# RL^2 : FAST REINFORCEMENT LEARNING VIA SLOW REINFORCEMENT LEARNING

ICLR 2017

**Yan Duan**[†‡], **John Schulman**[†‡], **Xi Chen**[†‡], **Peter L. Bartlett**[†], **Ilya Sutskever**[‡], **Pieter Abbeel**[†‡]
[†] UC Berkeley, Department of Electrical Engineering and Computer Science
[‡] OpenAI

# TL;DR

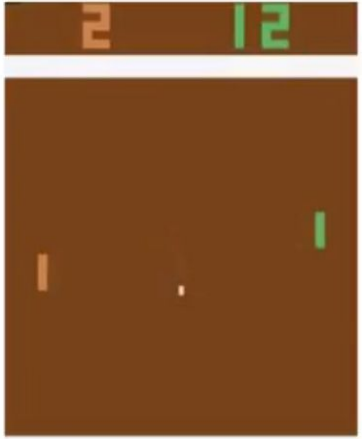
Deep RL (DQN)

Score: **18.9**

experience measured in real time: 40 days

"Slow"

vs.




Human

Score: 9.3

experience measured in real time: **2 hours**

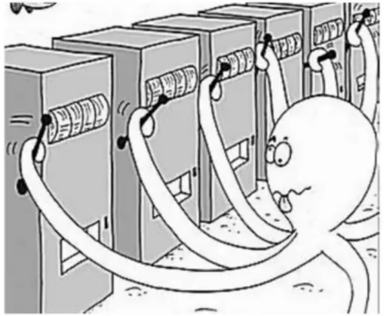
"Fast"

Why : The lack of good prior

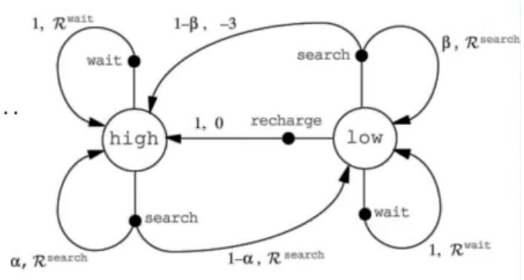
Solution : RNN(GRU)

# Environments

1. Multi-armed bandits



2. Tabular MDPs




3. ViZDoom

# PRELIMINARIES(프리림미너리스)

discrete-time finite-horizeon discounted Markov decision process

$\rho_{\mathcal{M}} : \mathcal{M} \rightarrow \mathbb{R}_{+}$

$$M = (S, A, P, r, \rho 0, \gamma, T)$$

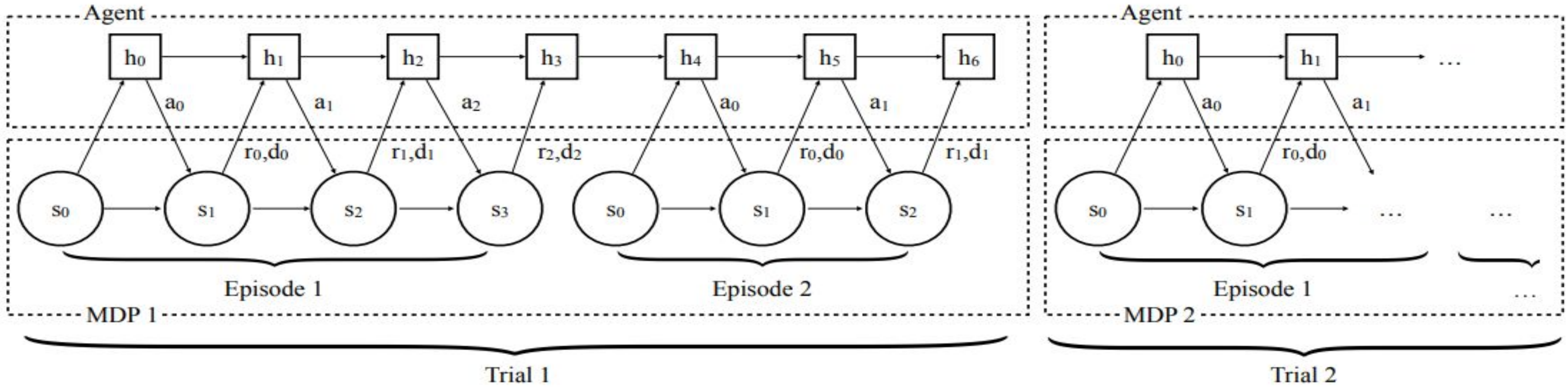
bounded reward $r : S \times A \rightarrow [-R_{max}, R_{max}]$

horizon

maximize $\eta(\pi_{\theta}) = E_{\tau}[\sum_{t=0}^{T} \gamma^{t} r(s_{t}, a_{t})], \text{ where } \tau = (s0, a0, ...)$

# FORMULATION

learning an RL algorithm as a reinforcement learning => RL^2




each trial consist of total episode **n**(=2) terminate flag **d_t** = 1 (if done)

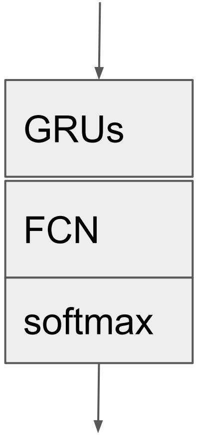
Figure 1: Procedure of agent-environment interaction

maximize the expected total discounted reward accumulated during a single trial rather than a single episode

minimizing the cumulative pseudo-regret 낯선 환경에서 빠른 적응력(=fast learning)을 획득

# POLICY REPRESENTATION & OPTIMIZATION

embedding function $\phi(\ s,\ a,\ r,\ d)$




action distribution

To alleviate the difficulty of training RNNs due to vanishing and exploding gradients we use GRU

TRPO. Why? excellent empirical performance, does not require excessive hyperparameter tuning

To reduce variance => baseline(RNN using GRUs as building blocks), GAE(Generalized Advantage Estimation)

# Evaluation

Q1 : RL^2이 특정한 MDP에서 잘되었던 기존 알고리즘보다 좋을까?

Q2 : RL^2이 high-dimensional task에서도 될까?

Q1 : RL^2이 특정한 MDP에서 잘되었던 기존 알고리즘보다 좋을까?

A : Multi Armed Bandit 와 tabular에서 잘된다면 인정

*MAB : subset of MDPs, agent’s environment is stateless, **k** arms(actions), every time step, agent pulls one of the arms(say *i* ), each arm is Bernoulli dist. with parameter p_i. The key challenge is E&E

We randomly generated MAB by sampling from uniform dist.

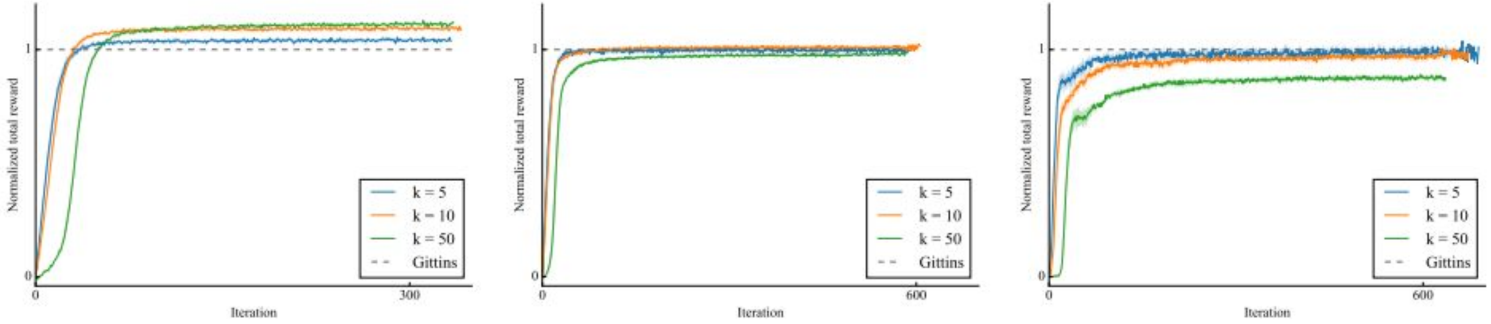
compared against :

Random, Gittins index, UCB1, Thompson sampling(TS, OTS), e-greedy, greedy n = episode

k = arms

total reward

| Setup | Random | Gittins | TS | OTS | UCB1 | $\epsilon$-Greedy | Greedy | RL$^2$ |
|---|---|---|---|---|---|---|---|---|
| $n = 10, k = 5$ | 5.0 | **6.6** | 5.7 | 6.5 | **6.7** | **6.6** | **6.6** | **6.7** |
| $n = 10, k = 10$ | 5.0 | **6.6** | 5.5 | 6.2 | **6.7** | **6.6** | **6.6** | **6.7** |
| $n = 10, k = 50$ | 5.1 | 6.5 | 5.2 | 5.5 | **6.6** | 6.5 | 6.5 | **6.8** |
| $n = 100, k = 5$ | 49.9 | **78.3** | 74.7 | **77.9** | **78.0** | 75.4 | 74.8 | **78.7** |
| $n = 100, k = 10$ | 49.9 | **82.8** | 76.7 | 81.4 | 82.4 | 77.4 | 77.1 | **83.5** |
| $n = 100, k = 50$ | 49.8 | **85.2** | 64.5 | 67.7 | 84.3 | 78.3 | 78.0 | **84.9** |
| $n = 500, k = 5$ | 249.8 | **405.8** | **402.0** | **406.7** | **405.8** | 388.2 | 380.6 | **401.6** |
| $n = 500, k = 10$ | 249.0 | **437.8** | 429.5 | **438.9** | **437.1** | 408.0 | 395.0 | 432.5 |
| $n = 500, k = 50$ | 249.6 | **463.7** | 427.2 | 437.6 | 457.6 | 413.6 | 402.8 | 438.9 |




(a) $n = 10$ (b) $n = 100$ (c) $n = 500$

Figure 2: RL$^2$ learning curves for multi-armed bandits. Performance is normalized such that Gittins index scores 1, and random policy scores 0.

Q1 : RL^2이 특정한 MDP에서 잘되었던 기존 알고리즘보다 좋을까?

Tabular MDPs : MAB가 sequential decision making을 잘 나타내지 못하므로 사용.

compared against :

Random, (O)PSRL, BEB, UCRL2, e-greedy, greedy

$$|\mathcal{S}| = 10, |\mathcal{A}| = 5.$$

T = 10

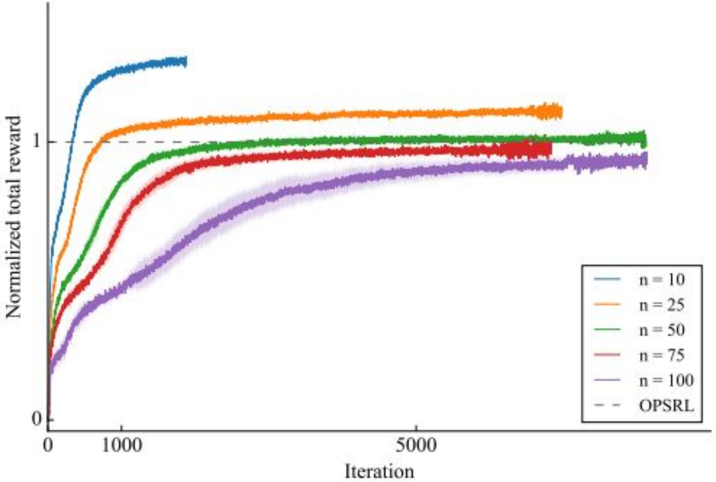
n = episode

Table 2: Random MDP Results

| Setup | Random | PSRL | OPSRL | UCRL2 | BEB | $\epsilon$-Greedy | Greedy | RL$^2$ |
|---|---|---|---|---|---|---|---|---|
| $n = 10$ | 100.1 | 138.1 | 144.1 | 146.6 | 150.2 | 132.8 | 134.8 | **156.2** |
| $n = 25$ | 250.2 | 408.8 | 425.2 | 424.1 | 427.8 | 377.3 | 368.8 | **445.7** |
| $n = 50$ | 499.7 | 904.4 | **930.7** | 918.9 | 917.8 | 823.3 | 769.3 | **936.1** |
| $n = 75$ | 749.9 | 1417.1 | **1449.2** | 1427.6 | 1422.6 | 1293.9 | 1172.9 | 1428.8 |
| $n = 100$ | 999.4 | 1939.5 | **1973.9** | 1942.1 | 1935.1 | 1778.2 | 1578.5 | 1913.7 |

reward is Gaussian dist with unit variance, mean is sampled from Normal(1,1)

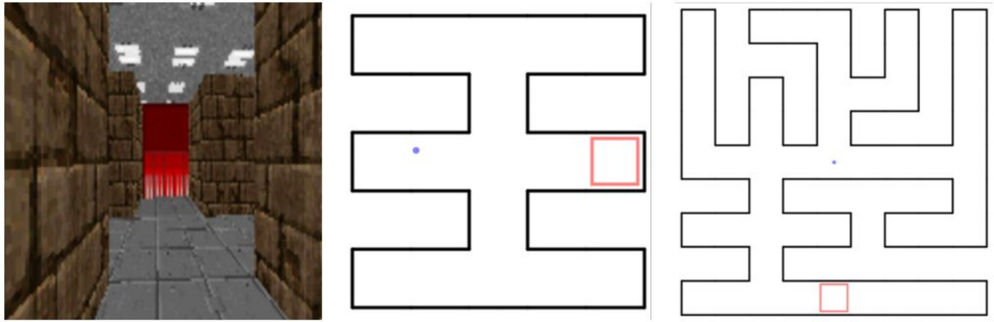
Transitions are sampled from flat Dirichlet dist.




n이 높으면 잘안됨.

reinforcement learning problem in the outer loop becomes more challenging to solve

## Q2 : RL^2이 high-dimensional task에서도 될까? + POMDP



(a) Sample observation (b) Layout of the 5 × 5 maze in (a) (c) Layout of a 9 × 9 maze

Figure 4: Visual navigation. The target block is shown in red, and occupies an entire grid in the maze layout.

During multiple episodes, maze structure and target position are held fixed.

During each trial, we sample 1 out of 1000 randomly generated configurations of map layout and target positions

**Reward** : +1 reaches the target, -0.001 when hit the wall, -0.04 per time step. (Sparse reward)

**The optimal strategy** : explore the maze efficiently during the first episode,

and after locating the target, act optimally against the current maze and target based on the collected information

Table 3: Results for visual navigation. These metrics are computed using the best run among all runs shown in Figure 5. In 3c, we measure the proportion of mazes where the trajectory length in the second episode does not exceed the trajectory length in the first episode.

(a) Average length of successful trajectories

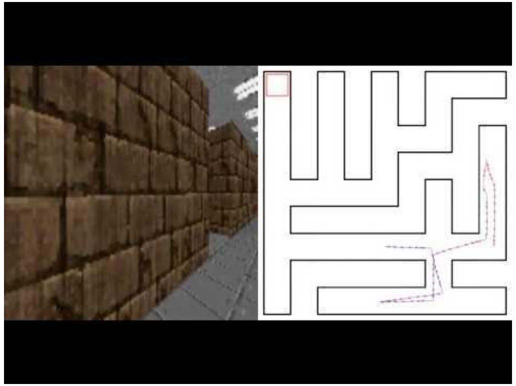
| Episode | Small | Large |
|---|---|---|
| 1 | $52.4 \pm 1.3$ | $180.1 \pm 6.0$ |
| 2 | $39.1 \pm 0.9$ | $151.8 \pm 5.9$ |
| 3 | $42.6 \pm 1.0$ | $169.3 \pm 6.3$ |
| 4 | $43.5 \pm 1.1$ | $162.3 \pm 6.4$ |
| 5 | $43.9 \pm 1.1$ | $169.3 \pm 6.5$ |

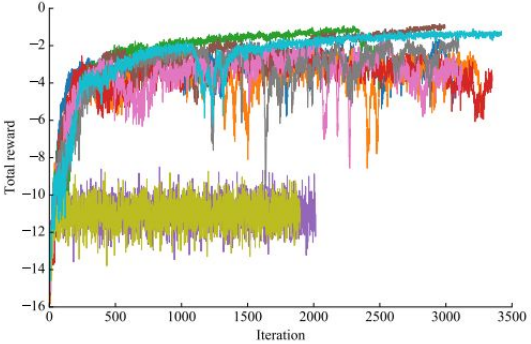
(b) %Success

| Episode | Small | Large |
|---|---|---|
| 1 | 99.3% | 97.1% |
| 2 | 99.6% | 96.7% |
| 3 | 99.7% | 95.8% |
| 4 | 99.4% | 95.6% |
| 5 | 99.6% | 96.1% |

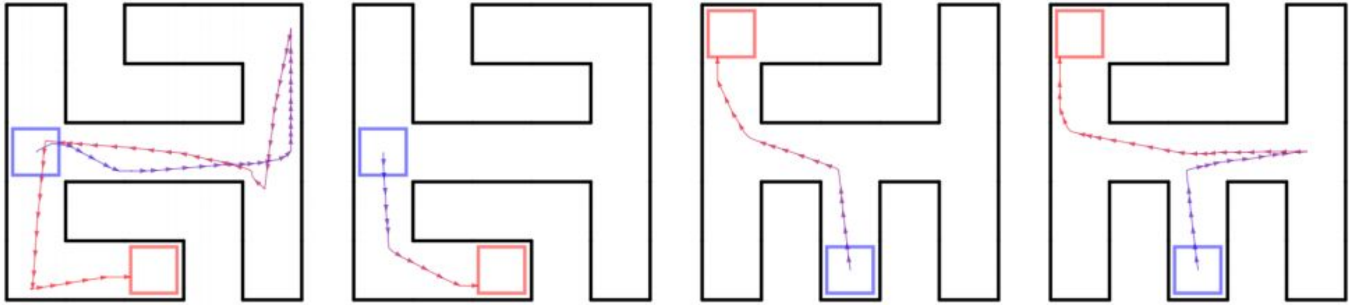
(c) %Improved

| Small | Large |
|---|---|
| 91.7% | 71.4% |






each curve different random initialization of the RNN



(a) Good behavior, 1st episode (b) Good behavior, 2nd episode (c) Bad behavior, 1st episode (d) Bad behavior, 2nd episode

Figure 6: Visualization of the agent's behavior. In each scenario, the agent starts at the center of the blue block, and the goal is to reach anywhere in the red block.

a,b는 과거 정보를 잘 활용했으나 c,d는 과거정보를 활용못함.

논문에선 outer-loop에서 더좋은 RL techniques가 해결해줄거라고 믿음

# Related work

1. Auto tuning of hyperparameters (Ishii et al., 2002)
2. Taylor & Stone (2009) survey the multi-task and transfer learning aspects
3. Fu et al. (2015) propose a model-based approach on top of iLQG with unknown dynamics (Levine & Abbeel, 2014), 이전 task에서 얻은 sample을 새로운 task에서 one-shot learning, but related tasks thanks to reduced sample complexity.
4. Deep neural networks for multi-task learning and transfer learning (Parisotto et al., 2015; Rusu et al., 2015; 2016a; Devin et al., 2016; Rusu et al., 2016b)
5. Our work draws inspiration from formulates meta-learning as an optimization problem, and can thus be optimized end-to-end via gradient descent(2016)