Interactively Shaping Agents via Human Reinforcement

W. Bradley Knox 、 Peter Stone
The University of Texas at Austin
K-CAP-2009

Introduction

- Background: As computational learning agents move into domains that incur real costs (e.g., autonomous driving or financial investment), it will be necessary to learn good policies without numerous high-cost learning trials.
- Goal: learn from human to speed learning

Learning from Advice

Learning from Advice

- suggesting an action when a certain condition is true.
- [1]created a domain-specific natural language interface for giving advice to a reinforcement learner.
- general natural language recognition is unsolved
- Moreover, work still remains on how to embed advice into agents that learn from experience.

[1]G. Kuhlmann, P. Stone, R. Mooney, and J. Shavlik. Guiding a reinforcement learner with natural language advice: Initial results in RoboCup soccer

Learning from demonstration

- Human provide trajectories
- Learn reward or policy

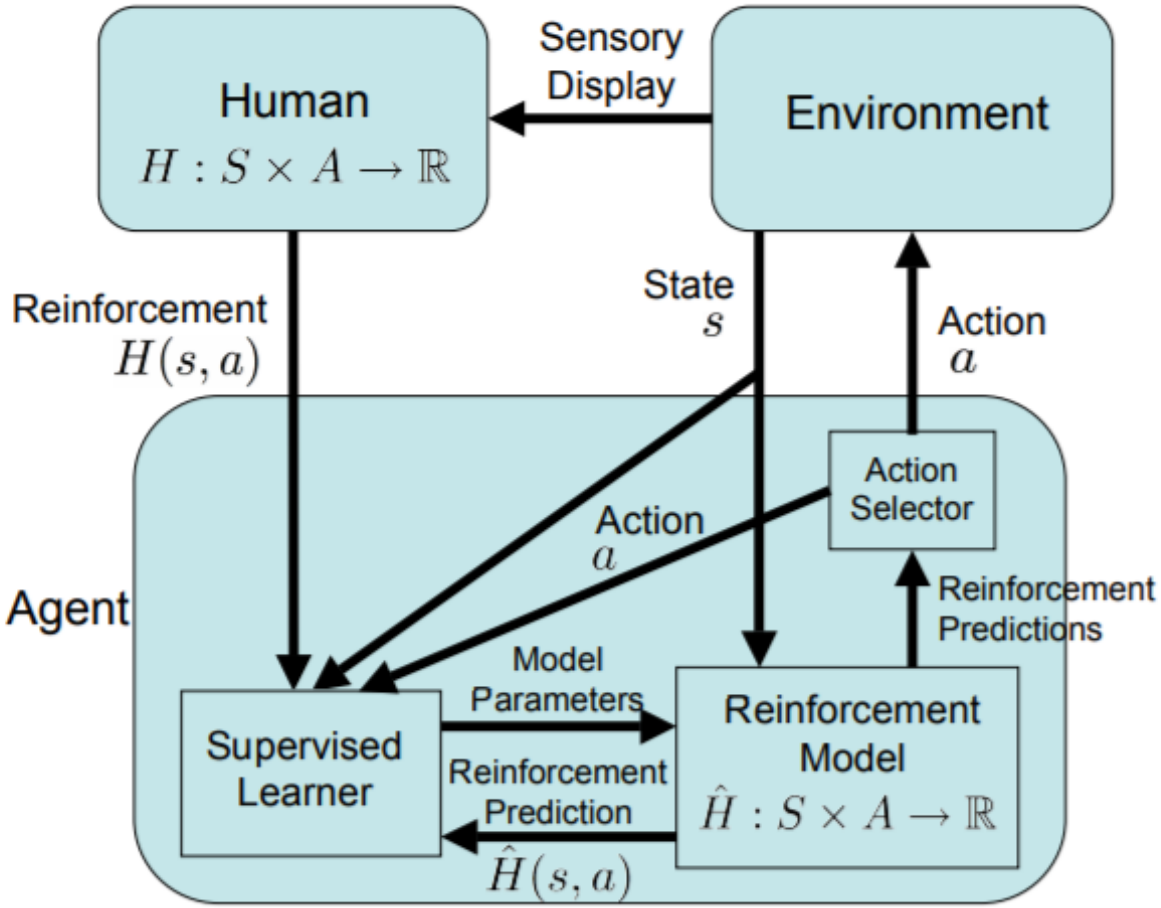
Learning from Reinforcement

Provide human signal to shape the agent

[1] Their agent seeks to maximize the sum of human reinforcement and environmental reward.

A. Thomaz and C. Breazeal. Reinforcement Learning with Human Teachers: Evidence of Feedback and Guidance with Implications for Learning Performance

Framework



MDP/R

Algorithm

Algorithm 1 A general greedy TAMER algorithm

Require: *Input: stepSize*

ReinfModel.init(stepSize)
$\vec{s} \leftarrow \vec{0}$
$\vec{f} \leftarrow \vec{0}$
while *true* **do**
  $h \leftarrow getHumanReinfSincePreviousTimeStep()$
  if $h \neq 0$ **then**
    $error \leftarrow h$ - $ReinfModel.predictReinf(\vec{f})$
    $ReinfModel.update(\vec{f}, error)$
  end if
  $\vec{s} \leftarrow getStateVec()$
  $a \leftarrow argmax_a(ReinfModel.predict(getFeatures(\vec{s}, a)))$
  $\vec{f} \leftarrow getFeatures(\vec{s}, a)$
  $takeAction(a)$
  wait for next time step
end while

模型更新

动作选择

Credit assignment

$$
\begin{aligned}
c_t &= P(event\ starting\ at\ t_i\ was\ targeted) \\
&= P(target\ event\ between\ t_{i-1}\ and\ t_i\ sec.\ ago) \\
&= \int_{t_{i-1}}^{t_i} f(x)dx
\end{aligned}
$$

$$
\sum_{i=1}^{n} P(event\ starting\ at\ t_i\ was\ targeted) = 1
$$

Algorithm2

Require: *Input: stepSize, windowSize,*

1: $Crediter.init(windowSize)$
2: $\overrightarrow{s} \leftarrow \overrightarrow{0}$
3: $\overrightarrow{f} \leftarrow \overrightarrow{0}$
4: $\overrightarrow{w} \leftarrow \overrightarrow{0}$
5: **while** *true* **do**
6: $Crediter.updateTime(clockTime())$
7: $h \leftarrow getHumanReinfSincePreviousTimeStep()$
8: **if** $h \neq 0$ **then**
9: $\overrightarrow{credFeats} \leftarrow 0$
10: **for all** $(\overrightarrow{f_t}, t) \in Crediter.historyWindow$ **do**
11: $c_t \leftarrow Crediter.assignCredit(t)$
12: $\overrightarrow{credFeats} \leftarrow \overrightarrow{credFeats} + (c_t \times \overrightarrow{f_t})$
13: **end for**
14: $error \leftarrow h - (\overrightarrow{w} \cdot \overrightarrow{credFeats})$
15: $\overrightarrow{w} \leftarrow \overrightarrow{w} + (stepSize \times error \times \overrightarrow{credFeats})$
16: **end if**
17: $\overrightarrow{s} \leftarrow getStateVec()$
18: $a \leftarrow argmax_a(\overrightarrow{w} \cdot (getFeatures(\overrightarrow{s}, a)))$
19: $\overrightarrow{f} \leftarrow getFeatures(\overrightarrow{s}, a)$
20: $takeAction(a)$
21: $Crediter.updateWindow(\overrightarrow{f})$
22: wait for next time step
23: **end while**

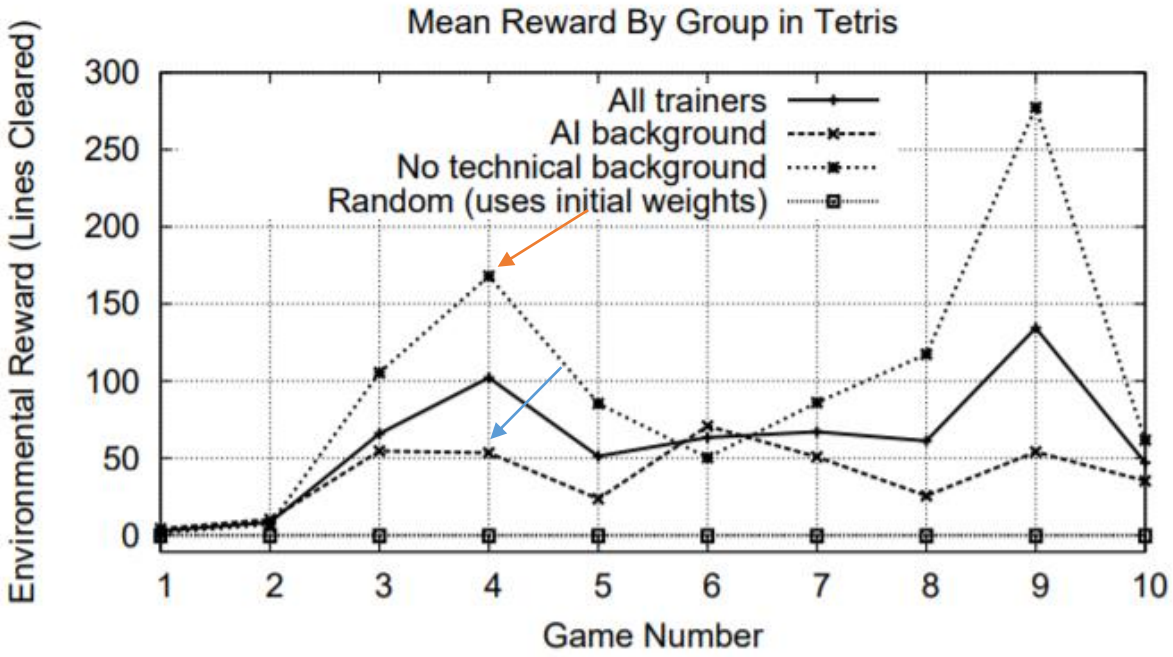
模型更新

动作选择

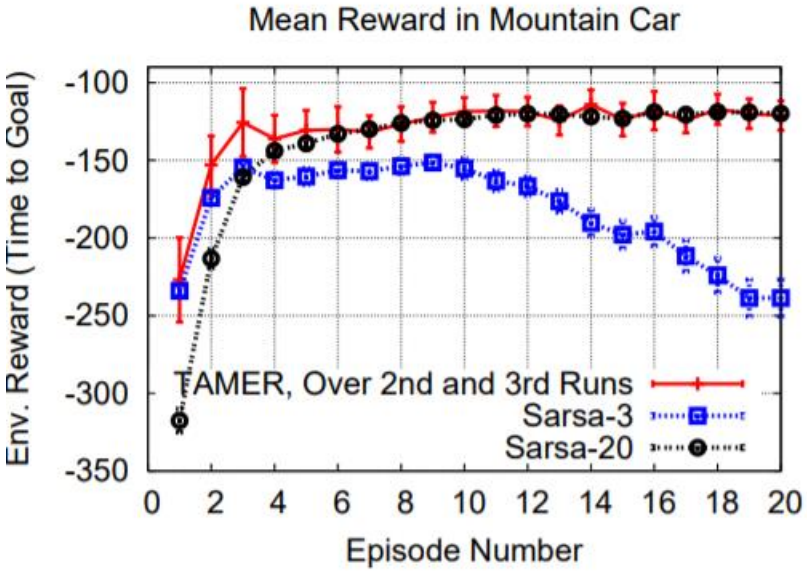
Experiment-Tetris

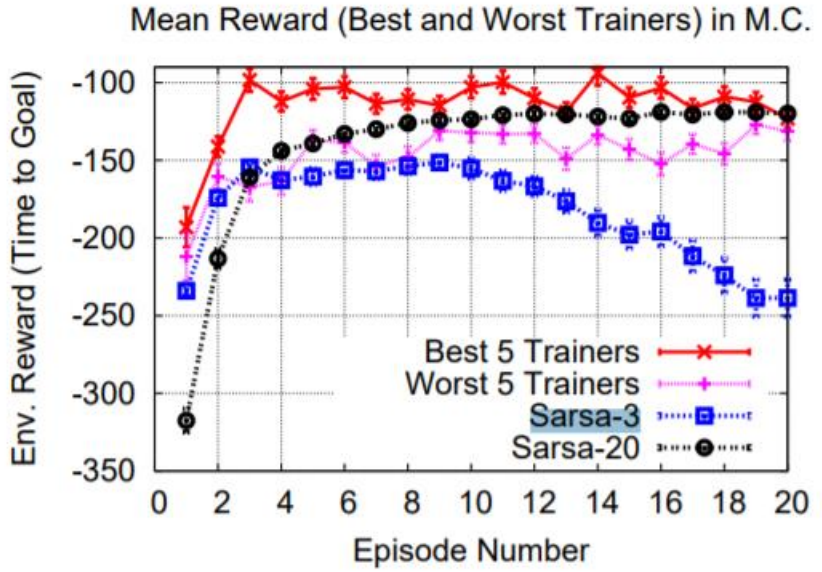
Table 1: Results of various Tetris agents.

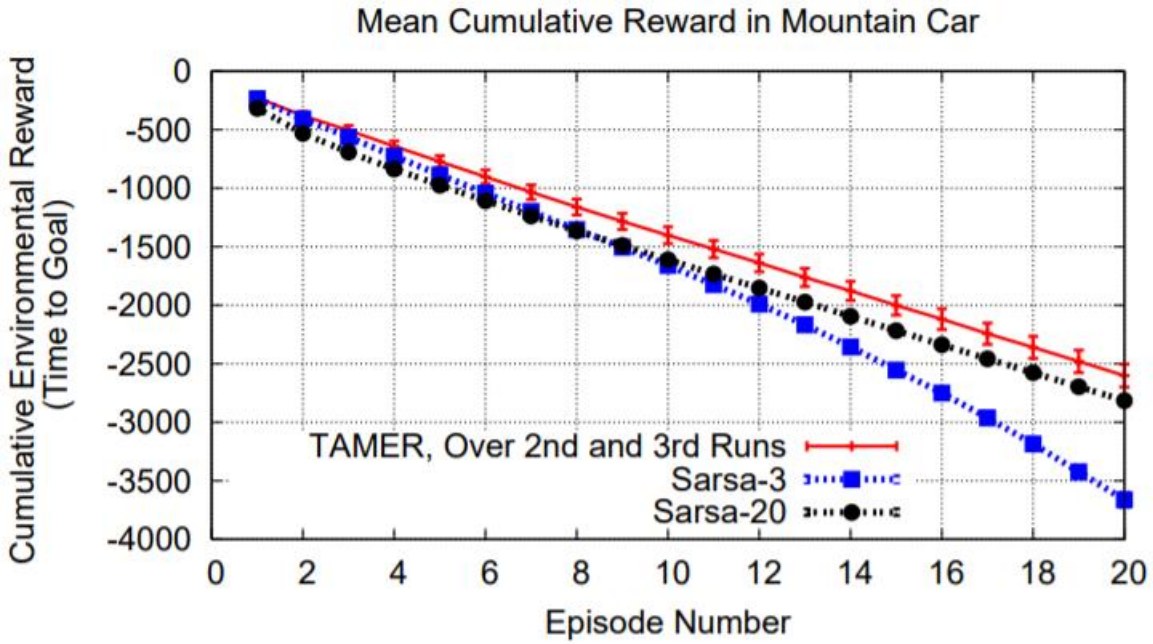
Method	Mean Lines Cleared		Games
	at Game 3	at Peak	for Peak
TAMER	**65.89**	**65.89**	**3**
RRL-KBR [15]	5	50	120
Policy Iteration [2]	~ 0 (no learning until game 100)	3183	1500
Genetic Algorithm [5]	~ 0 (no learning until game 500)	586,103	3000
CE+RL [17]	~ 0 (no learning until game 100)	348,895	5000



Mountain Car





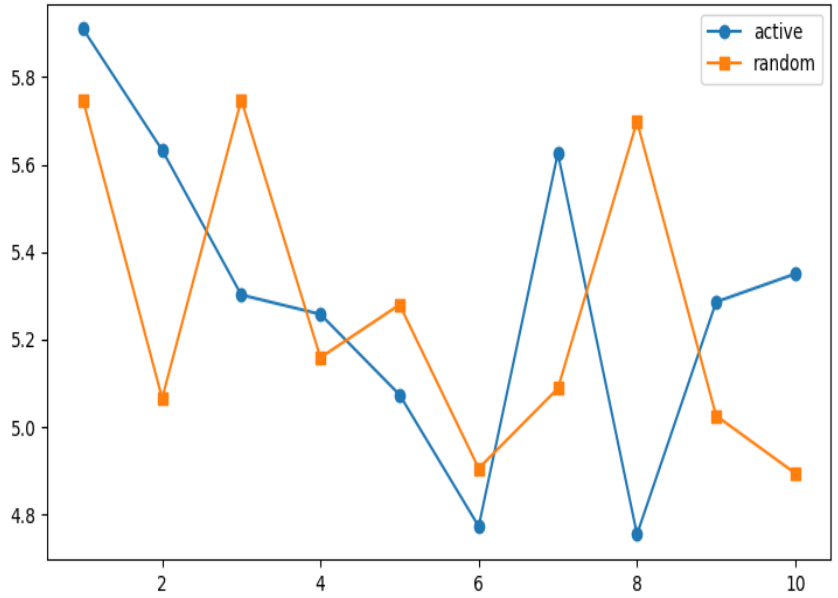


Conclusion

- works in the absence of an environmental reward function,
- reduces sample complexity
- is accessible to people who lack knowledge of computer science

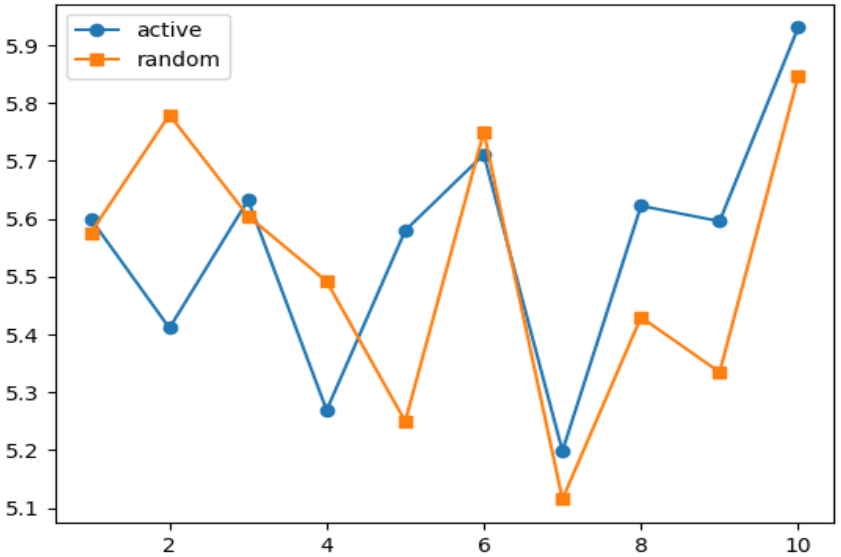
实验

逆强化学习算法：最大熵逆强化学习



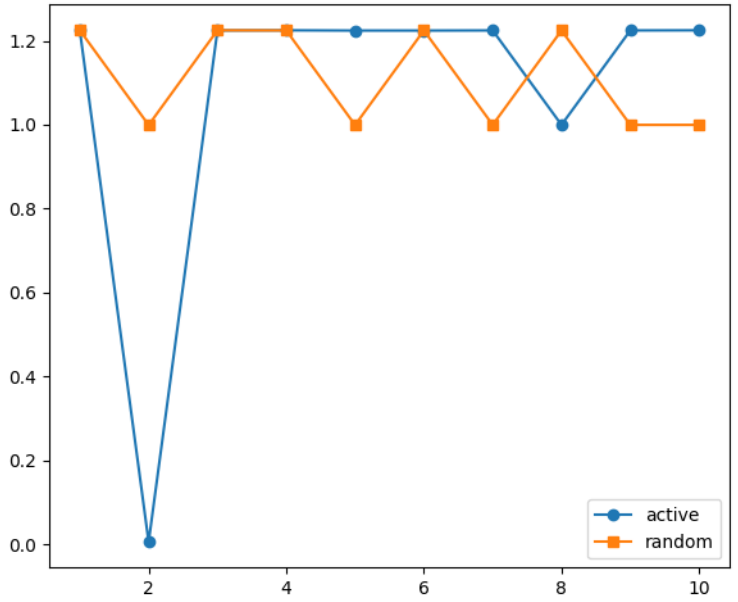
主动选初始状态以及步长vs随机选初始状态和步长

X:轨迹数量
Y:reward loss



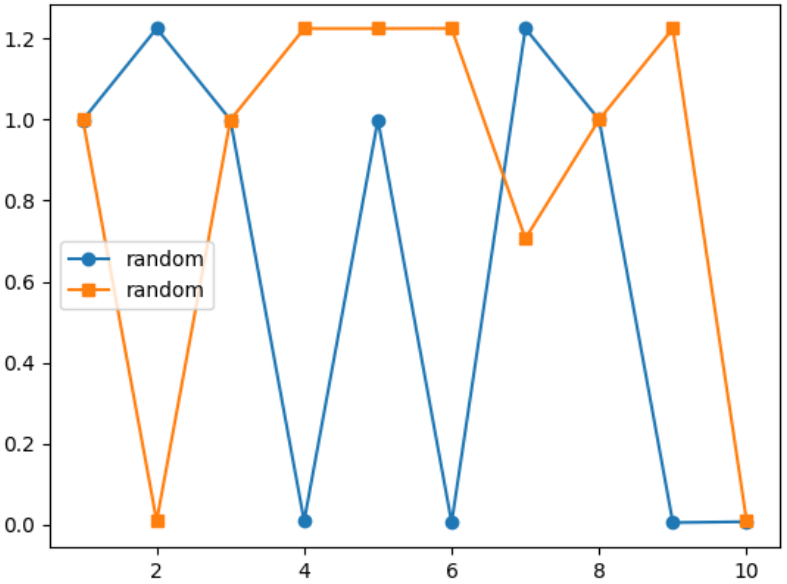
步长相同
主动选初始状态vs随机选初始状态

X:轨迹数量
Y:value loss



步长相同
主动选初始状态vs随机选初始状态

X:轨迹数量
Y:value loss



固定初始状态vs固定选初始状态