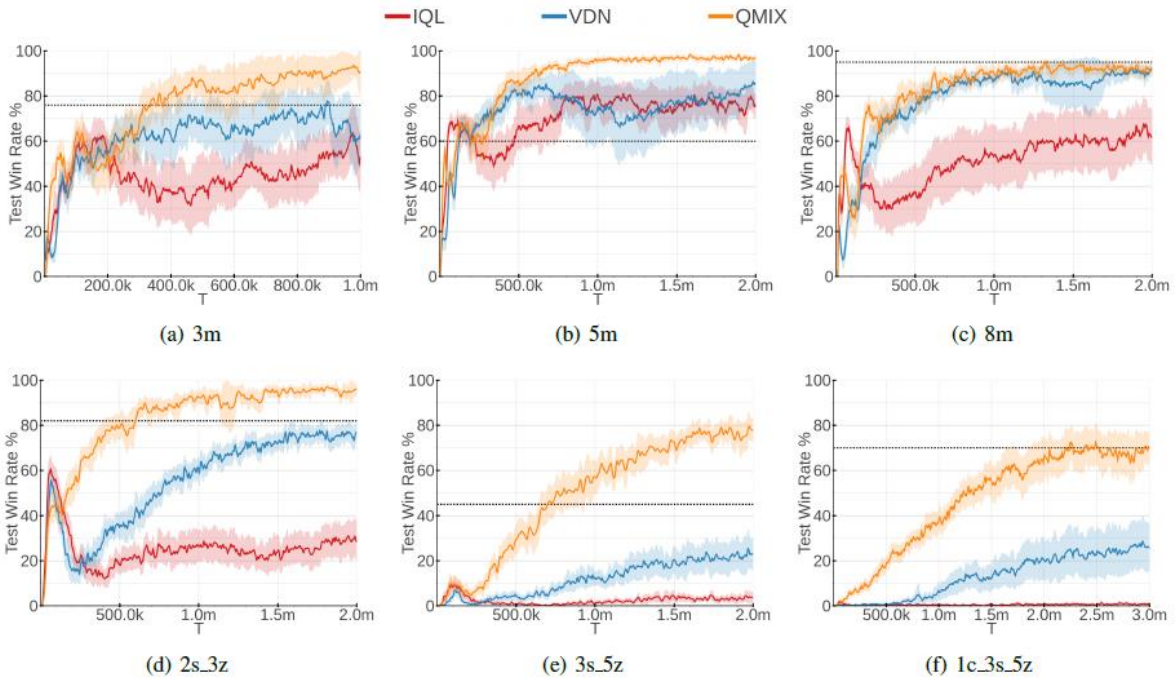
$$\mathcal{L}(\theta) = \sum_{i=1}^{b} \left[\left(y_i^{tot} - Q_{tot}(\boldsymbol{\tau}, \mathbf{u}, s; \theta) \right)^2 \right]$$



Deep Reinforcement Learning Research Trends

❖ Multi-Agent Reinforcement Learning

1. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning (2018, ICML)
   - ✓ 여러 에이전트를 다루는 StarCraft II 환경을 사용
   - ✓ 6개의 시나리오에 기존 방법론과 비교하여 우수함을 입증



Deep Reinforcement Learning Research Trends

- Multi-Agent Reinforcement Learning

  2. RODE: Learning Roles to Decompose Multi-Agent Tasks (2021, ICLR)

Published as a conference paper at ICLR 2021

RODE: LEARNING ROLES TO DECOMPOSE MULTI-AGENT TASKS

Tonghan Wang[†], **Tarun Gupta**[‡], **Anuj Mahajan**[‡], **Bei Peng**[‡]
[†] Institute for Interdisciplinary Information Sciences, Tsinghua University
[‡] Univeristy of Oxford
tonghanwang1996@gmail.com
{tarun.gupta, anuj.mahajan, bei.peng}@cs.ox.ac.uk

Shimon Whiteson[*]
Univeristy of Oxford
shimon.whiteson@cs.ox.ac.uk

Chongjie Zhang[*]
Tsinghua University
chongjie@tsinghua.edu.cn

ABSTRACT

Role-based learning holds the promise of achieving scalable multi-agent learning by decomposing complex tasks using roles. However, it is largely unclear how to efficiently discover such a set of roles. To solve this problem, we propose to first decompose joint action spaces into restricted role action spaces by clustering actions according to their effects on the environment and other agents. Learning a role selector based on action effects makes role discovery much easier because it forms a bi-level learning hierarchy: the role selector searches in a smaller role space and at a lower temporal resolution, while role policies learn in significantly reduced primitive action-observation spaces. We further integrate information about action effects into the role policies to boost learning efficiency and policy generalization. By virtue of these advances, our method (1) outperforms the current state-of-the-art MARL algorithms on 9 of the 14 scenarios that comprise the challenging StarCraft II micromanagement benchmark and (2) achieves rapid transfer to new environments with three times the number of agents. Demonstrative videos can be viewed at *https://sites.google.com/view/rode-marl*.

Deep Reinforcement Learning Research Trends

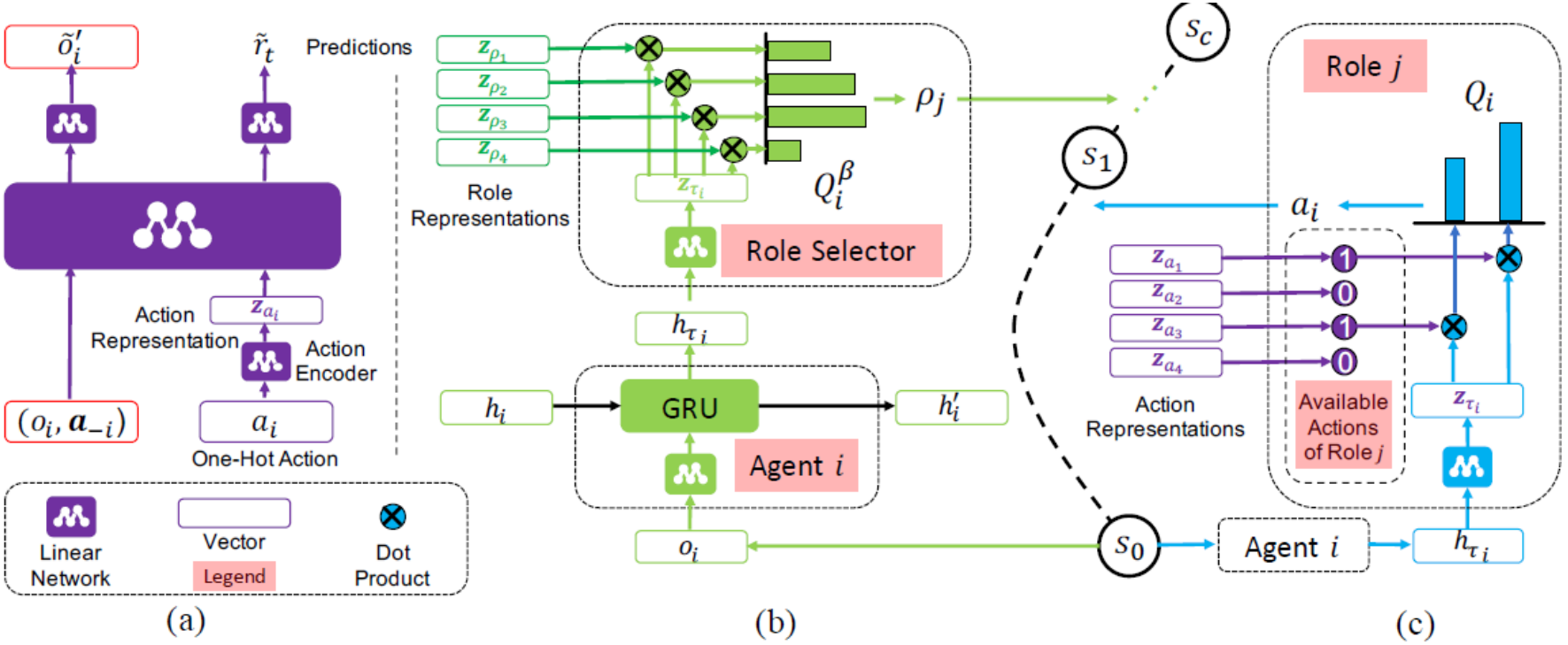
❖ Multi-Agent Reinforcement Learning

2. RODE: Learning Roles to Decompose Multi-Agent Tasks (2021, ICLR)

✓ 효율적인 협력을 위해서는 에이전트마다 적절한 Role이 배정되어야 함

✓ 에이전트들이 행동하는 **Action에 대한 적절한 Role**을 배정하여 효율적인 협력 및 학습하도록 함

✓ Rode에 대한 전반적인 아키텍처



Deep Reinforcement Learning Research Trends

❖ Multi-Agent Reinforcement Learning

2. RODE: Learning Roles to Decompose Multi-Agent Tasks (2021, ICLR)

- ✓ (a) Action Representation 산출
- ✓ 에이전트의 여러 행동에 대한 특징을 파악해야 함
- ✓ 행동에 대한 특징은 Action Encoder를 통해 나온 Action Representation 값
- ✓ 학습 초기 에이전트가 선택한 행동은 올바른 행동이 아닐 수 있어 초기 50K 타임 스텝만큼 (a)의 아키텍처 학습

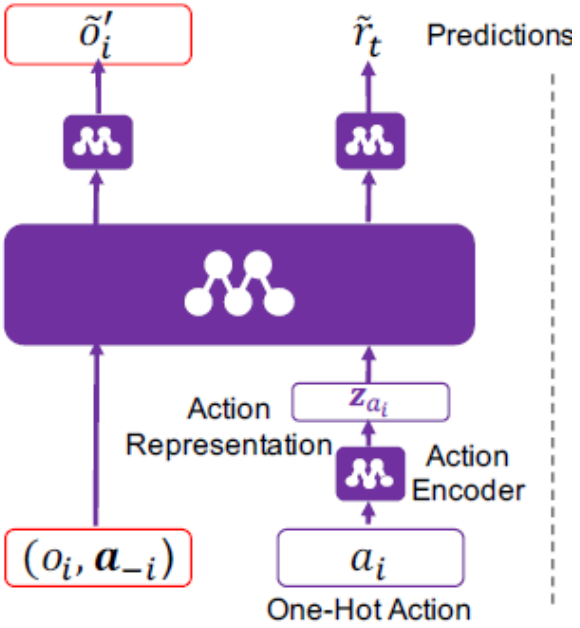
$$\mathcal{L}_e(\theta_e, \xi_e) = \mathbb{E}_{(o,a,r,o')\sim\mathcal{D}}\left[\sum_i \left|\left|p_o(z_{a_i}, o_i, a_{-i}) - o_i'\right|\right|_2^2 + \lambda_e \sum_i \left(p_r(z_{a_i}, o_i, a_{-i}) - r\right)^2\right]$$

- ✓ p_o, p_r: 다음 관측 상태와 보상을 예측하는 Predictor
- ✓ $a_i, (o_i, a_{-i})$를 입력하여 $\tilde{o}_i', \tilde{r}_t$ 예측
- ✓ 위의 손실 함수로 50K 타임 스텝만큼 학습
- ✓ 학습된 Action Representation (z_{a_i})의 정보를 사용하여 군집화
- ✓ 군집 개수만큼 Role을 생성 → **Role은 에이전트별 취할 수 있는 행동이 제한**

Role 1: [1, 0, 1, 1, 1, 0, 0, 0, 0]

Role 2: [0, 0, 0, 0, 0, 1, 1, 0, 0]

Role 3: [1, 0, 0, 1, 1, 0, 0, 0, 1]



Deep Reinforcement Learning Research Trends

❖ Multi-Agent Reinforcement Learning

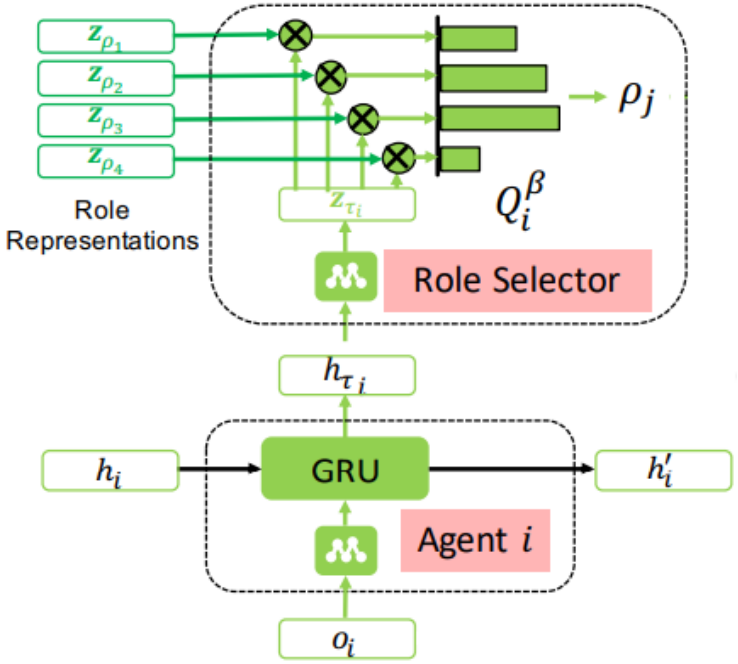
2. RODE: Learning Roles to Decompose Multi-Agent Tasks (2021, ICLR)

- ✓ (b) Role Selector
- ✓ 에이전트별로 적절한 Role을 배정하는 역할
- ✓ 에이전트의 관측 정보와 GRU 셀의 은닉 정보로 함축된 정보 축력
- ✓ 함축된 정보와 Role Representations 값과 내적 및 가치 산출

$$\mathcal{L}_\beta\left(\theta_{\tau_\beta}, \theta_\beta, \xi_\beta\right) = \mathbb{E}_{\mathcal{D}}\left[\left(\sum_{t'=0}^{c-1} r_{t+t'} + \gamma \max_{\rho'} \bar{Q}^\beta_{tot}(s_{t+c}, \rho') - Q^\beta_{tot}(s_t, \rho_t)\right)^2\right]$$

- ✓ $z_{\rho_j} = \frac{1}{|A_j|}\sum_{a_k \in A_j} z_{a_k}$ → Role Representations 값 계산식(j는 role 종류 인덱스)
- ✓ $Q_i^\beta(\tau_i, \rho_i) = z_{t_i}^T z_{\rho_j}$ → 함축된 정보와 Role Representations 값 내적 및 Role별로 가치 산출
- ✓ Role별로 가치를 학습하기 위해 QMIX의 Mixing Network를 활용

Role Q Value	Role 1 Q Value	Role 2 Q Value	Role 3 Q Value
Agent 1	0.087	0.213	0.124
Agent 2	0.321	0.025	0.145
Agent 3	0.115	0.047	0.234



Deep Reinforcement Learning Research Trends

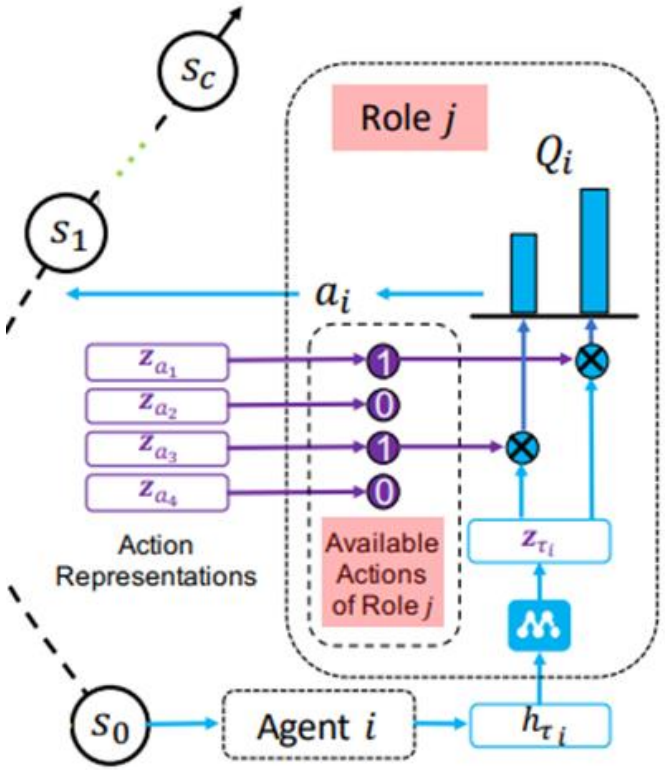
❖ Multi-Agent Reinforcement Learning

2. RODE: Learning Roles to Decompose Multi-Agent Tasks (2021, ICLR)

- ✓ (c) Action Selector
- ✓ 에이전트별로 배정된 Role을 통해 최종 Action을 선택
- ✓ 에이전트의 관측 정보와 GRU 셀의 은닉 정보로 함축된 정보 출력
- ✓ 함축된 정보와 Role에 따라 제한된 행동들의 Q-Value를 축력

$$\mathcal{L}_\rho\left(\theta_{\tau_\rho}, \theta_\rho, \xi_\rho\right) = \mathbb{E}_{\mathcal{D}}\left[\left(r + \gamma \max_{a'} \bar{Q}_{tot}(s', a') - Q_{tot}(s, a)\right)^2\right]$$

- ✓ $Q_i\ (\tau_i, a_k) = z_{\tau_i}^T z_{a_k}$ → 함축된 정보와 Role 기준 Action Representations 값과 내적을 통해 가치 산출
- ✓ 에이전트별 가치를 학습하기 위해 QMIX의 Mixing Network를 활용



Deep Reinforcement Learning Research Trends

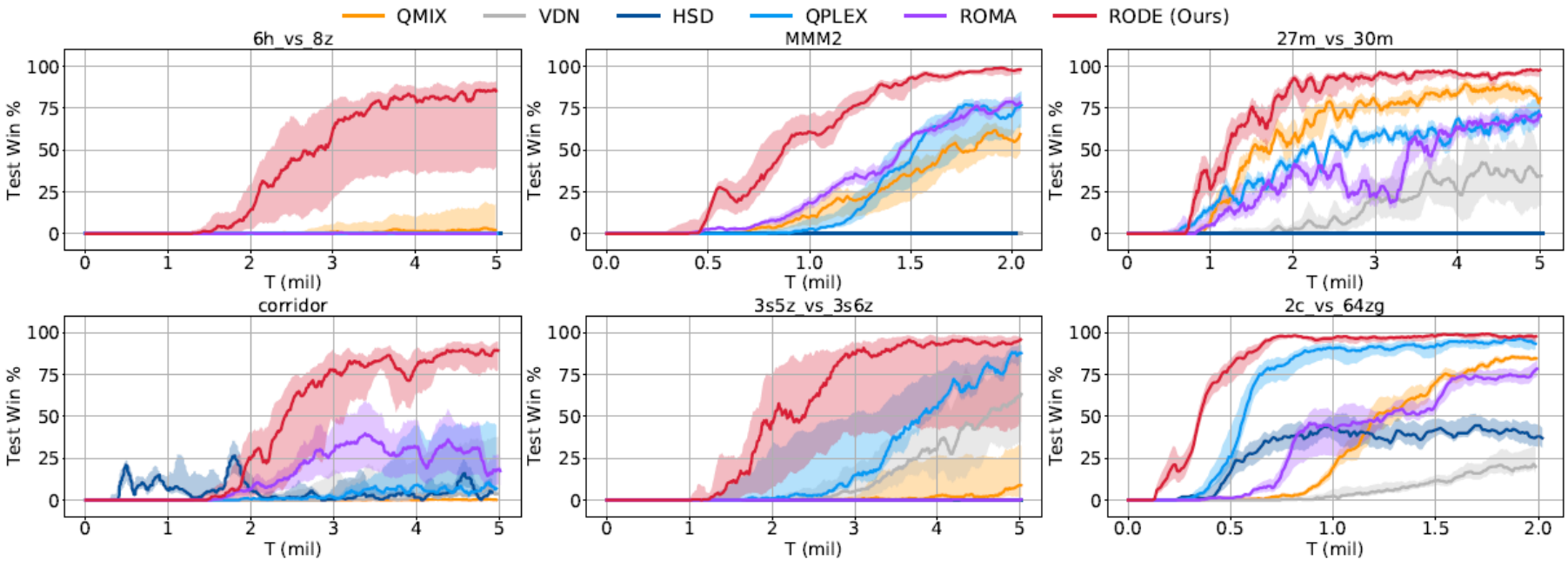
❖ Multi-Agent Reinforcement Learning

2. RODE: Learning Roles to Decompose Multi-Agent Tasks (2021, ICLR)

✓ 여러 에이전트를 다루는 StarCraft II 환경을 사용

✓ 6개의 어려운 StarCraft II 시나리오에 QMIX를 포함한 기존 방법론과 비교하여 우수함을 입증

✓ 적은 타임 스텝만으로 높은 승률을 달성



Conclusion

❖ 결론

- 딥러닝의 발전으로 심층 강화학습 연구가 매우 활발히 이루어지고 있음
- 심층 강화학습은 수많은 데이터를 사용하여 에이전트의 최적화된 정책을 찾음
- 실생활(Real-world)에 실용적이기 위해 여러 문제점들을 해결하는 연구들이 이루어짐
- Self-Supervised Learning, Data Augmentation, Multi-Task Learning 등 다양한 방법론을 강화학습과 결합하여 문제를 해결
- 소개한 연구 트렌드 외에도 Offline RL과 Multi-Task Learning 또는 Multi-Agent RL을 결합, 강화학습과 Meta Learning을 결합하는 연구들도 있음
- 심층 강화학습은 다양한 문제점이 존재하고 이를 해결하기 위한 많은 방법론들이 연구되고 있는 아주 **Hot**한 분야

References

DMQA_YouTube_Seminar

문석호(Self-Supervised Learning): http://dmqa.korea.ac.kr/activity/seminar/302

곽민구(Towards Contrastive Learning): http://dmqa.korea.ac.kr/activity/seminar/308

김재훈(Dive into BYOL): http://dmqa.korea.ac.kr/activity/seminar/310

조억(Cooperative Multi-Agent Reinforcement Learning Framework for Scalping Trading): http://dmqa.korea.ac.kr/activity/seminar/277

강화학습: http://dmqa.korea.ac.kr/activity/seminar?p=1&s%3Dreinforcement%20Learning

Self-Supervised Learning

MoCo: He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 9729-9738).

SimCLR: Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020, November). A simple framework for contrastive learning of visual representations. In International conference on machine learning (pp. 1597-1607). PMLR.

BYOL: Grill, J. B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., ... & Valko, M. (2020). Bootstrap your own latent: A new approach to self-supervised learning. arXiv preprint arXiv:2006.07733.

Reinforcement Learning

Rainbow: Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., ... & Silver, D. (2018, April). Rainbow: Combining improvements in deep reinforcement learning. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 32, No. 1).

Efficient Rainbow: van Hasselt, H., Hessel, M., & Aslanides, J. (2019). When to use parametric models in reinforcement learning?. arXiv preprint arXiv:1906.05243.

References

Representation Learning for Reinforcement Learning

CURL: Laskin, M., Srinivas, A., & Abbeel, P. (2020, November). Curl: Contrastive unsupervised representations for reinforcement learning. In International Conference on Machine Learning (pp. 5639-5650). PMLR..

SPR: Schwarzer, M., Anand, A., Goel, R., Hjelm, R. D., Courville, A., & Bachman, P. Data-Efficient Reinforcement Learning with Self-Predictive Representations.

SGI: Schwarzer, M., Rajkumar, N., Noukhovitch, M., Anand, A., Charlin, L., Hjelm, R. D., ... & Courville, A. (2021). Pretraining reward-free representations for data-efficient reinforcement learning. In Self-Supervision for Reinforcement Learning Workshop-ICLR (Vol. 2021).

Generalization in Reinforcement Learning

RAD: Laskin, M., Lee, K., Stooke, A., Pinto, L., Abbeel, P., & Srinivas, A. (2020). Reinforcement learning with augmented data. Advances in neural information processing systems, 33, 19884-19895.

SODA: Hansen, N., & Wang, X. (2021, May). Generalization in reinforcement learning by soft data augmentation. In 2021 IEEE International Conference on Robotics and Automation (ICRA) (pp. 13611-13617). IEEE.

Offline Reinforcement Learning

CQL: Kumar, A., Zhou, A., Tucker, G., & Levine, S. (2020). Conservative q-learning for offline reinforcement learning. Advances in Neural Information Processing Systems, 33, 1179-1191.

Blog: https://ropiens.tistory.com/144

그림: Lee, S., Seo, Y., Lee, K., Abbeel, P., & Shin, J. (2020). Addressing Distribution Shift in Online Reinforcement Learning with Offline Datasets.

그림: https://bair.berkeley.edu/blog/2020/12/07/offline/

그림: Levine, S., Kumar, A., Tucker, G., & Fu, J. (2020). Offline reinforcement learning: Tutorial, review, and perspectives on open problems. arXiv preprint arXiv:2005.01643.

그림: https://sites.google.com/view/d4rl/home

References

Multi-Task Reinforcement Learning

Distral: Teh, Y., Bapst, V., Czarnecki, W. M., Quan, J., Kirkpatrick, J., Hadsell, R., ... & Pascanu, R. (2017). Distral: Robust multitask reinforcement learning. Advances in neural information processing systems, 30.

Multi-Agent Reinforcement Learning

QMIX: Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., & Whiteson, S. (2018, July). QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In International Conference on Machine Learning (pp. 4295-4304). PMLR.

RODE: Wang, T., Gupta, T., Mahajan, A., Peng, B., Whiteson, S., & Zhang, C. RODE: Learning Roles to Decompose Multi-Agent Tasks. In International Conference on Learning Representations.


Thank
you

Appendix

- Policy Gradient

❖ 수식 유도

- $J(\theta) = E[\sum_{t=0}^{T-1} r_{t+1}|\pi_\theta] = E[r_1 + r_2 + r_3 + \cdots + r_\tau|\pi_\theta]$
- $\theta' = \theta + \alpha\nabla_\theta J(\theta)$, $\nabla_\theta J(\theta)$ = Policy Gradient

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

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

미분 불가능

Appendix

- Policy Gradient

❖ 수식 유도

- $J(\theta) = E[\sum_{t=0}^{T-1} r_{t+1}|\pi_\theta] = E[r_1 + r_2 + r_3 + \cdots + r_\tau|\pi_\theta]$
- $\theta' = \theta + \alpha\nabla_\theta J(\theta)$, $\nabla_\theta J(\theta)$ = Policy Gradient

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

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

미분 가능

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

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

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

Appendix

- Policy Gradient

❖ 수식 유도

- $P(s_t, a_t|\tau) = P(s_0, a_0, r_1, s_1, a_1, r_2, \ldots, s_t, a_t|\theta)$
- $P(s_0)\pi_\theta(a_0|s_0)P(s_1|s_0, a_0)\pi_\theta(a_1|s_1)P(s_2|s_1, a_1) \ldots$
- $logAB = logA + logB$

$$P(\tau|\theta) = P(s_0)\prod_{t=0}^{T-1}\pi_\theta(a_t|s_t)P(s_{t+1}|s_t, a_t)$$

$$logP(\tau|\theta) = logP(s_0) + \sum_{t=0}^{T-1}[log\pi_\theta(a_t|s_t) + logP(s_{t+1}|s_t, a_t)]$$

θ에 대한 미분 → 상태 전이 확률 미분 시 상수 제거

Appendix

- Policy Gradient

❖ 수식 유도

- $P(s_t, a_t|\tau) = P(s_0, a_0, r_1, s_1, a_1, r_2, \ldots, s_t, a_t|\theta)$
- $P(s_0)\pi_\theta(a_0|s_0)P(s_1|s_0, a_0)\pi_\theta(a_1|s_1)P(s_2|s_1, a_1)\ \ldots$
- $logAB = logA + logB$

$$P(\tau|\theta) = P(s_0)\prod_{t=0}^{T-1}\pi_\theta(a_t|s_t)P(s_{t+1}|s_t, a_t)$$

$$logP(\tau|\theta) = logP(s_0) + \sum_{t=0}^{T-1}[log\pi_\theta(a_t|s_t) + logP(s_{t+1}|s_t, a_t)]$$

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

Appendix

- Policy Gradient

❖ 수식 유도

- $P(s_t, a_t|\tau) = P(s_0, a_0, r_1, s_1, a_1, r_2, \ldots, s_t, a_t|\theta)$
- $P(s_0)\pi_\theta(a_0|s_0)P(s_1|s_0, a_0)\pi_\theta(a_1|s_1)P(s_2|s_1, a_1)\ldots$
- $logAB = logA + logB$

$$\nabla_\theta logP(\tau|\theta) = \nabla_\theta \sum_{t=0}^{T-1} log\pi_\theta(a_t|s_t) \xrightarrow{\text{대입}} E_\tau[\sum_{t=0}^{T-1} \nabla_\theta logP(s_t, a_t|\tau) r_{t+1}]$$

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

$$\approx E_\tau[\nabla_\theta \sum_{t=0}^{T-1} log\pi_\theta(a_t|s_t) G_t],\ where\ G_t = \sum_{t=0}^{T-1} \gamma^t r_{t+1} = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots + \gamma^{T-1} r_T$$

Discounted G_t → 단순 보상의 합 발산 방지