Using Natural Language for Reward Shaping in Reinforcement Learning

Hyungrak Kim

Content

*Abstraction

1. Introduction
2. Overview of the Approach
3. LanguageE-Action Reward Network
4. Using Language-based Reward in RL
5. Experimental Evaluation
6. Conclusion and Future Work

*Abstraction

Abstraction

- Real world environment에서는 reward가 sparse하다 따라서 Reward를 shaping하는 것이 중요함
- Reward가 sparse하면 학습 속도가 느리고, 학습 자체도 잘 안될 가능성 있음
- Reward shaping이 시간이 많이 들어가는 작업이라 쉽지 않음
- 따라서 본 논문에서는 NLI(Natural Language Instruction)를 이용하여 Reward shaping함
- 또한 LanguagE-Action Reward Network를 propose 하는 데 이것은 NLI와 action을 잘 Mapping해서 intermediate reward를 Agent에게 Feedback함
- NLI intermediate reward를 줌으로써 Based Reward+ NLI reward를 사용해 Reward Shaping함
- 실험결과 Montezuma's Revenge 게임에서 task를 해결하는데, 기존 평균보다 60%이상 높았음을 증명

1. Introduction

- 실제 Environment에서는 Sparse 한 Reward가 많다. 그래서 이것을 해결하기 위해서 Intermediate Reward Agent에게 주는데 이를 Reward Shaping이라 함
- Reward Shaping을 디자인하기 어려운 문제가 있음
- 따라서 이 논문에서는 NLI를 이용하여 Intermediate Reward를 줌으로써 Reward Shaping을 함

- **Contribution**
  - LEAN 을 도입함으로써 Agent 학습 속도가 빨라 졌다는 것을 실험으로 증명
- **Natural Language Instruction hard problem**
  - Symbol grounding: 적절한 Instruction mapping이 어려움
    - Ex) Jump over the snake
    - Jump -> action space
    - Snake -> state space(pixel)
  - NLI incomplete
    - 다양한 Goal을 획득할 수 있는 방법이 있다 -> Instruction으로써 취할 수 있는 방법이 많다
  - Natural language ambiguity and variation
    - 언어적으로 문장에 들어있는 정보량과 어려움 정도는 다르다
      - high-level subgoal: “Collect the key”
      - low-level instructions: “Jump over the obstacle. Climb up the ladder and jump to collect the key.”

1. Introduction

- 실제 Environment에서는 Sparse 한 Reward가 많다. 그래서 이것을 해결하기 위해서 Intermediate Reward Agent에게 주는데 이를 Reward Shaping이라 함
- Reward Shaping을 디자인하기 어려운 문제가 있음
- 따라서 이 논문에서는 NLI를 이용하여 Intermediate Reward를 줌으로써 Reward Shaping을 함

- **Contribution**
  - LEAN 을 도입함으로써 Agent 학습 속도가 빨라 졌다는 것을 실험으로 증명
- **Natural Language Instruction hard problem**
  - Symbol grounding: 적절한 Instruction mapping이 어려움
    - Ex) Jump over the snake
    - Jump -> action space
    - Snake -> state space(pixel)
  - NLI incomplete
    - 다양한 Goal을 획득할 수 있는 방법이 있다 -> Instruction으로써 취할 수 있는 방법이 많다
  - Natural language ambiguity and variation
    - 언어적으로 문장에 들어있는 정보량과 어려움 정도는 다르다
      - high-level subgoal: "Collect the key"
      - low-level instructions: "Jump over the obstacle. Climb up the ladder and jump to collect the key."

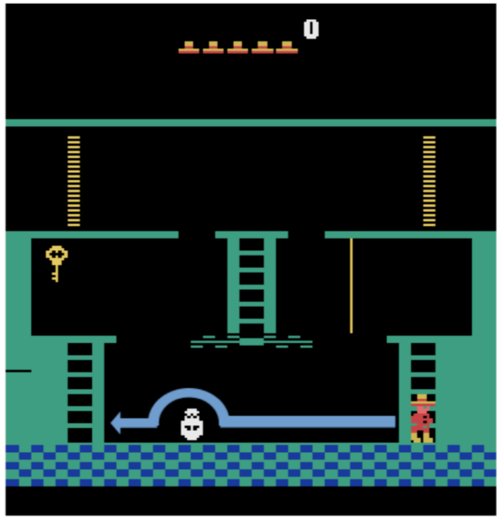
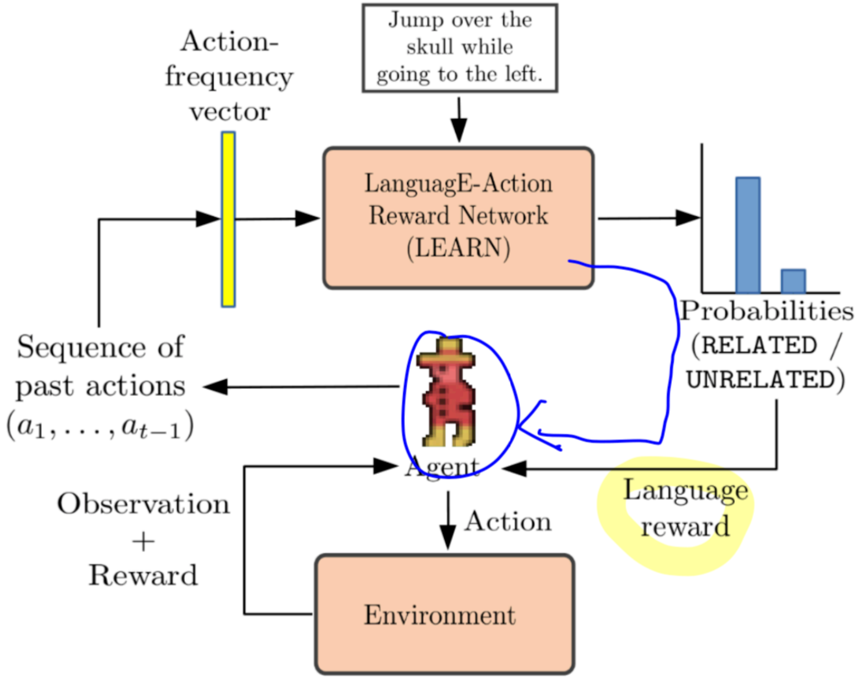


Figure 1: An agent exploring randomly to complete the task described by the blue trajectory may need considerable amount of time to learn the behavior. By giving natural language instructions like "Jump over the skull while going to the left", we can give intermediate signals to the agent for faster learning.

approach

2. Overview of the Approach

- MDP+L -> <S, A, R, T, l, γ>
- L >l is a language command describing the intended behavior (with L defined as the set of all possible language commands).
- Optimal MDP+L를 찾기 위한 two-phase approach
  - LanguagE-Action Reward Network (LEARN)
  - Language-aided RL


Action-frequency vector
Jump over the skull while going to the left.
LanguagE-Action Reward Network (LEARN)
Probabilities (RELATED / UNRELATED)
Sequence of past actions ($a_1, \ldots, a_{t-1}$)
Agent
Language reward
Observation + Reward
Action
Environment

3. LanguageE-Action Reward Network

3.1 Model

- Action-frequency vectors
  - Distinct timesteps i and j (i<j)
  - Let τ[i : j] denote the segment of τ between timesteps i and j
  - f =action frequency vectors
  - τ[i : j]에서 timesteps action k의 fraction
    - Ex) action (left, left, down) -> instruction(matching)
  - (Action frequency vector, Natural Language Instructino) = (f, l)
- Create Dataset
  - Positive Example
    - sampling f from a given trajectory τ and using the language description l associated with τ
  - Negative Example
    - 1. 기존 l를 제외한 Random uniformly sampling
    - 2. Random 하게 f' 만들고, l과 paring
- Language Encoder
  - Infersent -> pretrained sentence embedding model
  - Glove+RNN -> pretrained Glove word embedding + 2 layer GRU Encoder, decoder Layer
  - RNNOnly -> 2 layer GRU Encoder, decoder Layer

3. LanguageE-Action Reward Network

3.1 Model

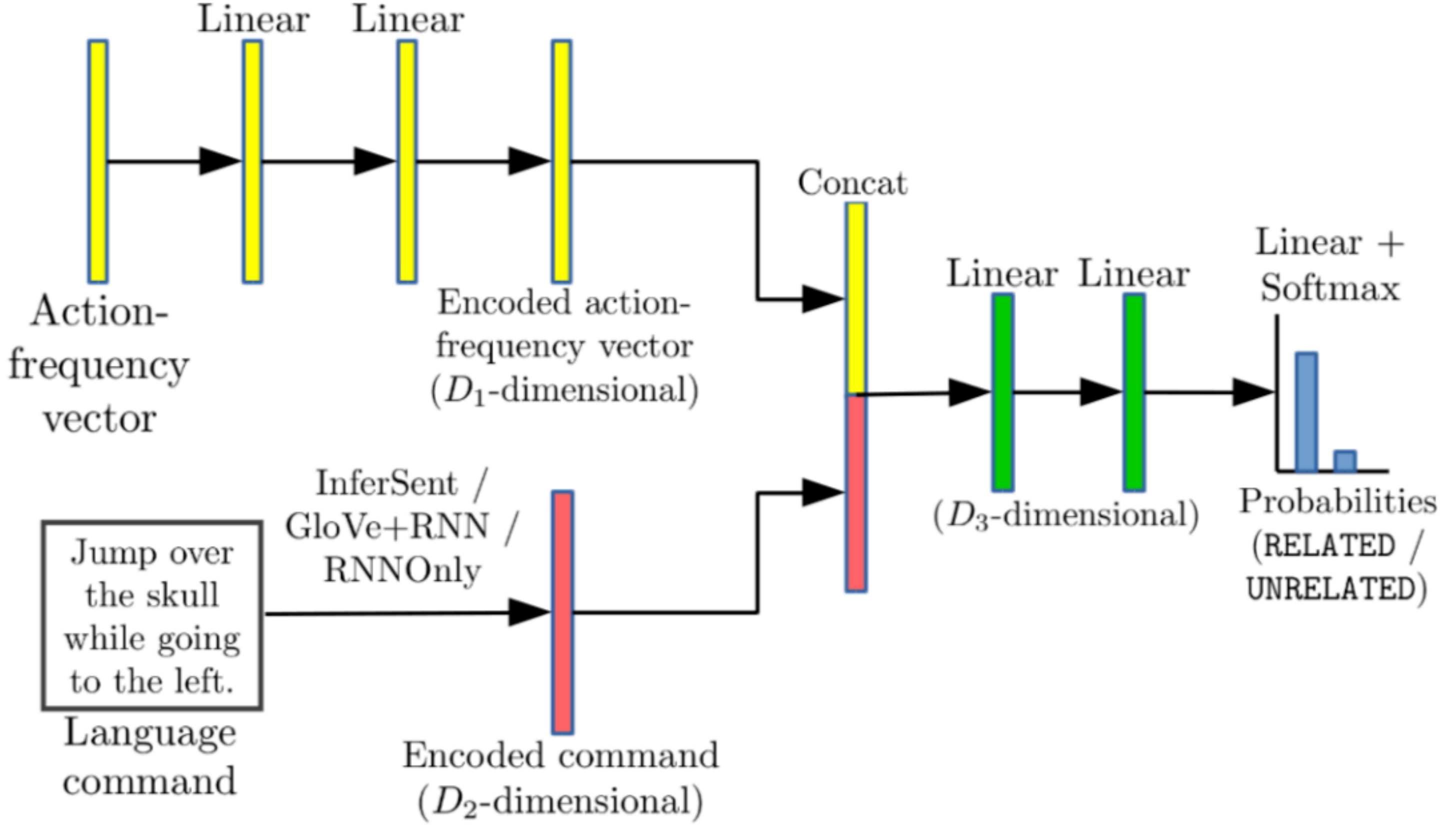

Linear
Linear
Action-frequency vector
Encoded action-frequency vector (D_1-dimensional)
Concat
Linear
Linear
Linear + Softmax
(D_3-dimensional)
Probabilities (RELATED / UNRELATED)
InferSent / GloVe+RNN / RNNOnly
Jump over the skull while going to the left.
Language command
Encoded command (D_2-dimensional)

Figure 3: Neural network architecture for LEARN (Section 3.1)

3. LanguageE-Action Reward Network

3.2 Data Collection

- AMT(Amazon Machanical Turk)를 이용한 Game Frame Data 수집
- Filter out bad Annotation
  - Good Game, Well played (generic statement)
  - 비슷한거 제거
- 6,870개 Language Description Collection (Noisy 포함)
  - 오타, 문법 오류 등

4. Using Language-based Reward in RL

- output probabilities corresponding to classes RELATED and UNRELATED be denoted as $p_R(f_t)$ and $p_u(f_t)$.
- Note that since l is fixed for a given MDP+L, $p_R(f_t)$ and $p_u(f_t)$ are functions of only the current action-frequency vector f_t
- Instruction에 존재하는 Action(jump)이 관련성이 높으면 1, 아니면 0

- **Potential function:**
  - $\boldsymbol{\phi(f_t) = p_R(f_t) - p_u(f_t)}$
- **Shaping Reward(Intermediate Reward):**
  - $\boldsymbol{R_{lang} = \gamma * \phi(f_t) - \phi(f_{t-1})}$
- **Original Reward** : $\boldsymbol{R_{ext}}$
- **Final new Reward:**
  - $\boldsymbol{R_{total} = (R_{lang} + \lambda * R_{ext})}$

5. Experimental Evaluation

5.1 How much does Language help

- **Task : Montezuma's Revenge**
  - **Total Room: 24**
    - **Training: 14 room, 160,000**
    - **Test, Validation : 10 room, 40,000**
  - **Reward:**
    - **Goal Reaching: +1**
    - **All other case : 0**

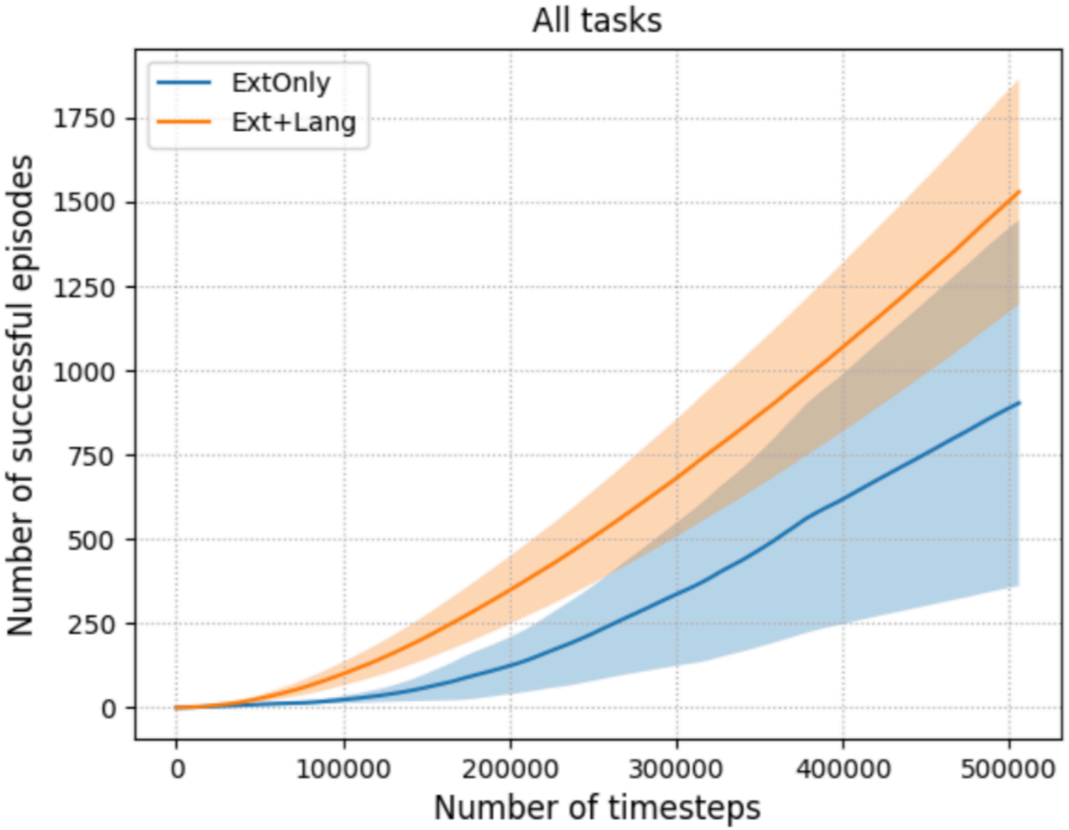


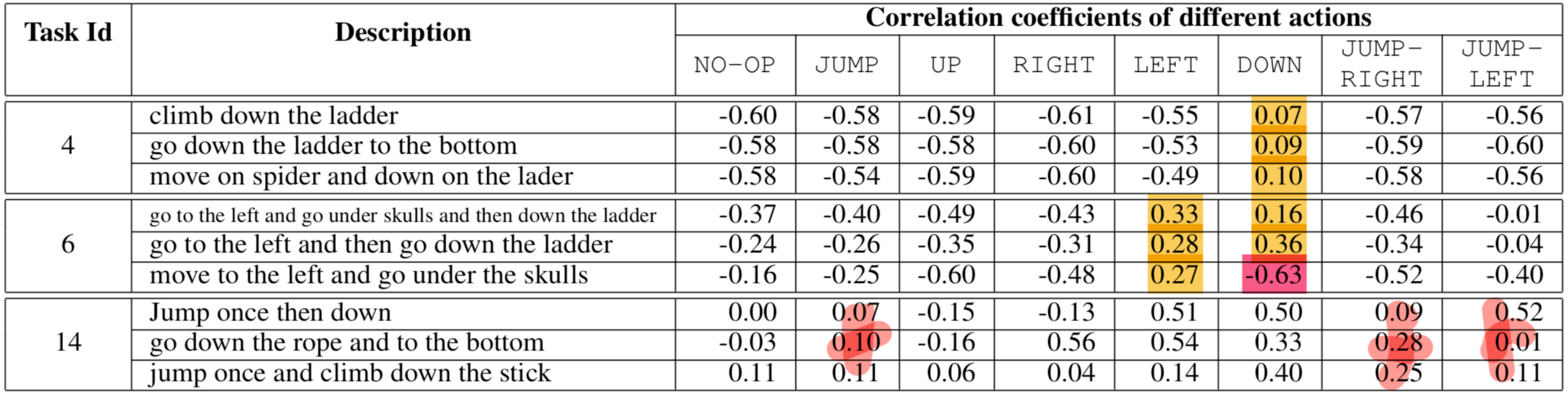
Figure 4: Comparison of different reward functions: The solid lines represent the mean successful episodes averaged over all tasks, and the shaded regions represent 95% confidence intervals.

5. Experimental Evaluation

5.1 How much does Language help

- **Performance is evaluated using two metrics:**
  - **AUC:**
    - From each policy training run, we plot a graph with the number of timesteps on the x-axis and the number of successful episodes on the y-axis. The area under this curve is a measure of **how quickly the agent learns**, and is the metric we use to compare two policy training runs.
  - **Final Policy:**
    - To compare the final learned policy with ExtOnly and Ext+Lang, we perform policy evaluation at the end of 500,000 training steps. For each policy training run, we use the learned policy for an **additional 10,000 timesteps without updating it(Policy training)**, and record the number of successful episodes.
- **Result**
  - average number of successful episodes for ExtOnly after 500,000 timesteps is 903.12, while Ext+Lang achieves that score only after 358,464 timesteps
    - 30% speed up,
  - 500,000 timesteps Ext+Lang completes 1529.43 episodes on average
    - 60% improvement
  - Natural Language Instruction 효과가 좋다고 증명(좋은 reward shaping 방법)

5. Experimental Evaluation



Task Id	Description	Correlation coefficients of different actions							
		NO-OP	JUMP	UP	RIGHT	LEFT	DOWN	JUMP-RIGHT	JUMP-LEFT
4	climb down the ladder	-0.60	-0.58	-0.59	-0.61	-0.55	0.07	-0.57	-0.56
	go down the ladder to the bottom	-0.58	-0.58	-0.58	-0.60	-0.53	0.09	-0.59	-0.60
	move on spider and down on the lader	-0.58	-0.54	-0.59	-0.60	-0.49	0.10	-0.58	-0.56
6	go to the left and go under skulls and then down the ladder	-0.37	-0.40	-0.49	-0.43	0.33	0.16	-0.46	-0.01
	go to the left and then go down the ladder	-0.24	-0.26	-0.35	-0.31	0.28	0.36	-0.34	-0.04
	move to the left and go under the skulls	-0.16	-0.25	-0.60	-0.48	0.27	-0.63	-0.52	-0.40
14	Jump once then down	0.00	0.07	-0.15	-0.13	0.51	0.50	0.09	0.52
	go down the rope and to the bottom	-0.03	0.10	-0.16	0.56	0.54	0.33	0.28	0.01
	jump once and climb down the stick	0.11	0.11	0.06	0.04	0.14	0.40	0.25	0.11

Table 1: Analysis of language-based rewards

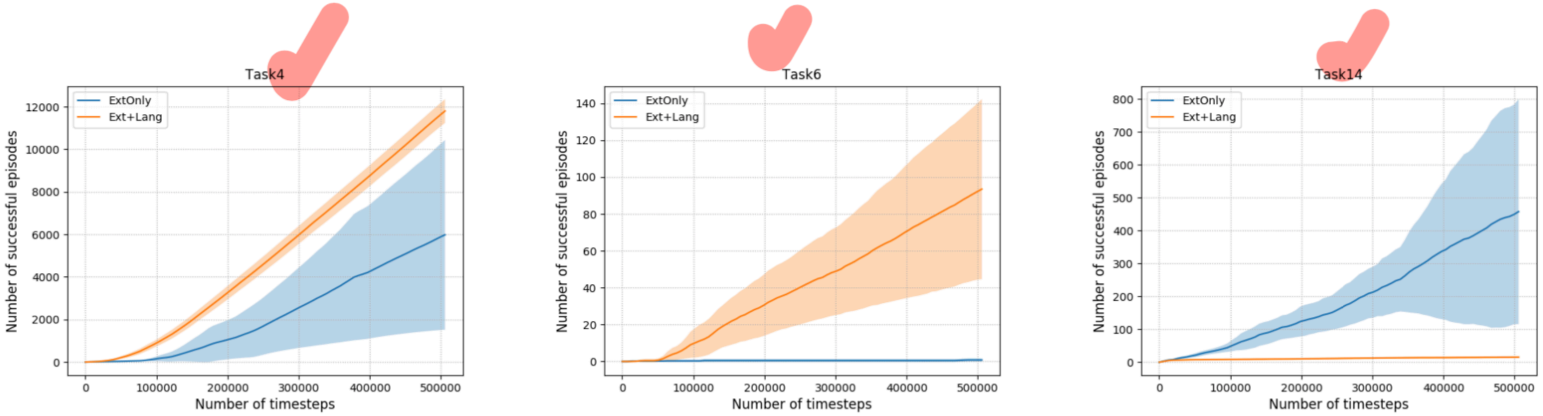


Figure 5: Comparisons of different reward functions for selected tasks

6. Conclusion and Future Work

- **본 논문에서 제안한** LanguagE Action Reward Network (LEARN)이 Sparse한 Reward환경에서 Agent가 빠르게 학습 할 수 있도록 Reward shaping한다는 것을 보임
- Possible Extensions of the approach
  - Temporal ordering:
    - losing temporal information 문제, 따라서 complete action sequences를 사용해서 실험 할 수 있다
  - State-based Reward:
    - State 정보 추가
  - Multi-step instruction
    - 전체 명령어 조합이 완성 됬는지 체크하는 모델을 두고, 최종 명령이 완성 되었을 때만action을 취하는 것
      - Jump the skull + down the ladder (check the complete total instruction)
- My Opinion
  - 강화학습의 최종 버젼은 Natural Language Instruction을 통한 학습이지 않을까? 생각이 된다

Q&A

Thank you