Neural Logic Reinforcement Learning

Zhengyao Jiang, Shan Luo

Department of Computer Science,
University of Liverpool,
Liverpool, United Kingdom

z.jiang22@student.liverpool.ac.uk
shan.luo@liverpool.ac.uk

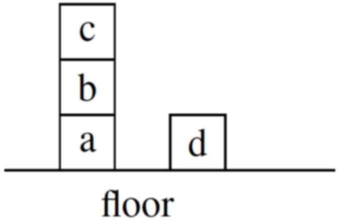
Differentiable Inductive Logic Programming

- It is to learn logic rules to express a concept, for example, grandfather(X,Y)
- It needs background knowledge:
  - father(david, bert), father(bert, tom), father(richard, lily), father(sam, richard)
- The positive and negative examples are provided:
  - Positive: grandfather(david, tom), grandfather(sam, lily)
  - Negative: any other atoms involving grandfather
- We can induce: grandfather(X,Y):- father(X,Z), father(Z,Y)
- The differentiable ILP algorithms parameterise possible rules and use gradient descent to induce the correct rule
  - The algorithm we will use for NLRL is ∂ILP [1]

[1] R. Evans and E. Grefenstette. Learning Explanatory Rules from Noisy Data. *JAIR*, 2018.

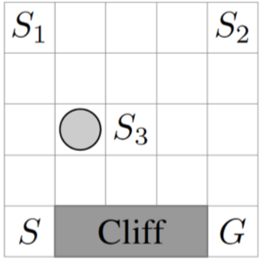
Learn Logic Rules with Policy Gradient - NLRL

- The state and background knowledge is encoded as a knowledge base
- In each decision step, after deduction of DILP, the truth score of the action atoms is used to derive the action probability
- The DILP architecture is trained with REINFORCE [2]



state: $top(d)$, $top(c), on(d, floor)$, $on(c, b), on(b, a)$, $on(a, floor)$

background: $isFloor(floor)$



state: $current(1, 2)$

background: $zero(0)$, $last(4), succ(0, 1)$, $succ(1, 2), succ(2, 3)$, $succ(3, 4)$

[2] R. J. Willia, Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning*, 1992.

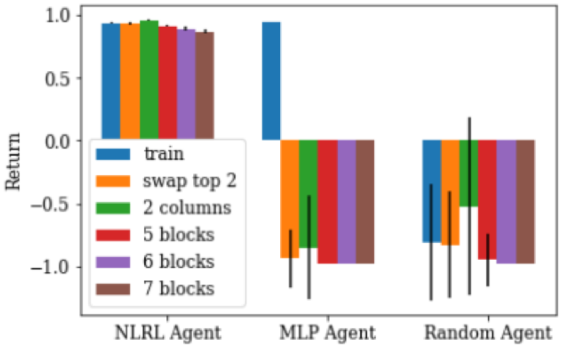
NLRL vs Symbolic Planning

	NLRL	Symbolic Planning
Algorithm Based On	Policy Gradient	Searching
Known Dynamics	Not required	Required
Data (Experience)	Required	Not Required
Reward	Any Step, Any Value	0/1 at the last step
Allow Stochascity	Yes	No

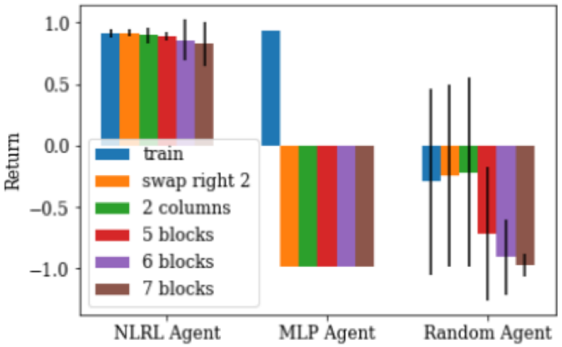
NLRL vs Deep Reinforcement Learning

	NLRL	DRL
Differentialble Architecture	Differeintialble ILP	Neural Network
Interpretability	Good	Poor
Generalizability	Strong	Weak
Allow Continuous Action	No	Yes
Allow Sensory Input	No	Yes

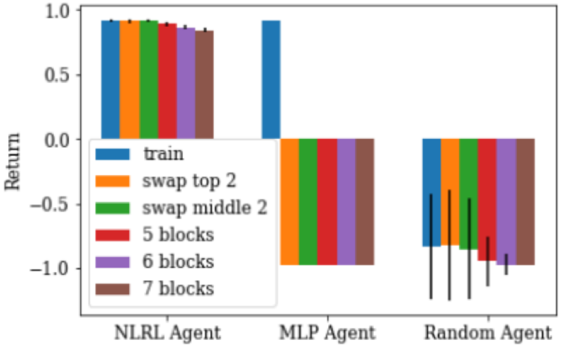
Experiment Results



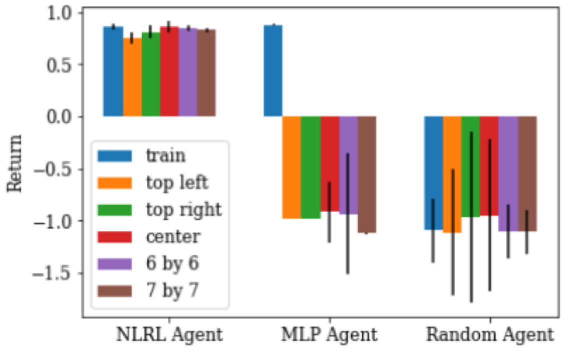
(a) *UNSTACK*



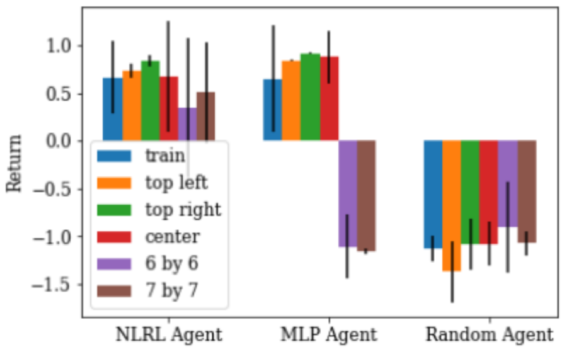
(b) *STACK*



(c) *ON*



(d) Cliff-walking



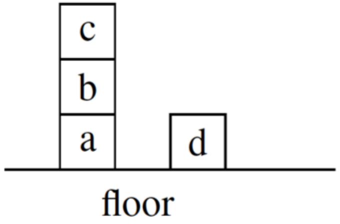
(e) Windy Cliff-walking

Explain the Induced Policies

- Take the UNSTACK as an example:

$$0.972 : move(X,Y) \leftarrow isFloor(Y), pred(X)$$
$$0.987 : pred(X) \leftarrow pred2(X), top(X)$$
$$0.997 : pred2(X) \leftarrow on(X,Y), on(Y,Z)$$

- *pred2* labels blocks whose height is at least 2.
  - In the figure below, {b,c}
- *pred* labels blocks to be moved.
  - On the top and whose height is at least 2. In the figure below, {c}



state: $top(d)$, $top(c), on(d, floor)$, $on(c,b), on(b,a)$, $on(a, floor)$

background: $isFloor(floor)$

Neural Logic Reinforcement Learning

Welcome to our poster: Pacific Ballroom #104

Code available: https://github.com/ZhengyaoJiang/NLRL