Deep Recurrent Q-Learning for Partially Observable MDPs

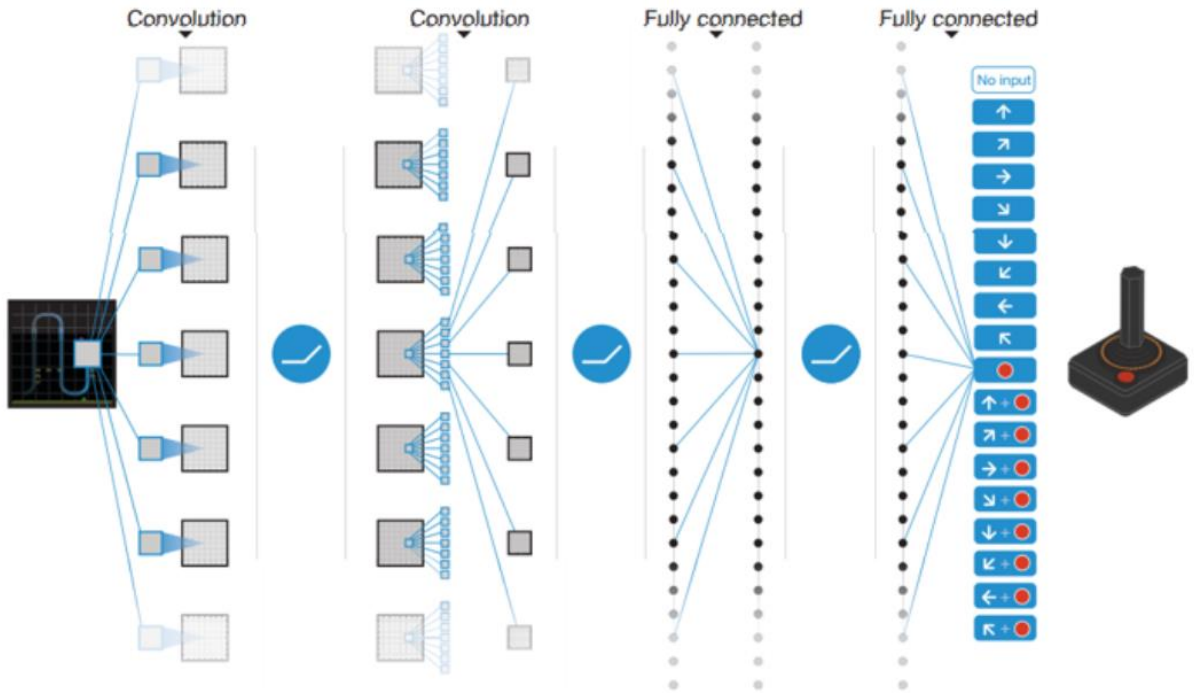
Matthew Hausknecht and Peter Stone

2015 aaai fall symposium series

Seoyoung Lee
2021.12.23

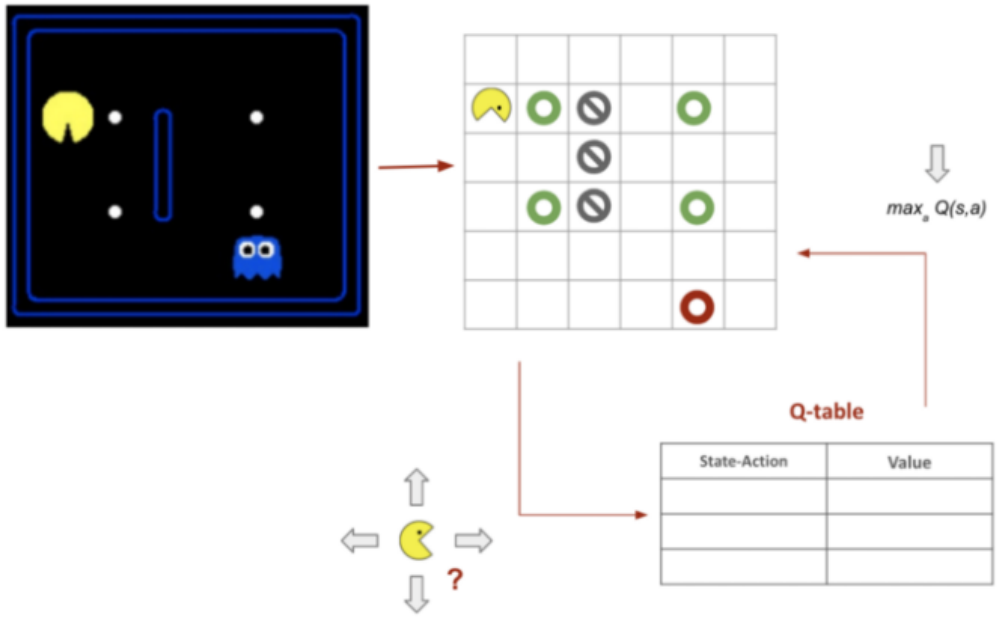
01. Introduction

- 이전 논문에서 제안한 DQN이 연속적인 State를 알아야 한다는 문제의 해결을 위해 recurrent neural network을 적용하여 개선하고자 함



02. Background

Q-learning



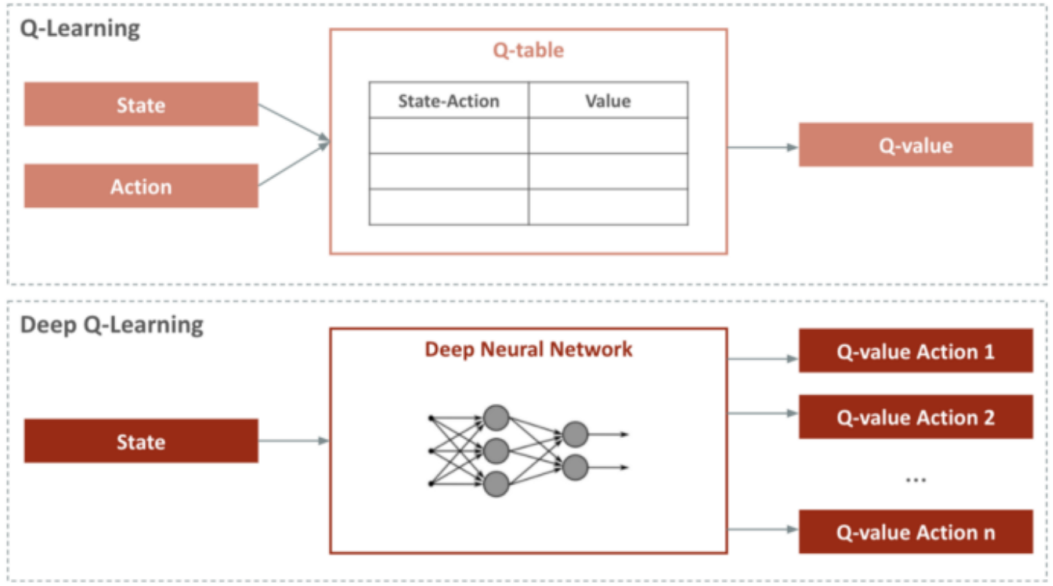
- Q-learning은 모델 없이 학습하는 강화 학습 기법
- 주어진 상태 s에서 특정한 행동 a를 취하는 것이 가져올 보상의 기댓값을 예측하는 Q(s,a)를 학습

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\Big(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \Big)}^{\text{temporal difference}}$$

$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{new value (temporal difference target)}}$$

02. Background

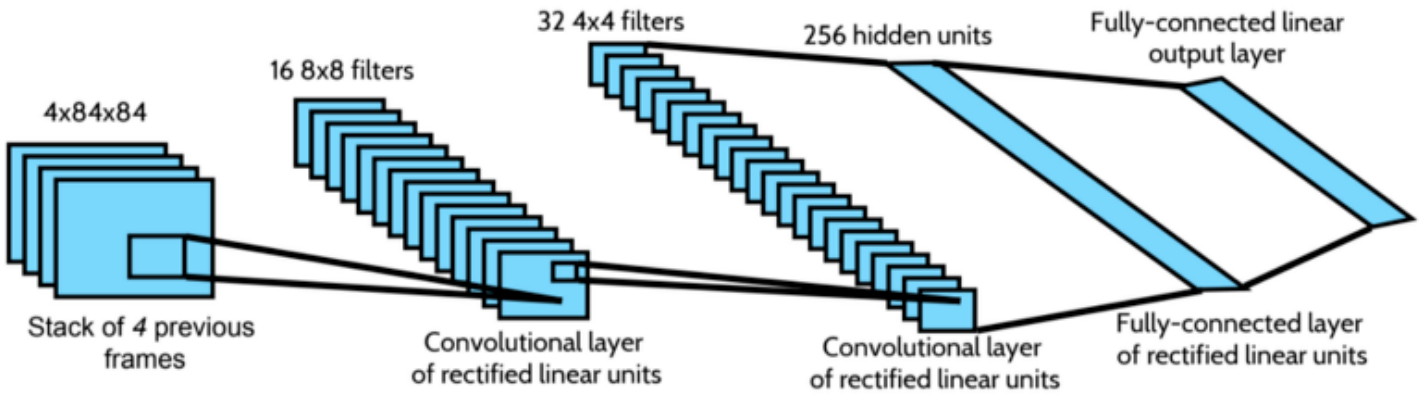
Deep Q-Network

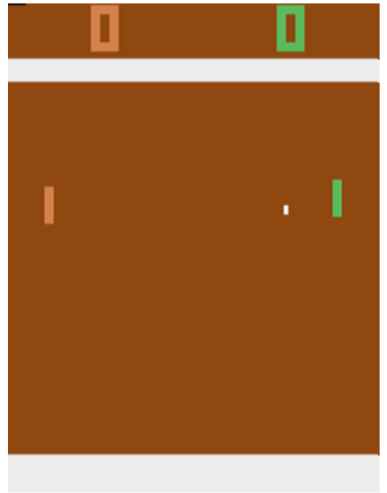


- Q-learning은 연속적이고 큰 State에 대해서는 비효율적
- 딥러닝 모델을 사용하여 Q함수를 학습하여 주어진 State에서 최적의 Action을 도출하고자 함
- Playing Atari with Deep Reinforcement Learning논문에서 DQN을 통해 다양한 Atari 게임에서 사람 수준의 control policy를 학습 했음을 밝힘

02. Background

Limit of DQN



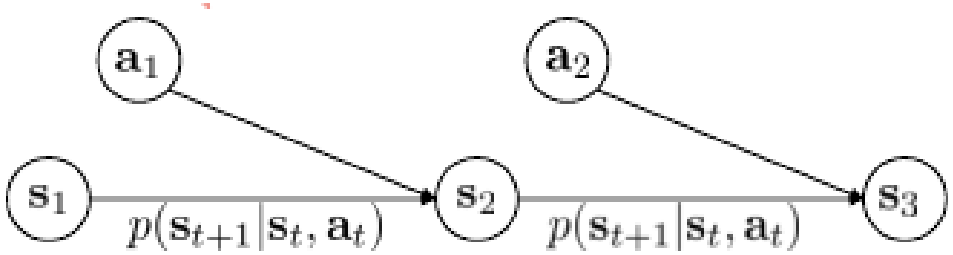


- DQN의 단점은 제한된 길이의 과거 상태에서 학습한다는 것

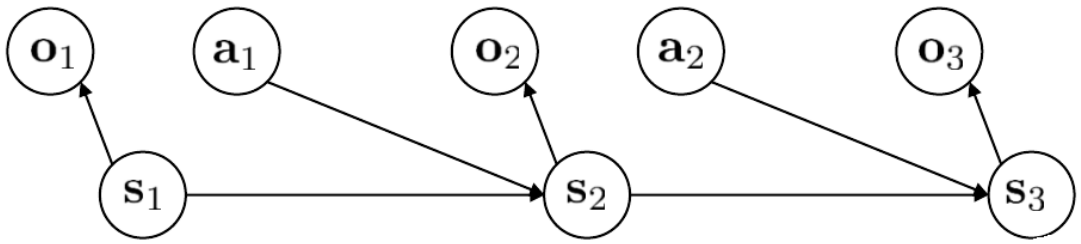
- 해당 길이 이상의 과거의 정보를 필요로 하는 게임들은 POMDP(Partially Observable MDP)가 되어 DQN 적용 불가능

02. Background

POMDP



$$\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{T}, r\}$$

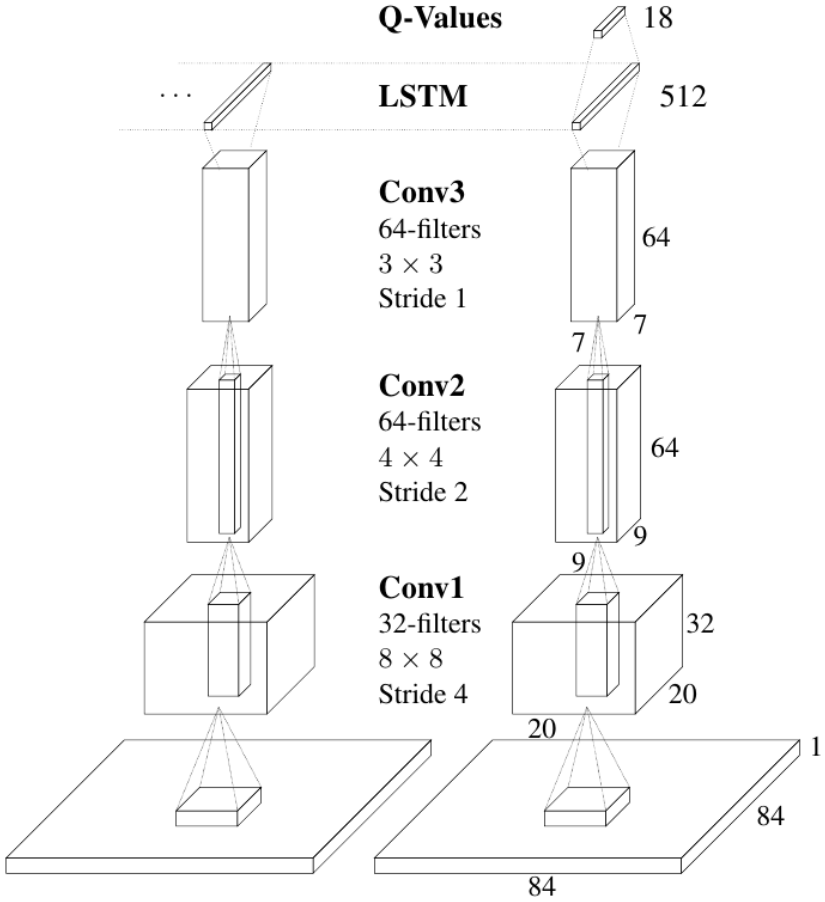


$$\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T}, \mathcal{E}, r\}$$

- 실 세계에서의 Task들은 Agent에게 현재 상태에 대한 모든 정보를 제공해주기 어려움
- State의 일부 정보만을 가지는 Observation이라는 개념이 추가됨

03. Deep Recurrent Q-Network

Network Architecture



- 기존의 첫 번째 fully connected layer 대신 LSTM을 사용
- 한 번에 4프레임을 입력으로 넣던 DQN과 달리 단일 프레임을 입력으로 사용

03. Deep Recurrent Q-Network

Stable Recurrent Updates

- Bootstrapped Sequential Updates
  - ➢ replay memory에서 episode를 random하게 선택을 하고 해당 episode의 시작 부분에서 부터 쭉 학습을 진행

- Bootstrapped Random Updates
  - ➢ replay memory에서 episode를 random하게 선택 하고 해당 episode 내에서도 random하게 시작 지점을 선택 및 network가 update 되는 시점까지만 학습

03. Deep Recurrent Q-Network

Atari Games: MDP or POMDP?

(a) Pong (b) Frostbite (c) Double Dunk

- 단일 화면에 대해서는 POMDP가 성립하나 4프레임의 화면을 통해 State를 구성하면 MDP가 성립

03. Deep Recurrent Q-Network

Flickering Atari Games

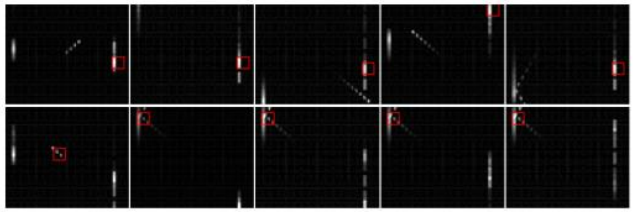
- 확률 p =0.5에 따라 게임 화면이 가려지게 수정함
- 약 절반 정도의 프레임이 가려지게 되므로 agent가 State의 전체 정보를 알 수 없게 되어 POMDP가 성립
- Flickering Pong Games의 경우 프레임 전반에 걸쳐 패들과 공의 위치와 속도를 정확하게 추정하는 것이 중요하고, 가려질 수 있는 여러 입력에 대해 영향을 적게 받아야 함

03. Deep Recurrent Q-Network

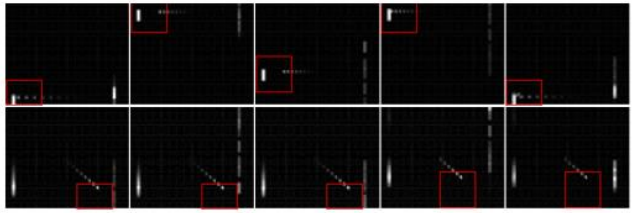
Flickering Atari Games

- DRQN은 time-step 당 1개의 입력 만으로도 물체의 속도를 감지하고 높은 레벨의 pong 이벤트를 감지가능

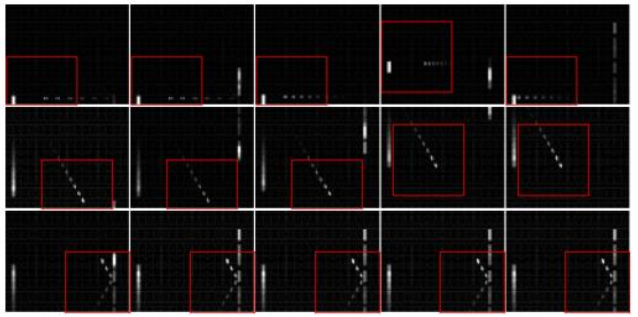
- a, b, c는 10프레임 DQN의 convolution filter, d는 DRQN의 LSTM Unit 시각화



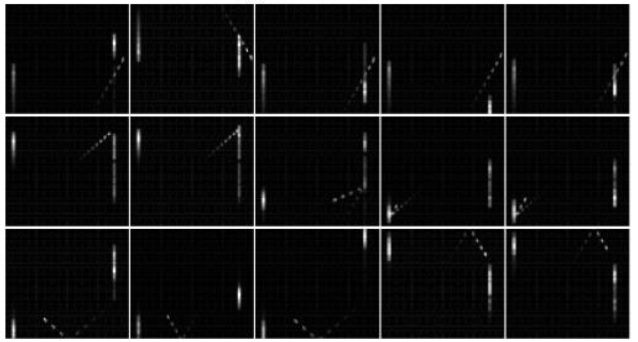
(a) Conv1 Filters



(b) Conv2 Filters



(c) Conv3 Filters



(d) Image sequences maximizing three sample LSTM units

04. Experiments

Evaluation on Standard Atari Games

	DRQN $\pm std$	DQN $\pm std$	
Game		Ours	Mnih et al.
Asteroids	1020 ($\pm$312)	1070 ($\pm$345)	1629 ($\pm$542)
Beam Rider	3269 ($\pm$1167)	**6923** ($\pm$1027)	6846 ($\pm$1619)
Bowling	62 ($\pm$5.9)	72 ($\pm$11)	42 ($\pm$88)
Centipede	3534 ($\pm$1601)	3653 ($\pm$1903)	8309 ($\pm$5237)
Chopper Cmd	2070 ($\pm$875)	1460 ($\pm$976)	6687 ($\pm$2916)
Double Dunk	**-2** ($\pm$7.8)	-10 ($\pm$3.5)	-18.1 ($\pm$2.6)
Frostbite	**2875** ($\pm$535)	519 ($\pm$363)	328.3 ($\pm$250.5)
Ice Hockey	-4.4 ($\pm$1.6)	-3.5 ($\pm$3.5)	-1.6 ($\pm$2.5)
Ms. Pacman	2048 ($\pm$653)	2363 ($\pm$735)	2311 ($\pm$525)

Table 1: On standard Atari games, DRQN performance parallels DQN, excelling in the games of Frostbite and Double Dunk, but struggling on Beam Rider. Bolded font indicates statistical significance between DRQN and our DQN.[5]

- 9가지의 Atari 게임에 대해 평가를 실행
- Flickering이 없는 MDP 환경에서는 DRQN이 DQN을 능가한다고 보기 어려움

04. Experiments

MDP to POMDP Generalization

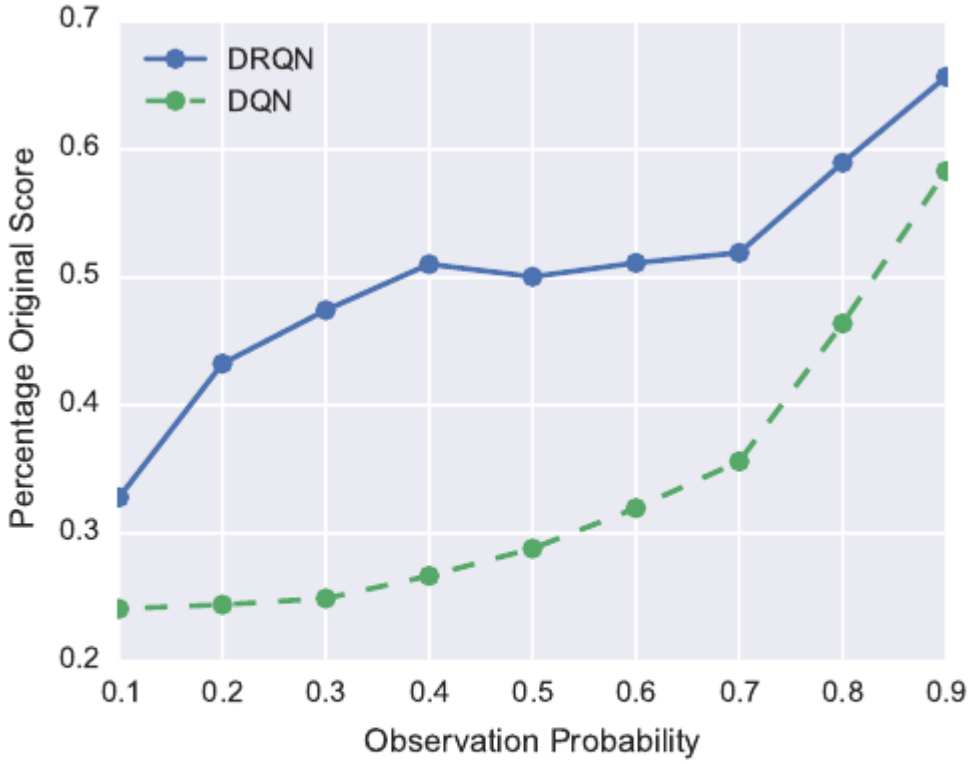


Figure 5: When trained on normal games (MDPs) and then evaluated on flickering games (POMDPs), DRQN's performance degrades more gracefully than DQN's. Each data point shows the average percentage of the original game score over all 9 games in Table 1.

- 일반 MDP환경에서 모델을 학습시키고 Flickering이 존재하는 POMDP환경에서 성능 평가한 결과

- 두 알고리즘 모두 누락된 정보로 인해 성능이 저하되나 DRQN이 비교적 더 많은 정보를 추출하는 것을 확인 가능

05. Conclusion

- DRQN은 각 time-step에서 단일 프레임만 제공 받아도 화면 상의 물체의 속도와 같은 정보를 감지 가능
- POMDP환경의 Pong게임에서 기존 DQN보다 더 적합함
- POMDP에서 학습 후 일반화를 통해 MDP에서도 좋은 성능을 냄
- 일반적인 MDP 환경에서는 DRQN이 사용 가능한 방법이지만 DQN에 비해 이점을 지닌다고 보기 어려움

Thank you