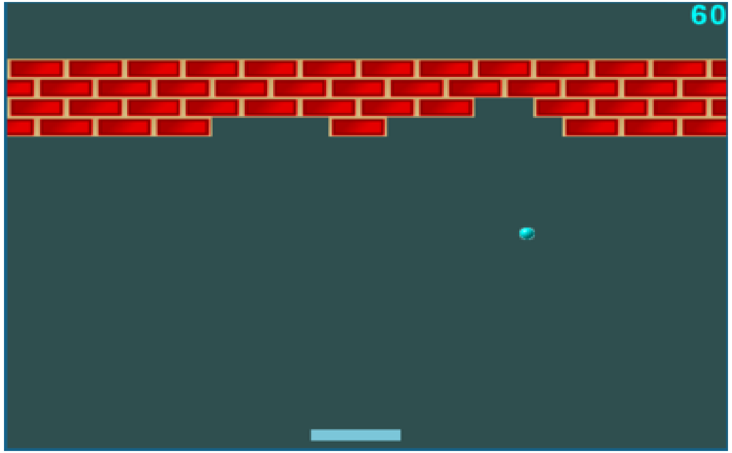
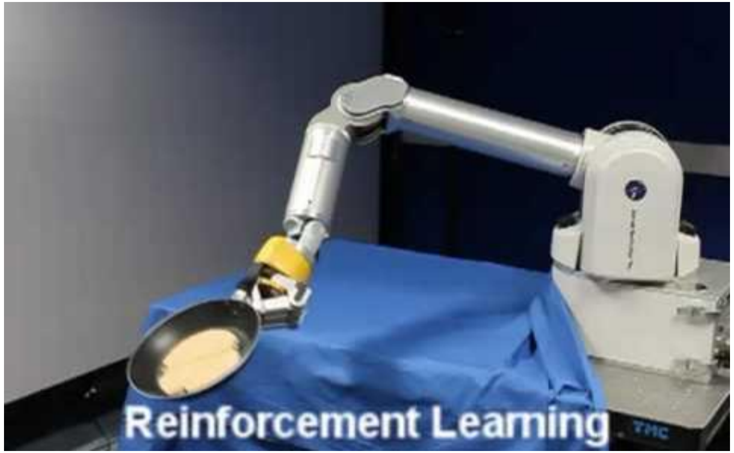
Using Natural Language for Reward Shaping in Reinforcement Learning

Prasoon Goyal, Scott Niekum
and Raymond J. Mooney

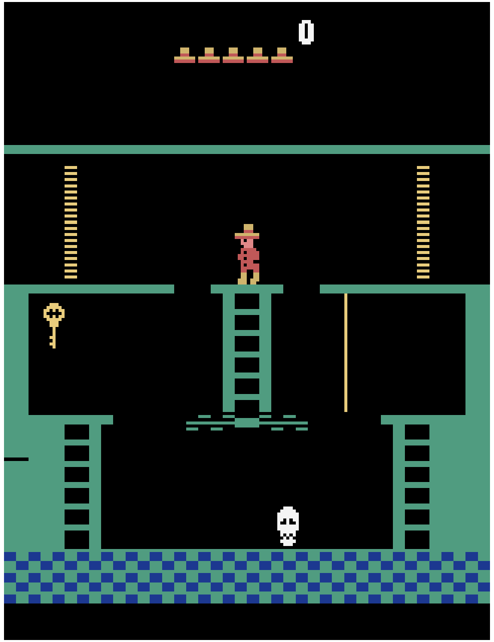
The University of Texas at Austin

Motivation



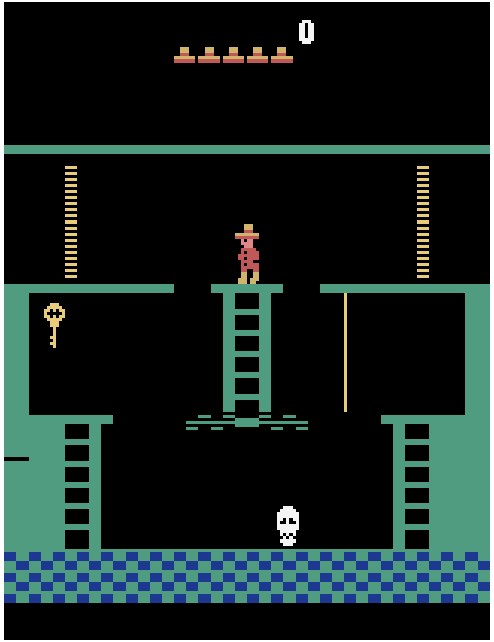


Motivation



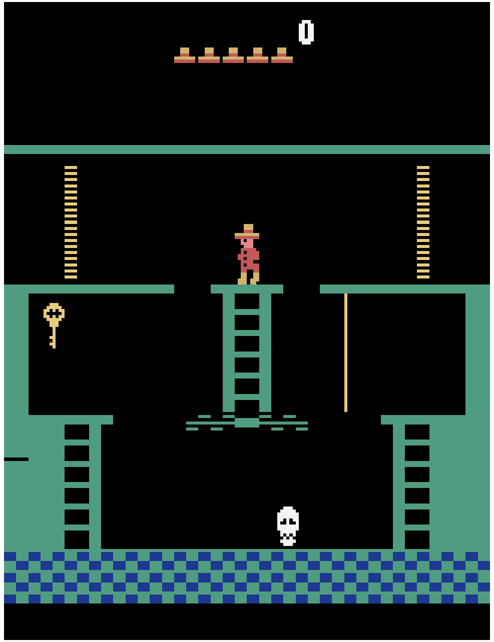
- In sparse reward settings, random exploration has very high sample complexity.

Motivation



- In sparse reward settings, random exploration has very high sample complexity.
- Reward shaping: Intermediate rewards to guide the agent towards the goal.

Motivation



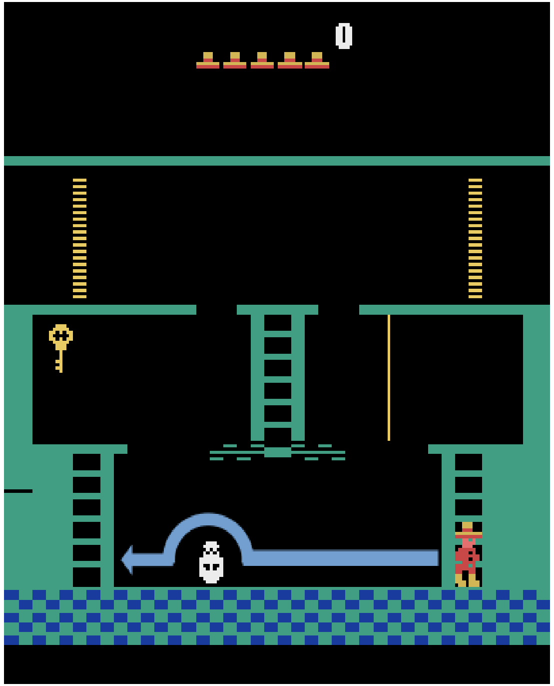
- In sparse reward settings, random exploration has very high sample complexity.
- Reward shaping: Intermediate rewards to guide the agent towards the goal.
- Designing intermediate rewards by hand is challenging.

Motivation

Can we use natural language to provide intermediate rewards to the agent?

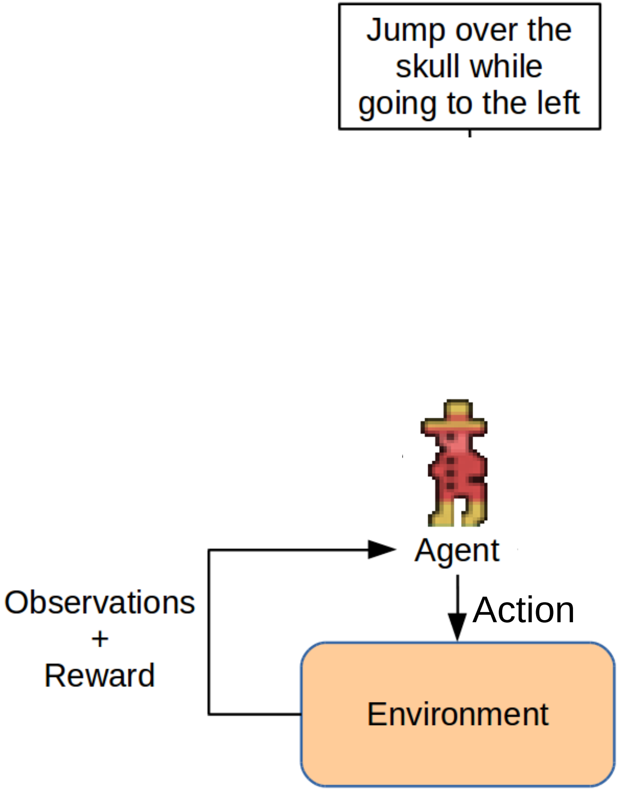
Motivation

Can we use natural language to provide intermediate rewards to the agent?



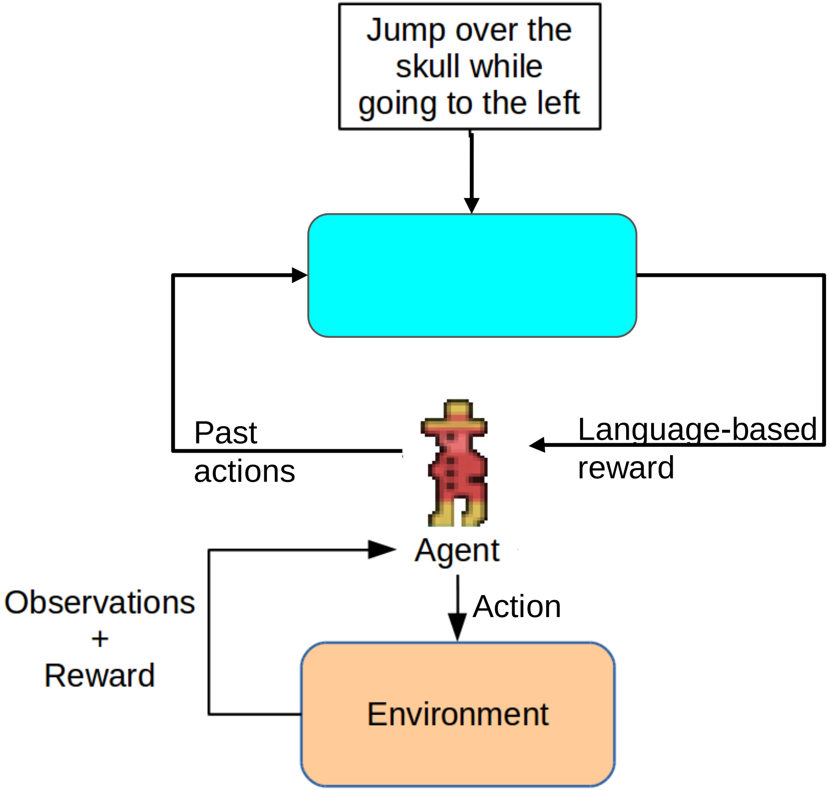
Jump over the skull while going to the left

Problem Statement



- Standard MDP formalism, plus a natural language command describing the task.

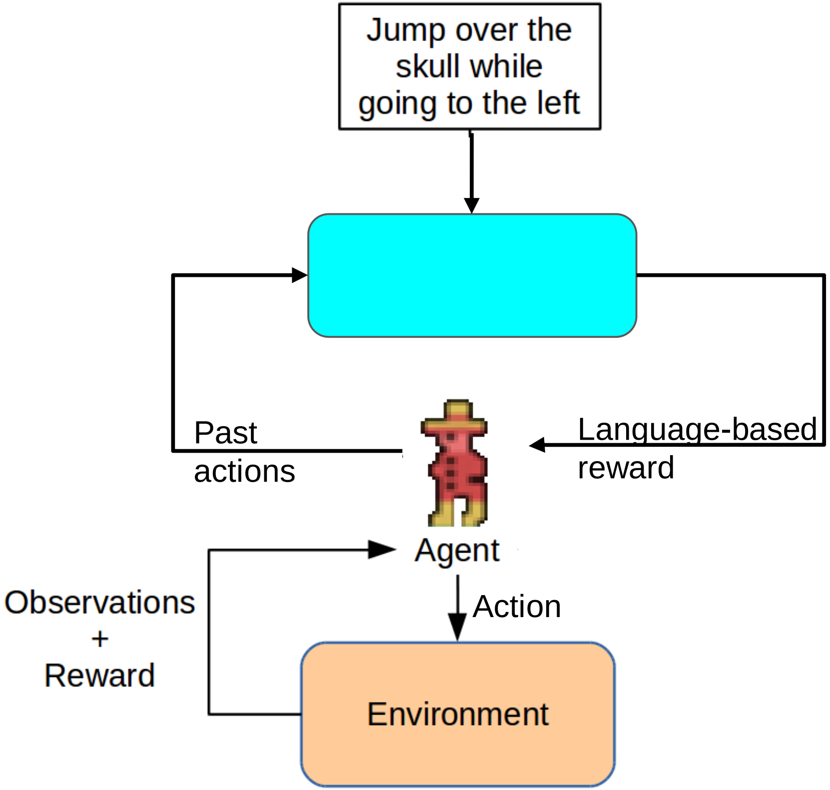
Approach Overview



- Standard MDP formalism, plus a natural language command describing the task.

- Use agent's past actions and the command to generate rewards.

Approach Overview



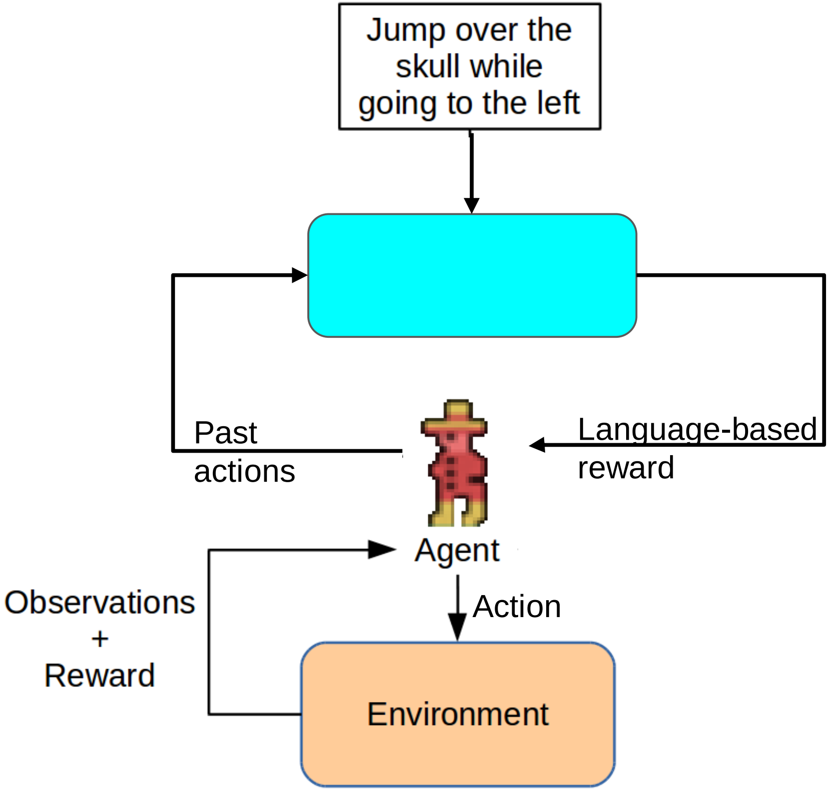
- Standard MDP formalism, plus a natural language command describing the task.
- Use agent's past actions and the command to generate rewards.

For example,

Past actions		**Reward**
LLLJLLL	→	High
RRRUULL	→	Low

[L: Left, R: Right, U: Up, J: Jump]

Approach Overview



- Standard MDP formalism, plus a natural language command describing the task.
- Use agent's past actions and the command to generate rewards.

For example,

Past actions		Reward
4441444	→	High
3332244	→	Low

[4: Left, 3: Right, 2: Up, 1: Jump]

LanguagE-Action Reward Network (LEARN)

Problem: Given a sequence of actions (e.g. 4441444) and a command (e.g. “Jump over the skull while going to the left”), are they related?

LanguagE-Action Reward Network (LEARN)

Problem: Given a sequence of actions (e.g. 4441444) and a command (e.g. "Jump over the skull while going to the left"), are they related?

- Using the sequence of actions, generate an *action-frequency vector*:

ϵ	⇒	[0	0	0	0	0	0	0	0]
4	⇒	[0	0	0	0	1	0	0	0]
42	⇒	[0	0	0.5	0	0.5	0	0	0]
422	⇒	[0	0	0.7	0	0.3	0	0	0]

LanguagE-Action Reward Network (LEARN)

Problem: Given a sequence of actions (e.g. 4441444) and a command (e.g. “Jump over the skull while going to the left”), are they related?

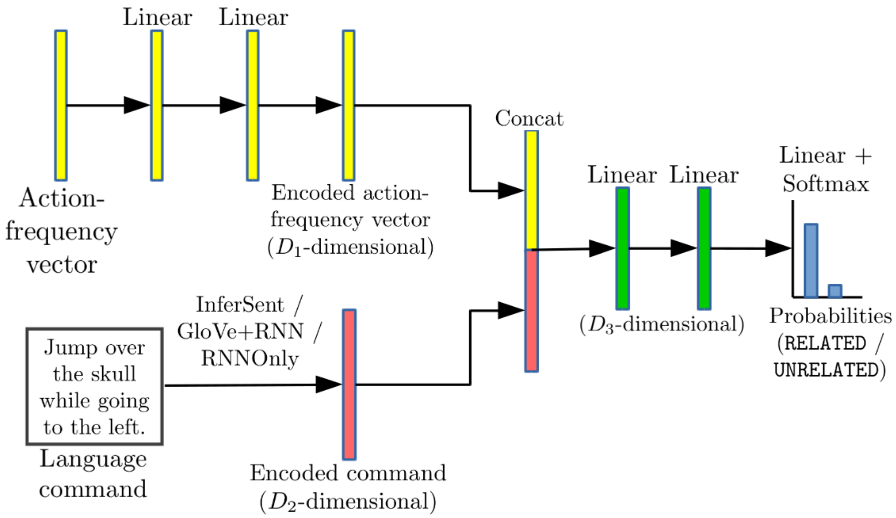
- Using the sequence of actions, generate an *action-frequency vector*:

ϵ	$\Rightarrow$	[0	0	0	0	0	0	0	0]
4	$\Rightarrow$	[0	0	0	0	1	0	0	0]
42	$\Rightarrow$	[0	0	0.5	0	0.5	0	0	0]
422	$\Rightarrow$	[0	0	0.7	0	0.3	0	0	0]

- Train a neural network that takes in the action-frequency vector and the command to predict whether they are related or not.

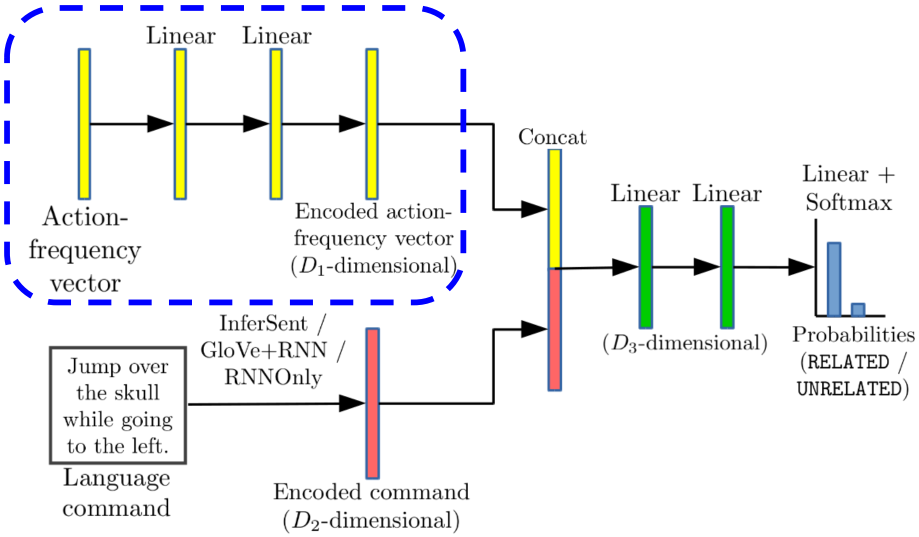
LanguagE-Action Reward Network (LEARN)

Neural Network Architecture



LanguagE-Action Reward Network (LEARN)

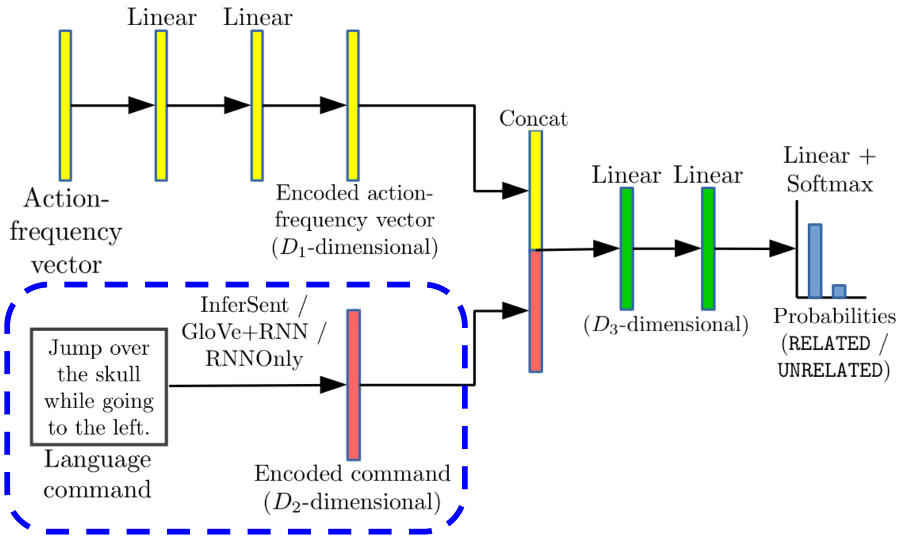
Neural Network Architecture



- Action-frequency vector passed through 3 linear layers.

LanguagE-Action Reward Network (LEARN)

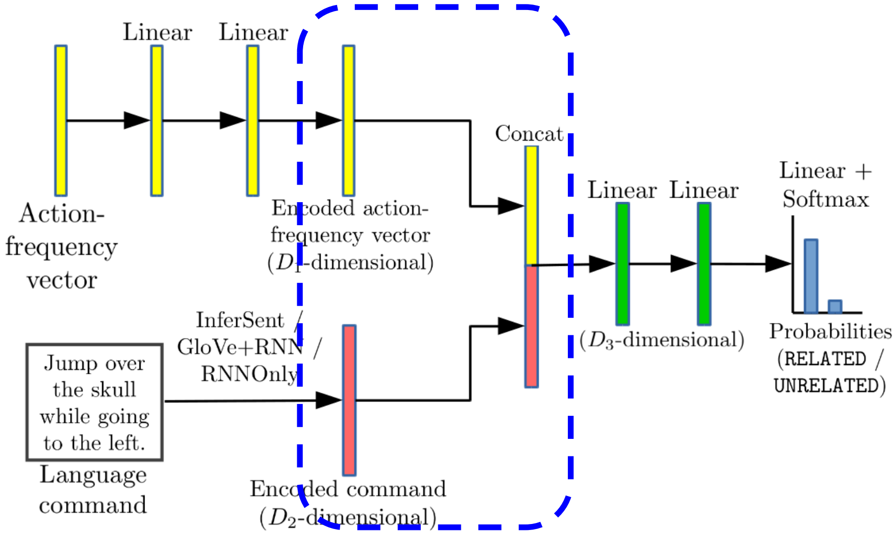
Neural Network Architecture



- Action-frequency vector passed through 3 linear layers.
- Three language encoders:
  - InferSent
  - GloVe+RNN
  - RNNOnly

LanguagE-Action Reward Network (LEARN)

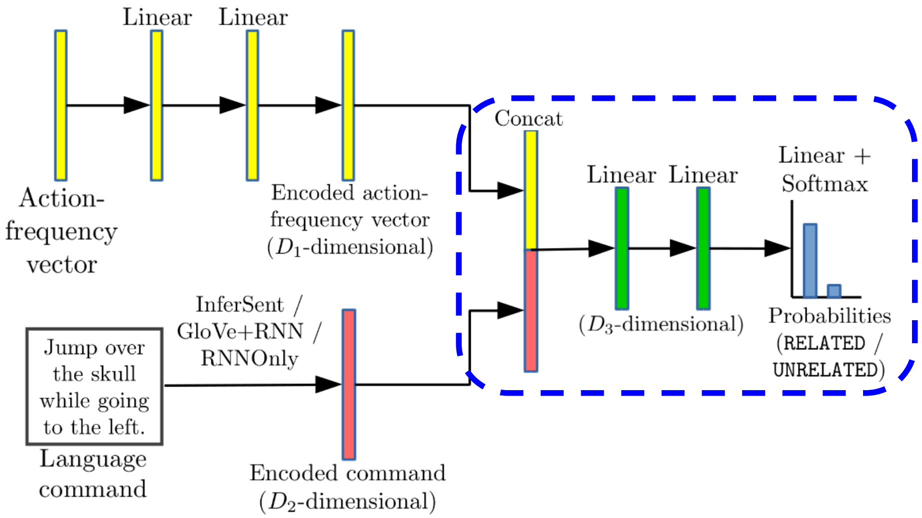
Neural Network Architecture



- Action-frequency vector passed through 3 linear layers.
- Three language encoders:
  - InferSent
  - GloVe+RNN
  - RNNOnly
- Concatenate encoded action-frequency vector and encoded language.

LanguagE-Action Reward Network (LEARN)

Neural Network Architecture

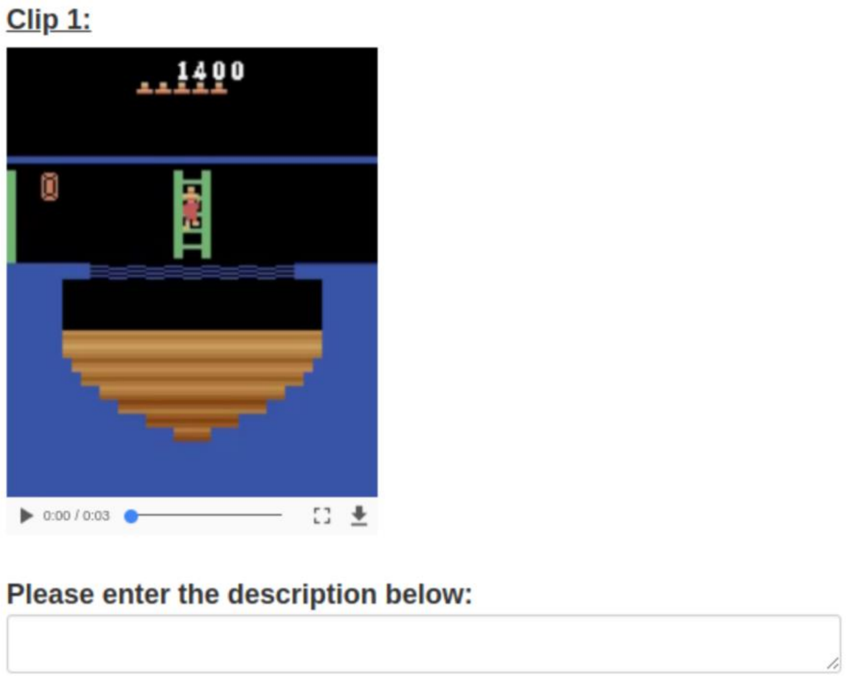


- Action-frequency vector passed through 3 linear layers.
- Three language encoders:
  - InferSent
  - GloVe+RNN
  - RNNOnly
- Concatenate encoded action-frequency vector and encoded language.
- Pass through linear layers followed by softmax layer.

LanguagE-Action Reward Network (LEARN)

Data Collection

- Used Amazon Mechanical Turk to collect language descriptions for trajectories.



LanguagE-Action Reward Network (LEARN)

Data Collection

- Used Amazon Mechanical Turk to collect language descriptions for trajectories.

- Minimal postprocessing to remove low quality data.

Ill-formed

Spelling errors

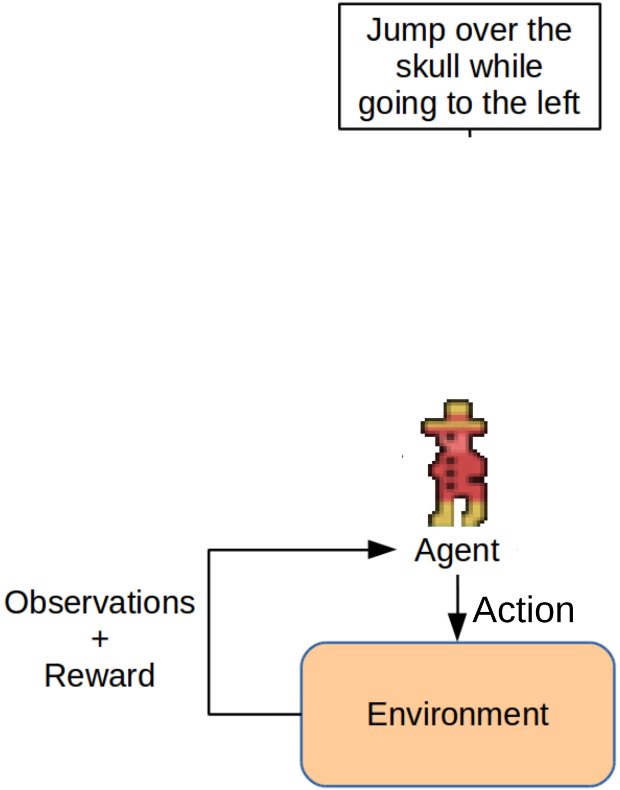
1.	wait
2.	using the ladder on standing
3.	going slow and climb down the ladder
4.	move down the ladder and walk left
5.	go left watch the trap and move on
6.	climbling down the ladder
7.	ladder dwon and running this away
8.	stay in place on the ladder.
9.	go down the ladder
10.	go right and climb up the ladder
11.	just jump and little move to right side
12.	run all the way to the left.
13.	go left jumping once
14.	go left
15.	move right and jump over green creature then go down the ladder
16.	hop over to the middle ledge
17.	wait for the two skulls and dodge them in the middle
18.	walk to the left and then jump down
19.	jump to collected gold coin and little move
20.	wait for the platform to materialize then walk and leap to your right to collect the coins.

LanguagE-Action Reward Network (LEARN)

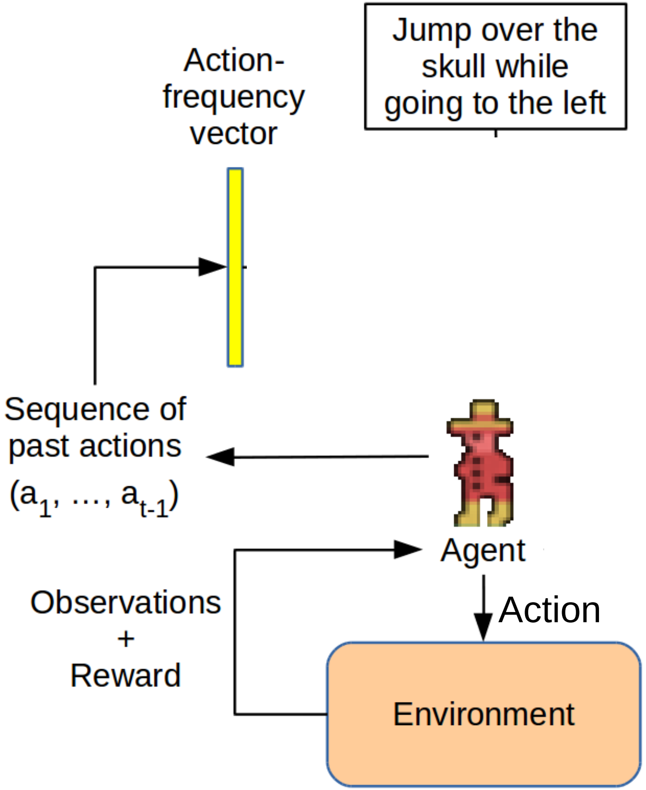
Data Collection

- Used Amazon Mechanical Turk to collect language descriptions for trajectories.
- Minimal postprocessing to remove low quality data.
- Used random pairs to generate negative examples.

Putting it all together...

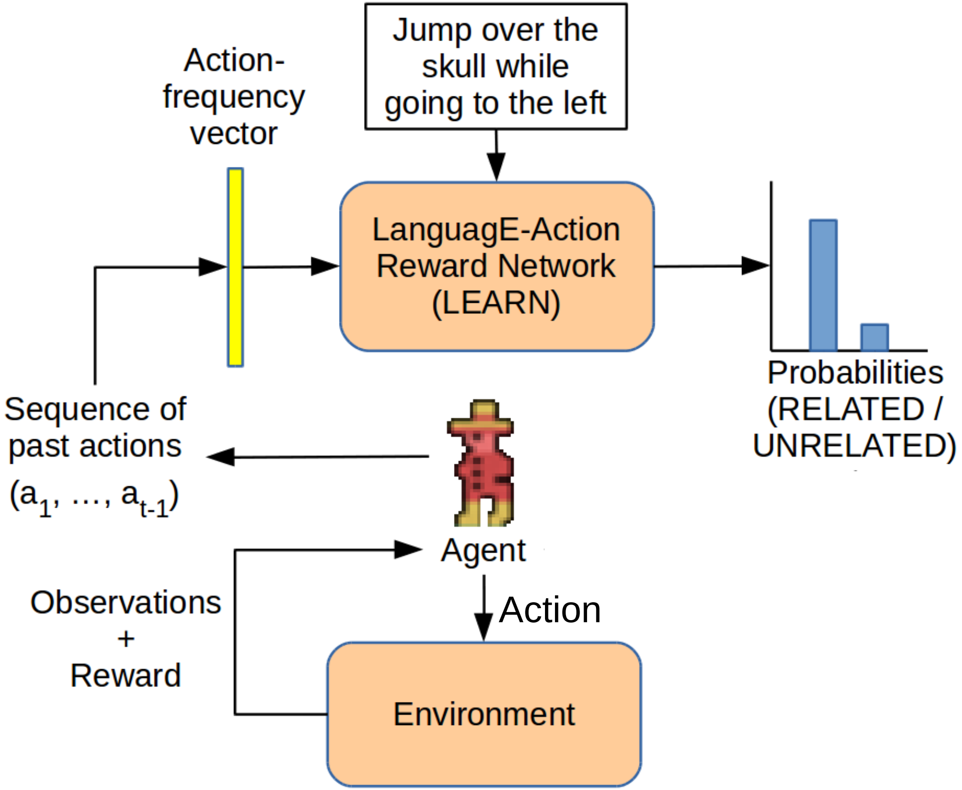


Putting it all together...



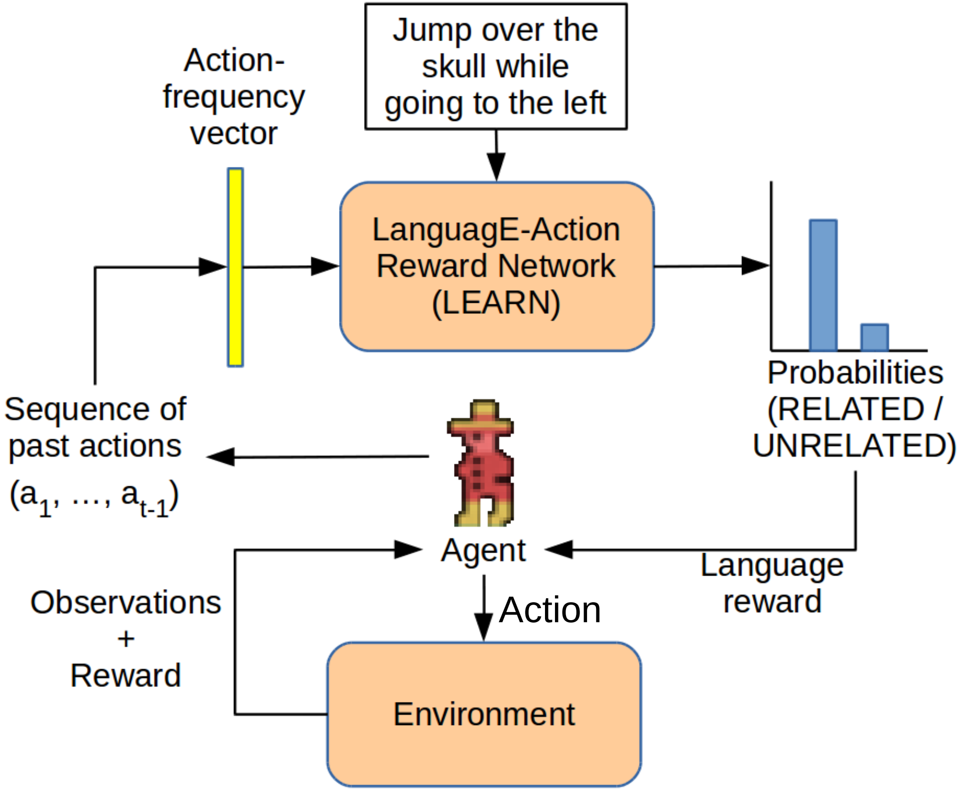
- Using the agent's past actions, generate an *action-frequency vector*.

Putting it all together...



- Using the agent's past actions, generate an *action-frequency vector*.
- LEARN: scores the relatedness between the action-frequency vector and the language command.

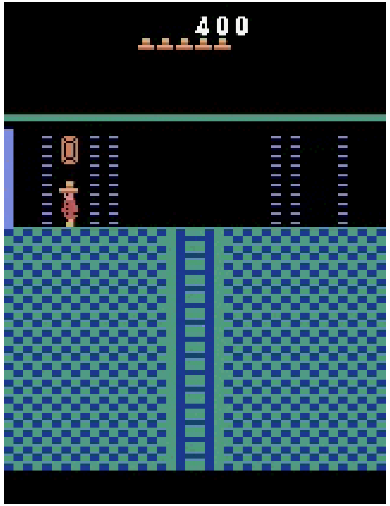
Putting it all together...

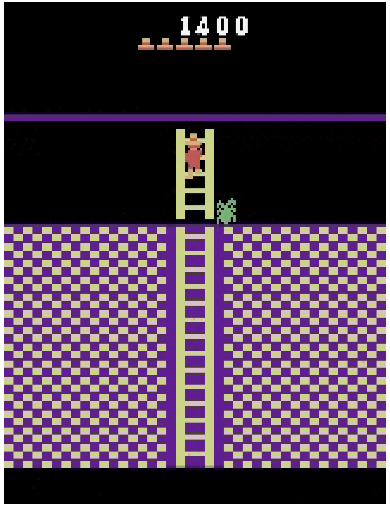


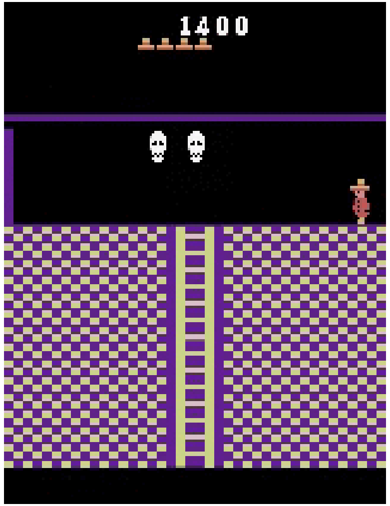
- Using the agent's past actions, generate an *action-frequency vector*.
- LEARN: scores the relatedness between the action-frequency vector and the language command.
- Use the relatedness scores as intermediate rewards, such that the optimal policy does not change.

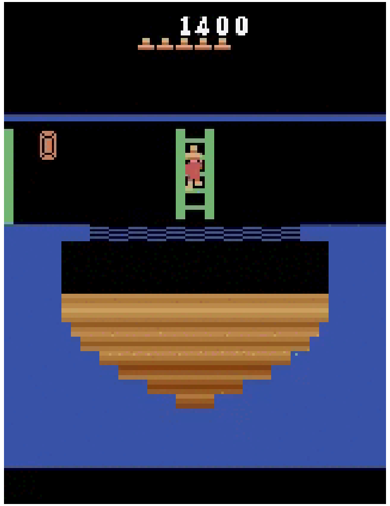
Experiments

- 15 tasks



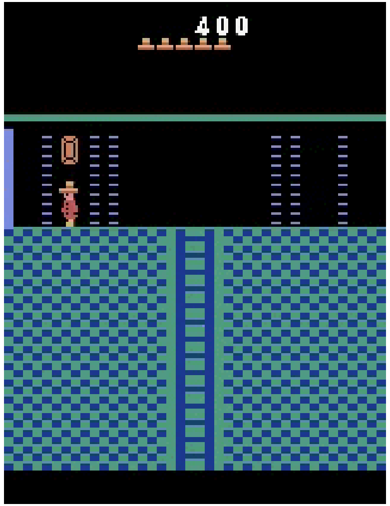






Experiments

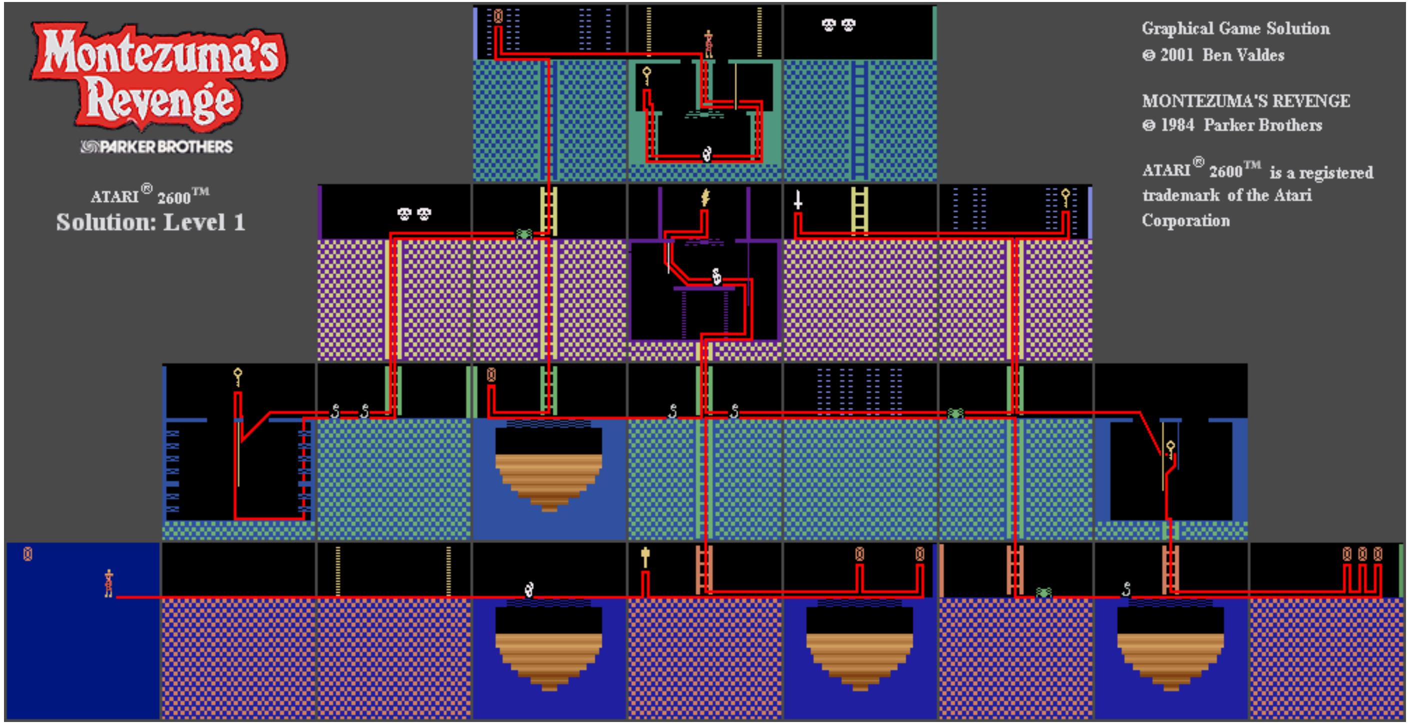
- Amazon Mechanical Turk to collect 3 descriptions for each task.



- JUMP TO TAKE BONUS WALK RIGHT AND LEFT THE CLIMB DOWNWARDS IN LADDER
- Jump Pick Up The Coin And Down To Step The Ladder
- jump up to get the item and go to the right

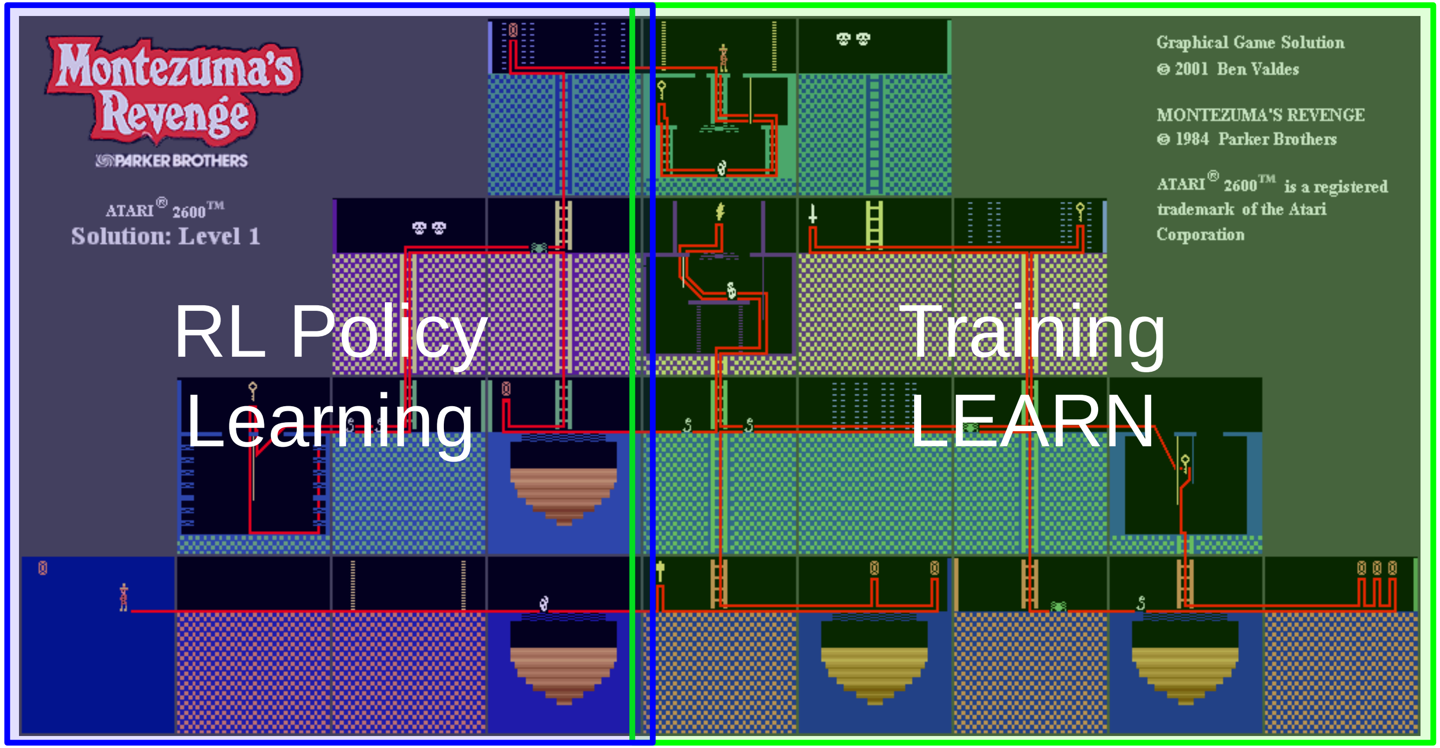
Experiments

- Different rooms used for training LEARN and RL policy learning.



Experiments

- Different rooms used for training LEARN and RL policy learning.

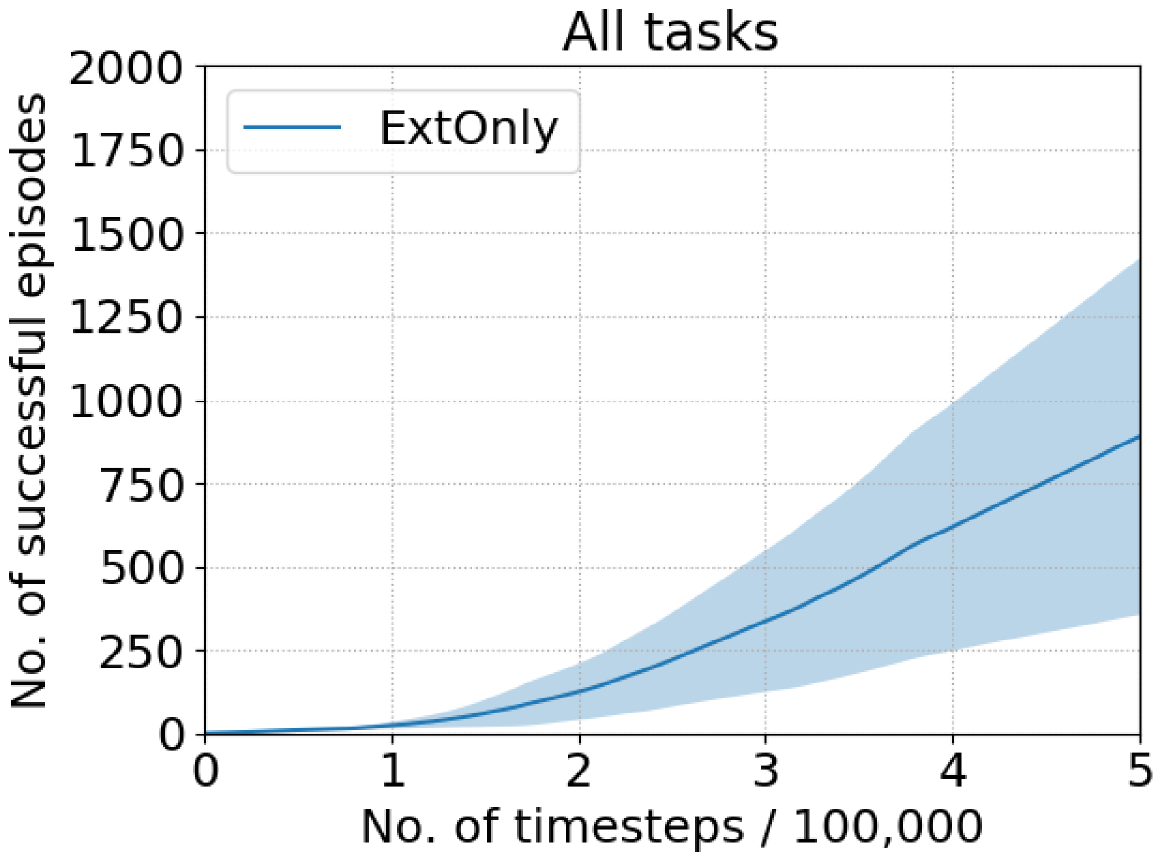


Results

- Compared RL training using PPO algorithm with and without language-based reward.

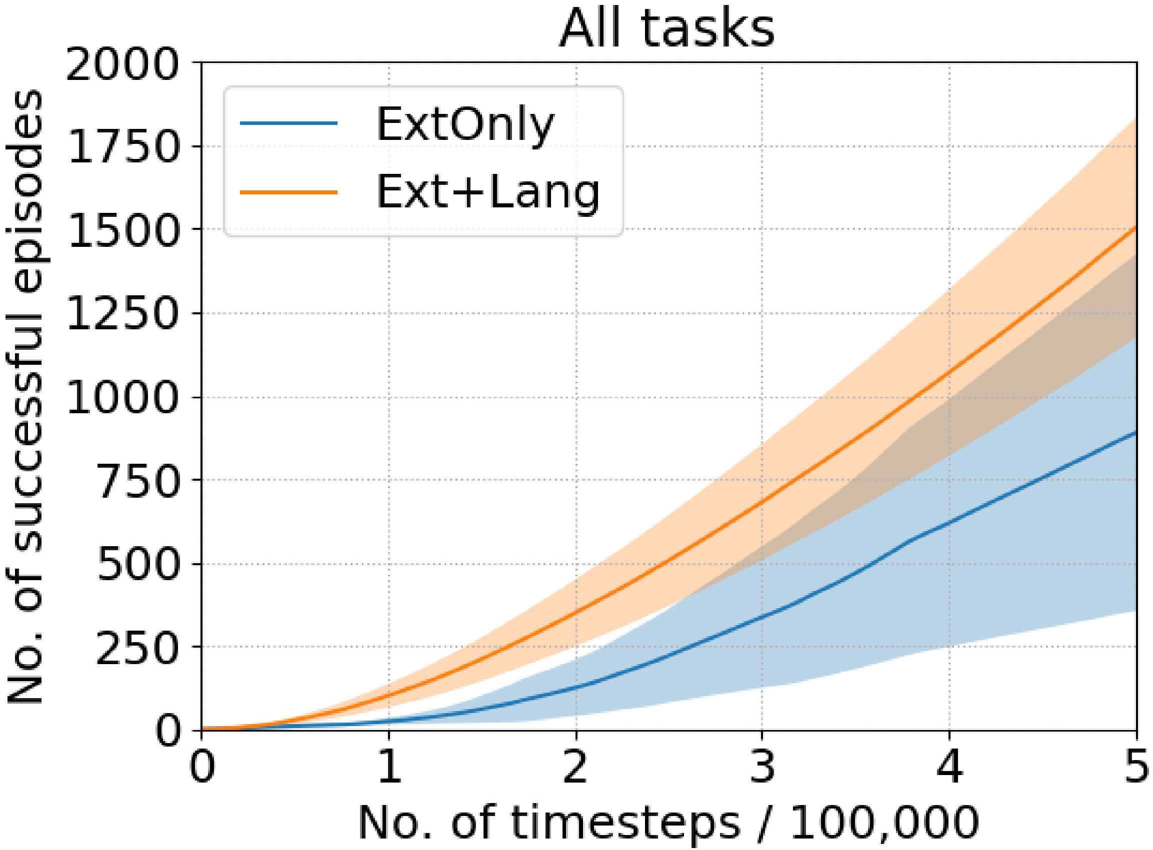
Results

- Compared RL training using PPO algorithm with and without language-based reward.
- ExtOnly: Reward of 1 for reaching the goal, reward of 0 in all other cases.



Results

- Compared RL training using PPO algorithm with and without language-based reward.
- ExtOnly: Reward of 1 for reaching the goal, reward of 0 in all other cases.
- Ext+Lang: Extrinsic reward plus language-based intermediate rewards.



Analysis

Analysis

- For a given RL run, we have a fixed natural language description.

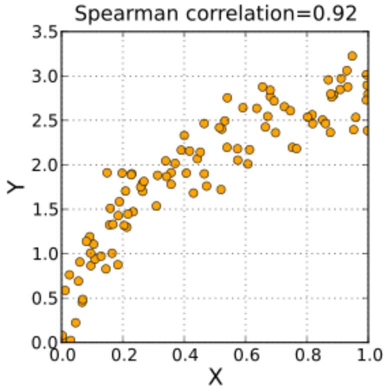
Analysis

[0	0	0	0	1	0	0	0]	0.2
[0	0	0.5	0	0.5	0	0	0]	0.1
.								
.								
.								
[0.1	0.1	0.1	0.3	0.1	0.1	0.1	0.1]	0.3
.								
.								
.								

- For a given RL run, we have a fixed natural language description.
- At every timestep, we get an action-frequency vector, and the corresponding prediction from LEARN.

Analysis

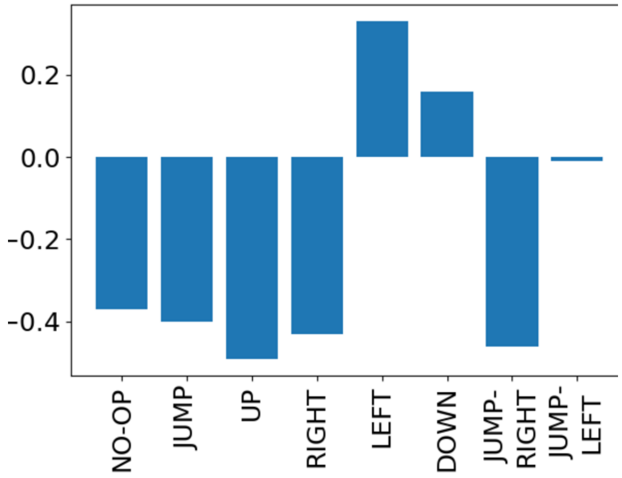
[0	0	0	0	1	0	0	0]	0.2
[0	0	0.5	0	0.5	0	0	0]	0.1
.								
.								
.								
[0.1	0.1	0.1	0.3	0.1	0.1	0.1	0.1]	0.3
.								
.								
.								



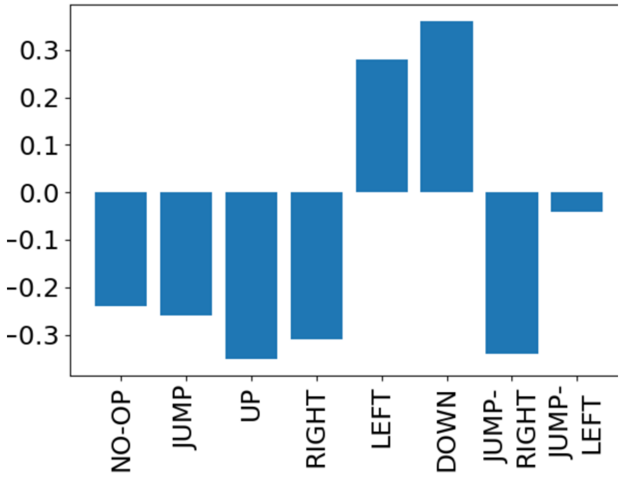
- For a given RL run, we have a fixed natural language description.
- At every timestep, we get an action-frequency vector, and the corresponding prediction from LEARN.
- Compute Spearman correlation coefficient between each component (action) and the prediction.

Analysis

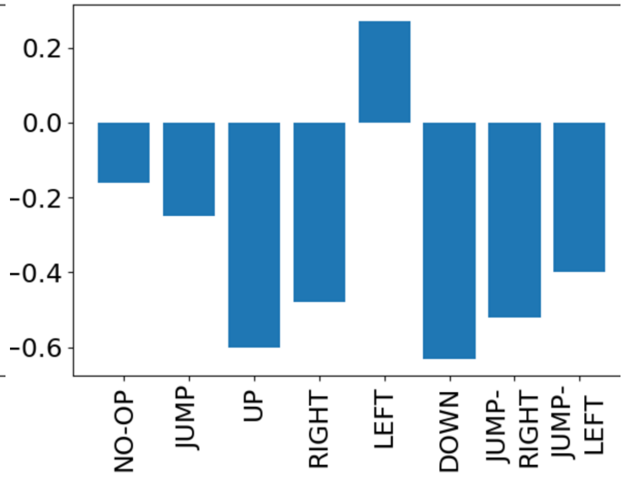
go to the left and go under skulls and then down the ladder



go to the left and then go down the ladder

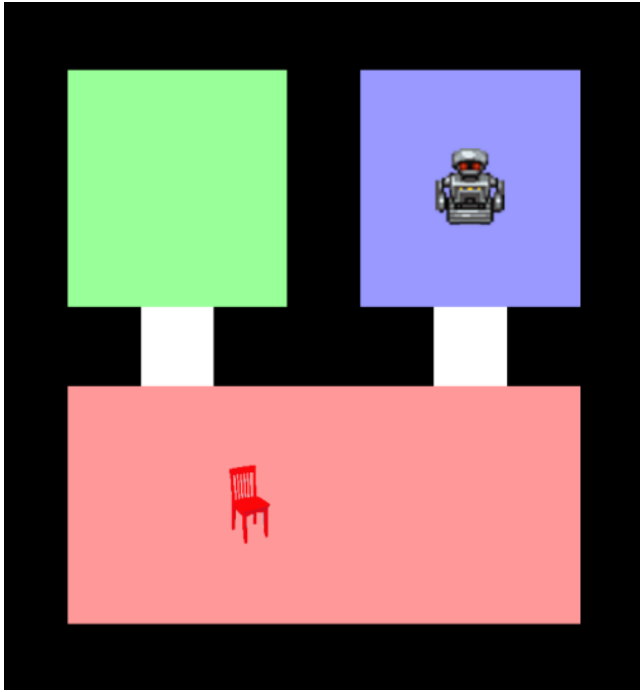


move to the left and go under the skulls



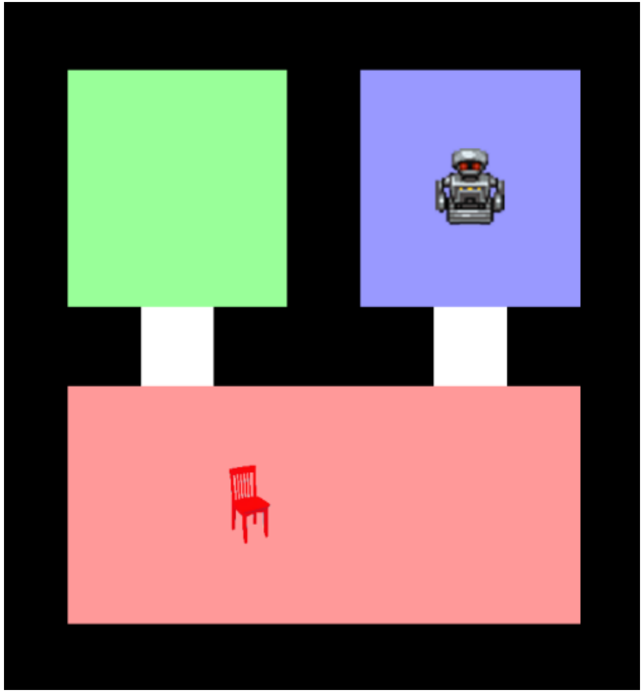
Related Work

Related Work

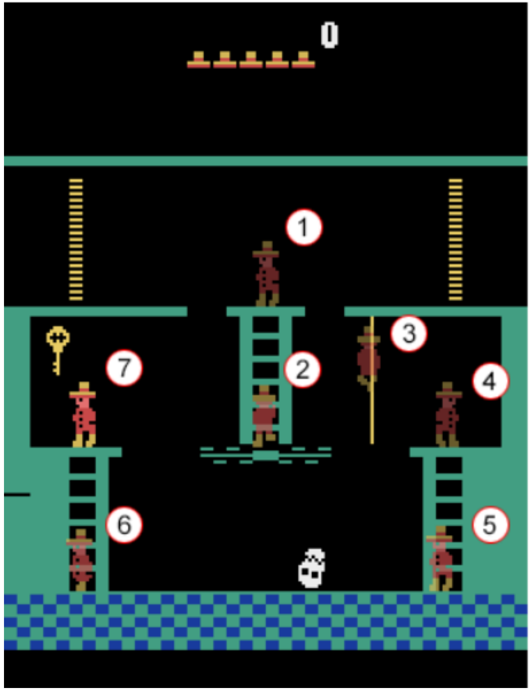


Language to Reward
[Williams et al. 2017,
Arumugam et al. 2017]

Related Work

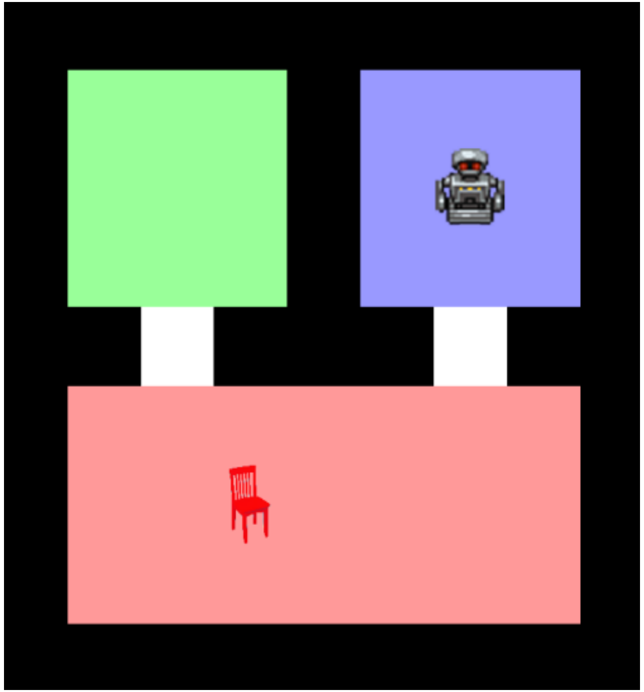


Language to Reward
[Williams et al. 2017,
Arumugam et al. 2017]

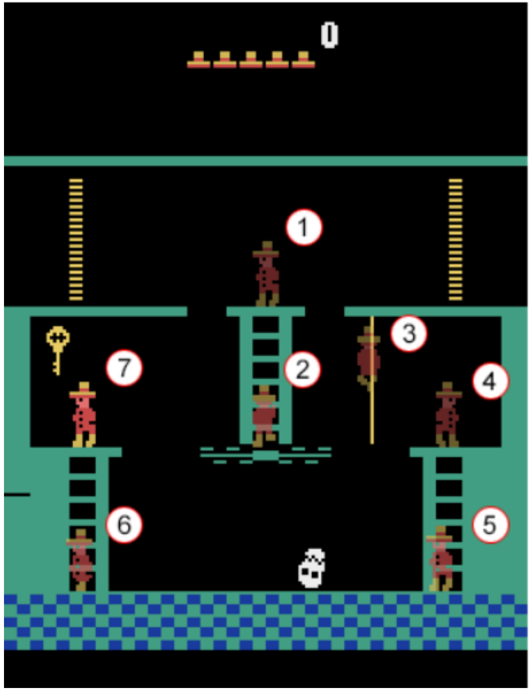


Language to Subgoals
[Kaplan et al. 2017]

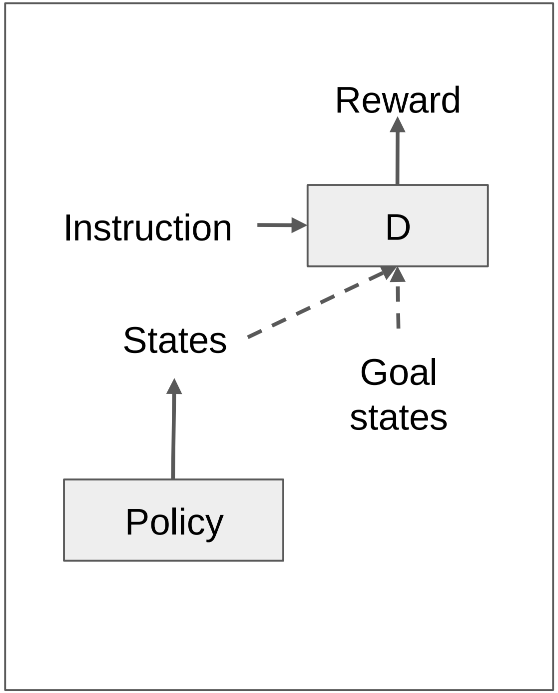
Related Work



Language to Reward
[Williams et al. 2017,
Arumugam et al. 2017]



Language to Subgoals
[Kaplan et al. 2017]



Adversarial Reward Induction
[Bahdanau et al. 2018]

Summary

- Proposed a framework to incorporate natural language to aid RL exploration.

Summary

- Proposed a framework to incorporate natural language to aid RL exploration.

- Two-phase approach:
  1. Supervised training of the LEARN module.
  2. Policy learning using *any* RL algorithm with language-based rewards from LEARN.

Summary

- Proposed a framework to incorporate natural language to aid RL exploration.

- Two-phase approach:
  1. Supervised training of the LEARN module.
  2. Policy learning using *any* RL algorithm with language-based rewards from LEARN.

- Analysis shows that the framework discovers mapping between language and actions.

Summary

- Proposed a framework to incorporate natural language to aid RL exploration.

- Two-phase approach:
  1. Supervised training of the LEARN module.
  2. Policy learning using *any* RL algorithm with language-based rewards from LEARN.

- Analysis shows that the framework discovers mapping between language and actions.

- Extensions:
  - Temporal information
  - Continuous action space
  - State information
  - Multi-step instructions

Summary

- Proposed a framework to incorporate natural language to aid RL exploration.

- Two-phase approach:
  1. Supervised training of the LEARN module.
  2. Policy learning using *any* RL algorithm with language-based rewards from LEARN.

- Analysis shows that the framework discovers mapping between language and actions.

- Extensions:
  - Temporal information
  - Continuous action space
  - State information
  - Multi-step instructions

Code and Data available at
www.cs.utexas.edu/~pgoyal