Hierarchical Programmatic Reinforcement Learning via Learning to Compose Programs

ICML 2023

Guan-Ting Liu*
En-Pei Hu*
Pu-Jen Cheng
Hung-Yi Lee
Shao-Hua Sun
Robot Learning via Deep Reinforcement Learning

Goal: maximize

\[
\sum_{t=0}^{H} \gamma^t R_t(s_t, a_t)
\]
Robot Learning via Deep Reinforcement Learning - Issues

Generalization

Simple task → Complex task

Interpretability

Trust, Safety, and Contestability

Deep neural network policy
Learning to Synthesize Programs as Interpretable and Generalizable Policies

NeurIPS 2021

Dweep Trivedi*
Jesse Zhang*
Joseph J. Lim
Shao-Hua Sun

```
DEF run()
    IF frontIsClear
        move
    ELSE
        IF markerPresent
            pickMarker
        ELSE
            turnRight

Program
```

Execute

Reward

Environment

Dweep Trivedi*
Jesse Zhang*
Shao-Hua Sun
Joseph J. Lim
Deep Reinforcement Learning

Deep Neural Network

Inference

Environment

Reward
Reinforcement Learning via Synthesizing Programs

Model → Synthesize → Program → Execute → Environment

LEAPS Program Synthesizer

DEF run()
  IF frontIsClear
    move
  ELSE
    IF markerPresent
      pickMarker
    ELSE
      turnRight

Reward
LEAPS: Learning Embeddings for Latent Program Synthesis

Stage 1  Learn a program embedding space from randomly generated programs

Goal  Learn the grammar and the environment dynamics

Program

```python
DEF run()
    IF frontIsClear
        move
    ELSE
        IF markerPresent
            pickMarker
        ELSE
            turnRight
```

Program Embedding Space

Reconstructed Program

```python
DEF run()
    IF NOT frontIsClear
        IF markerPresent
            pickMarker
        ELSE
            turnRight
    ELSE
        move
```

Learn a program embedding space from randomly generated programs

Stage 1  Learn the grammar and the environment dynamics
LEAPS: Learning Embeddings for IAtent Program Synthesis

Stage 2  Search for a task-solving program using the cross-entropy method (CEM)

Goal  Optimize the task performance

Sample Program Embeddings

Return

Iteration 1  0.32
Iteration 2  0.49
Iteration n  0.95

Update Distribution

Decoder

DEF run()
IF frontIsClear
move
turnLeft
REPEAT(2)
turnRight
putMarker

Decoded Program

Task Environment

Reward

Execute
Karel Tasks

StairClimber

Maze

FourCorners

TopOff

Harvester

CleanHouse
Quantitative Results

![Bar chart showing reward for various environments: Maze, StairClimber, TopOff, Harvester, FourCorner, CleanHouse. The chart compares performance across LEAPS, Deep RL, 1-stage Program, and VIPER.]
LEAPS: Learning Embeddings for IAtent Program Synthesis

Stage 2  Searching for a task-solving program using the cross-entropy method

Return

Iteration 1 0.32
Iteration 2 0.49
Iteration n 0.95

Sample Program Embeddings

Update Distribution

Decoder

Decoded Program

DEF run()
  IF frontIsClear
    move
turnLeft
  REPEAT(2)
    turnRight
    putMarker

Task Environment

Reward

Execute
Limited program distribution
Search in the program embedding space spanned by the dataset programs

Cannot synthesize longer or more complex programs

Poor credit assignment
Evaluate each candidate program solely based on the cumulative return of its execution trace

Cannot accurately attribute rewards to corresponding program parts
Hierarchical Programmatic Reinforcement Learning via Learning to Compose Programs

ICML 2023

Guan-Ting Liu*
En-Pei Hu*
Pu-Jen Cheng
Hung-Yi Lee
Shao-Hua Sun

Execute

Composed Program

Reward

Environment
Stage 1  Learning a compressed program embedding space from randomly generated programs
HPRL: Hierarchical Programmatic Reinforcement Learning

Stage 2  Learning a meta policy to produce a series of programs (i.e., predict a series of actions) to yield a composed task-solving program.

Initial State $s_0$:

Meta Policy

Predict Action/Program

Task Environment

Transition

Reward

Optimize Return

$r_1: 0.21$

$r_2: 0.14$

$r_3: 0.37$

Decoded Program

Decompose Program

DEF run() IF frontIsClear move turnLeft REPEAT (2) turnRight putMarker move DEF run() WHEN frontIsClear IF markerPresent pickMarker ELSE turnRight DEF run() IF NOT frontIsClear IF markerPresent pickMarker ELSE turnRight ELSE move

Initial States:

$s_0$: 

$s_1$: $r_1: 0.21$

$s_2$: $r_2: 0.14$

$s_3$: $r_3: 0.37$

Execute
Quantitative Results - Karel Tasks

- Maze
- StairClimber
- Harvester
- TopOff
- FourCorner
- CleanHouse

Graph showing rewards for different tasks and algorithms:
- HPRL
- LEAPS
- Deep RL
- 1-stage Program
- VIPER
Karel-Hard Tasks

OneStroke

Seeder

DoorKey

Snake

<table>
<thead>
<tr>
<th>Reward</th>
<th>HPRL</th>
<th>LEAPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Tasks: OneStroke, Seeder, DoorKey, Snake
Additional Experiments

**Limited program distribution**

Synthesize out-of-distributionally long programs

- HPRL can synthesize programs longer than the dataset programs (< 40 tokens) better than LEAPS

**Poor credit assignment**

Learning from episodic reward

- Dense: Reward each subprogram based on its execution trace
- Episodic: Reward the entire composed program at the end

- The hierarchical design of HPRL allows for better credit assignment with dense rewards, facilitating the learning progress
Thank You

Poster Session #4
Jul 26 Wed 2 p.m. — 3:30 p.m.
@ Exhibit Hall 1