



Rewards Prediction-Based Credit Assignment for Reinforcement Learning With Sparse Binary Rewards ——hoho

论文试图解决什么问题？

credit assignment 问题

这是否是一个新的问题？

否

这篇文章要验证一个什么科学假设？

只要识别出哪个动作是关键动作（key-action），还有其前后的相邻关键动作（adjacent-key-actions），然后将adjacent-key-actions也赋予奖励，则采样效率和模型收敛速度也会加快

有哪些相关研究？如何归类？谁是这一课题在领域内值得关注的研究员？

hoho_todo

论文中提到的解决方案之关键是什么？

核心：搭建额外的模型来预测每一步的奖励，识别出key-action。

总体流程如下图：

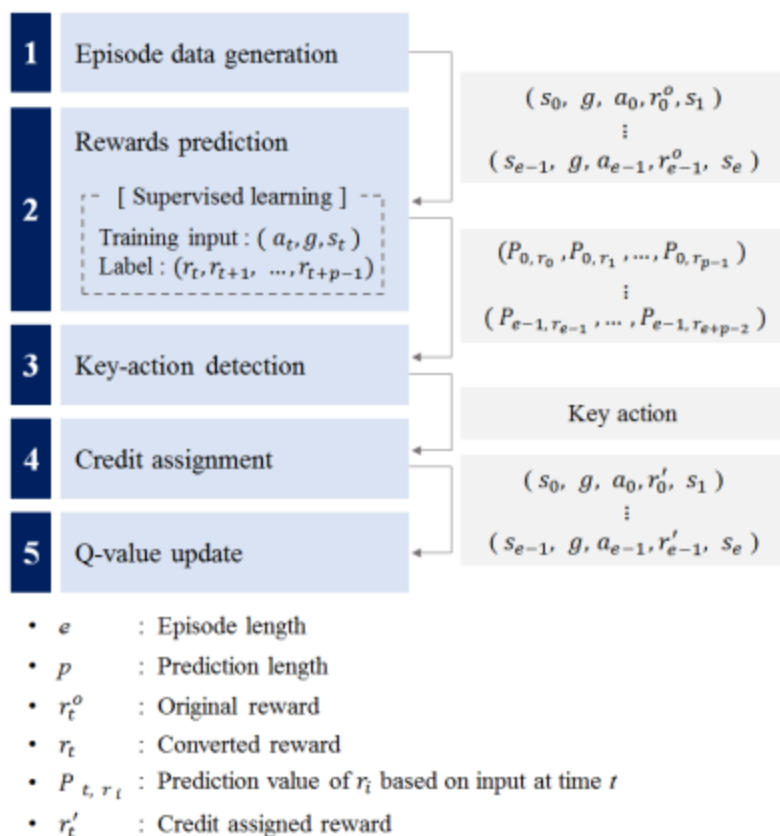


FIGURE 1. Process flow of the proposed method.

具体流程：

1. 用off-policy方式，agent与环境交互，生成回合数据 $(s_{t-1}, g, a_{t-1}, r_{t-1}^o, s_t)$ ， g 表示任务最后的目标
2. 使用1的数据进行奖励预测：
 - 2.1 预测模型的输入数据与标签构建：
 - 输入为： (a_t, g, s_t)
 - 标签构建：

定义一个超参 p ：预测奖励的时间步长度，即基于当前时间步，要预测接下来多长的时间步

converted reward：本文以二分奖励为例（失败奖励为-1，成功奖励为0），将原来的reward为-1的转为0，原来为0的转为1

extended reward：对最后一步而言，因为要预测p步，所以最后一步之后的p步的奖励赋值为跟最后一步的奖励相同

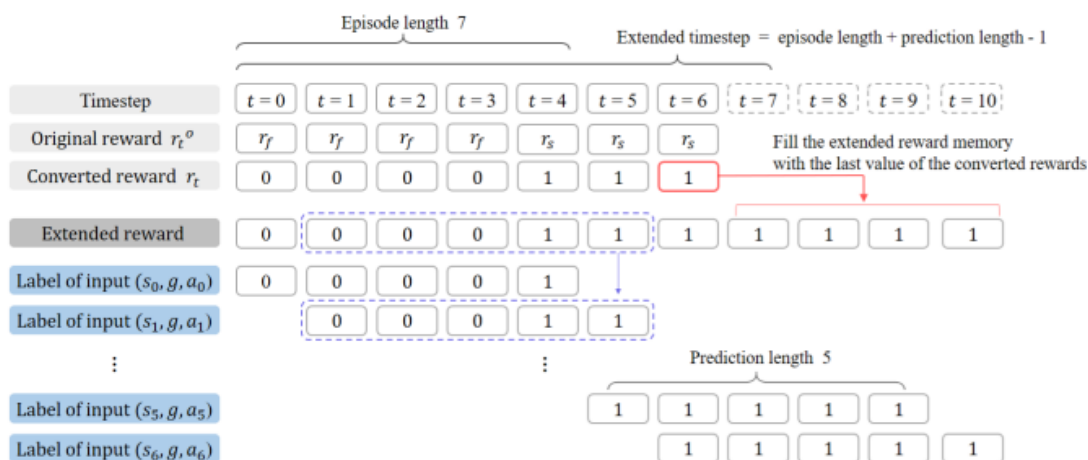


FIGURE 4. Framework for supervised learning of rewards prediction model.

2.2 奖励的预测：

以下的每一行表示每一步对之后p步的奖励预测：

Timestep	$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=6$
Converted reward r_t	0	0	0	0	1	1	1
Input	(s_0, g, a_0)	(s_1, g, a_1)	(s_2, g, a_2)	(s_3, g, a_3)	(s_4, g, a_4)	(s_5, g, a_5)	(s_6, g, a_6)
Predicted rewards at $t=0$	P_{0,r_0} -0.99	P_{0,r_1} -0.39	P_{0,r_2} 0.11	P_{0,r_3} 0.42	P_{0,r_4} 0.68		
Predicted rewards at $t=1$		P_{1,r_1} -0.36	P_{1,r_2} 0.15	P_{1,r_3} 0.53	P_{1,r_4} 0.75	P_{1,r_5} 0.81	
Predicted rewards at $t=2$			P_{2,r_2} 0.13	P_{2,r_3} 0.46	P_{2,r_4} 0.95	P_{2,r_5} 0.98	P_{2,r_6} 0.99
Predicted rewards at $t=3$				P_{3,r_3} 0.45	P_{3,r_4} 0.89	P_{3,r_5} 0.92	P_{3,r_6} 0.94
Predicted rewards at $t=4$					P_{4,r_4} 0.96	P_{4,r_5} 0.96	P_{4,r_6} 0.98
Predicted rewards at $t=5$						P_{5,r_5} 0.97	P_{5,r_6} 0.99
Predicted rewards at $t=6$							P_{6,r_6} 0.98

P_{t,r_e} 表示基于第t步预测的第e步的奖励

3. key-actions识别

基于以下规则进行对前后奖励计算差分（这同一列中进行）：

$$\Delta P_{t,r_l} = \begin{cases} 0, & \text{if } P_{t-1,r_l} \text{ does not exist} \\ P_{t,r_l} - P_{t-1,r_l}, & \text{if } P_{t-1,r_l} \text{ exists} \end{cases} \quad ($$

因为奖励变化大的地方极有可能跟关键奖励相关（Since abrupt change of the predicted rewards according to input over timesteps is our concern）

于是得到如下统计数据：

Timestep	t = 0	t = 1	t = 2	t = 3	t = 4	t = 5	t = 6
Converted reward r_t	0	0	0	0	1	1	1
Differential predicted rewards at t = 0	$\Delta P_{0,r_0}$ 0.00	$\Delta P_{0,r_1}$ 0.00	$\Delta P_{0,r_2}$ 0.00	$\Delta P_{0,r_3}$ 0.00	$\Delta P_{0,r_4}$ 0.00		
Differential predicted rewards at t = 1		$\Delta P_{1,r_1}$ 0.03	$\Delta P_{1,r_2}$ 0.04	$\Delta P_{1,r_3}$ 0.11	$\Delta P_{1,r_4}$ 0.07	$\Delta P_{1,r_5}$ 0.00	
Differential predicted rewards at t = 2			$\Delta P_{2,r_2}$ -0.02	$\Delta P_{2,r_3}$ -0.07	$\Delta P_{2,r_4}$ 0.20	$\Delta P_{2,r_5}$ 0.17	$\Delta P_{2,r_6}$ 0.00
Differential predicted rewards at t = 3				$\Delta P_{3,r_3}$ -0.01	$\Delta P_{3,r_4}$ -0.06	$\Delta P_{3,r_5}$ -0.06	$\Delta P_{3,r_6}$ -0.05
Differential predicted rewards at t = 4					$\Delta P_{4,r_4}$ 0.07	$\Delta P_{4,r_5}$ 0.04	$\Delta P_{4,r_6}$ 0.02
Differential predicted rewards at t = 5						$\Delta P_{5,r_5}$ 0.01	$\Delta P_{5,r_6}$ 0.01
Differential predicted rewards at t = 6							$\Delta P_{6,r_6}$ 0.01
Key action candidate	-	-	-	-	a_2	a_2	a_4

取出差分值大的时间步(以上红框)对应的action作为候选key-actions：

Action	a_0	a_1	Key action a_2	a_3	a_4	a_5	a_6
Nomination frequency	0	0	2	0	1	0	0

统计候选action的频次，可选频次大的为key-action（本文选两个频次最大的actions作为key-actions）

4. credit assignment

本文采用两种CA策略：

- CA_1: 分配key-action之前的action为adjacent-key-action

	Adjacent-key-actions		Key-action				
a_t	a_0	a_1	a_2	a_3	a_4	a_5	a_6
r_t^o	r_f	r_f	r_f	r_f	r_s	r_s	r_s
r_t'	R_2	R_1	R_0	r_f	r_s	r_s	r_s
Distribution length D							

(a) CA_1 strategy with symbols

a_t	a_0	a_1	a_2	a_3	a_4	a_5	a_6
r_t^o	-1	-1	-1	-1	0	0	0
r_t'	$-\frac{2}{3}$	$-\frac{1}{3}$	0	-1	0	0	0

(b) CA_1 strategy with numbers

FIGURE 7. Credit assignment strategy CA_1.

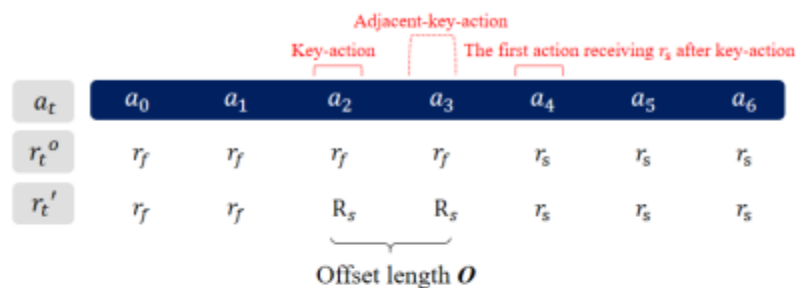
R_i 的计算方法：

$$R_i = r_s + \left(\frac{r_f - r_s}{D} \right) i,$$

r_f 为原来的失败奖励值， r_s 为原来的成功的奖励值

(图b为例子)

- CA_2：分配key-action和首次获得成功奖励的action之间的步骤为adjacent-key-action



(a) CA_2 strategy with symbols

a_t	a_0	a_1	a_2	a_3	a_4	a_5	a_6
r_t^o	-1	-1	-1	-1	0	0	0
r_t^i	-1	-1	0	0	0	0	0

(b) CA_2 strategy with numbers

FIGURE 8. Credit assignment strategy CA_2.

key-action和adjacent-key-actions都赋予成功的奖励值。

论文中的实验是如何设计的？

设计了Fetch push（机器人推箱子到目的地）和Fetch slide（机器人滑动撞击滑块到目标点）实验

用于定量评估的数据集是什么？代码有没有开源？

都无

论文中的实验及结果有没有很好地支持需要验证的科学假设？

CA_1和CA_2收敛都比普通的DDPG+HER方法快

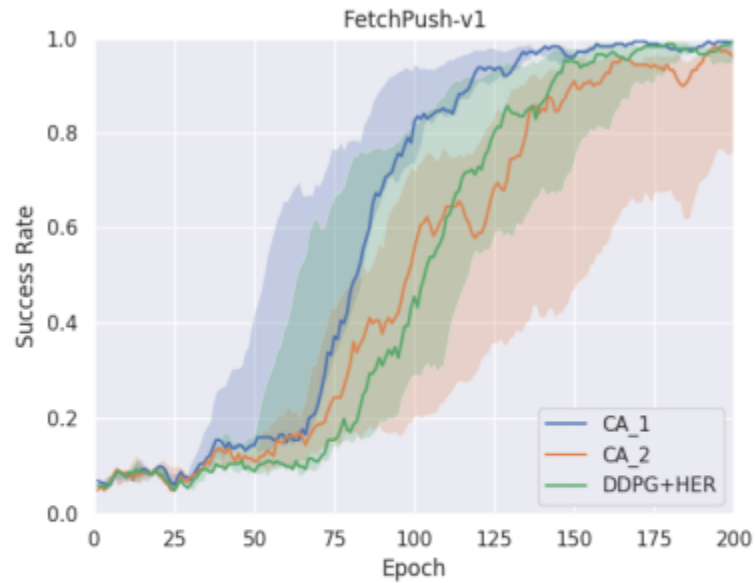


FIGURE 10. Success rate obtained by proposed method and DDPG+HER for the Fetch push task. Results show that proposed method achieves similar performance to that of DDPG+HER over 200 epochs. Success rates of CA_1 strategy, CA_2 strategy, DDPG+HER are 100%, 98%, 98%, respectively. It is seen that CA_1 strategy converges faster as compared to others.

CA_1和CA_2最后的Q值都和普通的DDPG+HER几乎一样

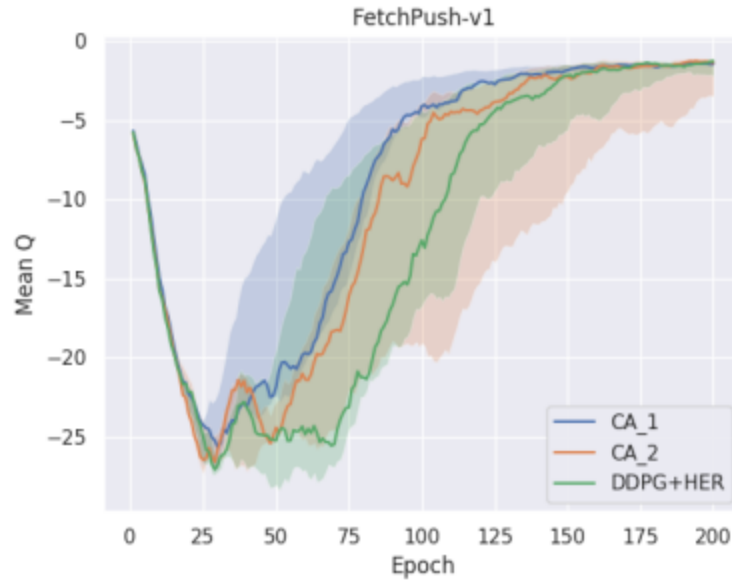


FIGURE 11. Variation of mean Q value during training with DDPG+HER and proposed method (CA_1 and CA_2 strategies) for the Fetch push task. At epoch index 200, the mean Q value is approximately -2 with DDPG+HER and proposed method. It is seen that CA_1 and CA_2 strategies converge to the optimum mean Q value -2 faster, which means more efficient exploration during training.

至于成功率CA_1较高

TABLE 1. Comparison between the proposed method and DDPG+HER for the Fetch push task.

Algorithm	Push succ %
DDPG+HER	$97.67\% \pm 1.5\%$
CA_1 strategy	$99.17\% \pm 0.15\%$
CA_2 strategy	$88.83\% \pm 7.25\%$

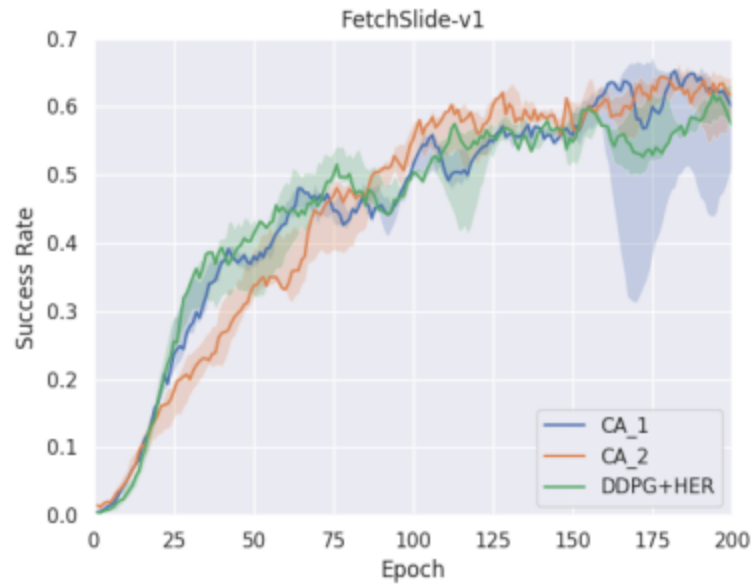


FIGURE 12. Success rate obtained by proposed method and DDPG+HER for Fetch slide task. Results show that the proposed method achieves similar performance to that of DDPG+HER over 200 epochs. Success rates of CA_2 strategy, CA_1 strategy, DDPG+HER are 62%, 60%, 57%, respectively. This success rate accounts for the Fetch slide task in near zone as well, which conceals higher success rate in far zone with the CA_2 strategy. The higher success rate in far zone is referred to Fig. 14.

这篇论文到底有什么贡献？

hoho_todo

下一步呢？有什么工作可以继续深入？

hoho_todo