最新实验进展

宁鲲鹏

2019-4-18

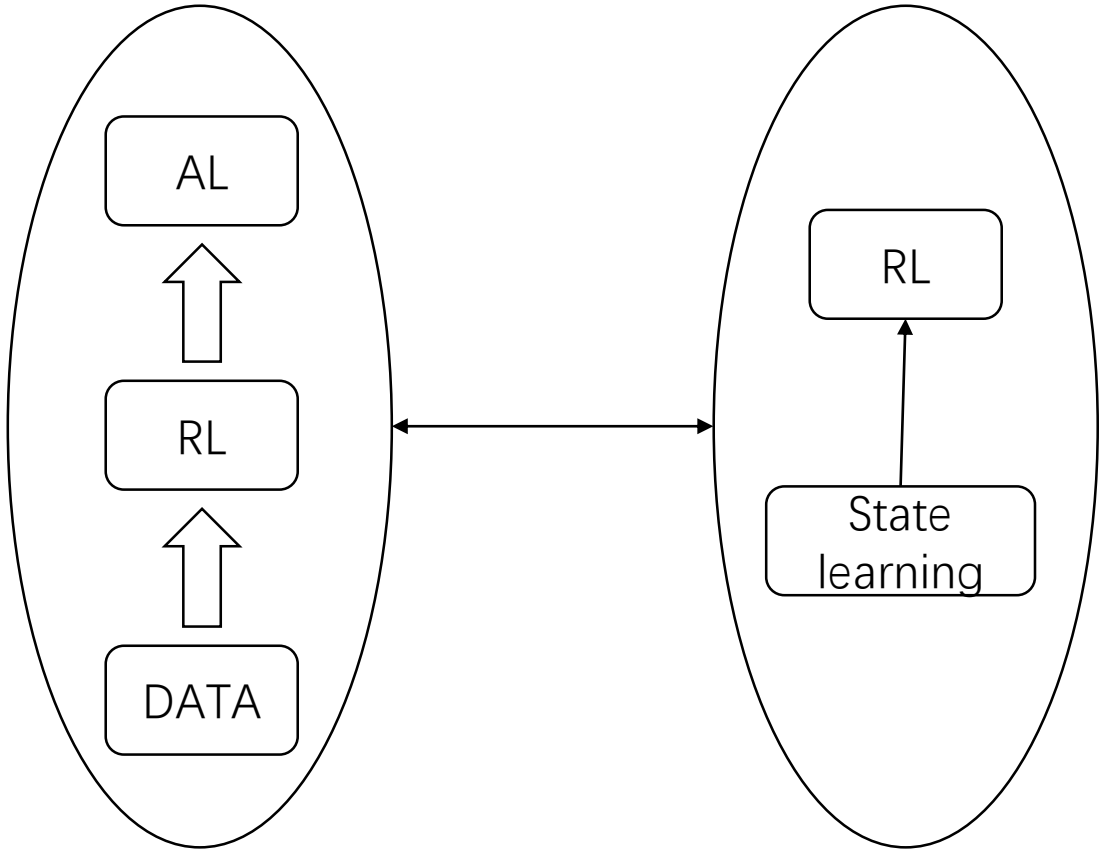
Contents

- Review
- Experiments
- Comparison
- Some experiments about inverse reinforcement learning

Review-plan

- 将状态网络直接部署到强化学习任务上看是否有效果
- 强化学习部分使用actor网络进行试验，从目标上感觉比DQN更合理

Review-idea



RL problem

提升强化学习效果

Experiments

- Try to learn state function via the thought of inverse reinforcement learning.
- Design an algorithm of learning state function.
- Make experiments with “CartPole” and “MountainCar”.

Comparison – State learning VS. Reward learning

- 1. Algorithm is different.
- 2. Network structure is different.
- 3. The parameters to be studied are different.
- 4. Trajectories generated by expert are different.
- 5. The process of parameter optimization is different.

Differences in algorithms

Reward learning

输入: 环境 E;
状态空间 X;
动作空间 A;
范例轨迹数据集 $D = \{\tau_1, \tau_2, \ldots, \tau_m\}$.

过程:

1: $\bar{\boldsymbol{x}}^* =$ 从范例轨迹中算出状态加权和的均值向量;
2: $\pi =$ 随机策略;
3: **for** $t = 1, 2, \ldots$ **do**
4: $\quad \bar{\boldsymbol{x}}_t^{\pi} =$ 从 π 的采样轨迹算出状态加权和的均值向量;
5: $\quad$ 求解 $\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} \min_{i=1}^{t} \boldsymbol{w}^{\mathrm{T}}(\bar{\boldsymbol{x}}^* - \bar{\boldsymbol{x}}_i^{\pi}) \quad \text{s.t.} \quad \|\boldsymbol{w}\| \leqslant 1$;
6: $\quad \pi =$ 在环境 $\langle X, A, R(\boldsymbol{x}) = \boldsymbol{w}^{*\mathrm{T}}\boldsymbol{x} \rangle$ 中求解最优策略;
7: **end for**

输出: 奖赏函数 $R(\boldsymbol{x}) = \boldsymbol{w}^{*\mathrm{T}}\boldsymbol{x}$ 与策略 π

State learning

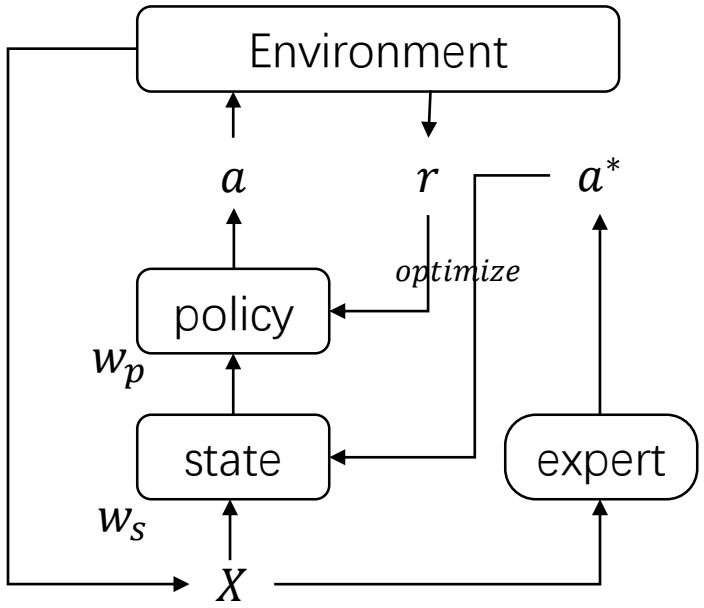
Algorithm

1. π= 随机策略
2. for t=1,2,⋯,do
3. 得到一组轨迹< $x_1^t, a_1^{*t}, r_1^t, x_2^t, a_2^{*t}, r_2^t, \ldots >$.
4. $W_s = argmin_{w_s} \frac{1}{n} \sum_{j=1}^{n} a_j^* \log(a_j)$, 优化$W_s$
5. π=在环境< $f_{w_s}(x), A, R >$ 中求解最优策略
6. end for

Output: f_{w_s}状态函数和策略π

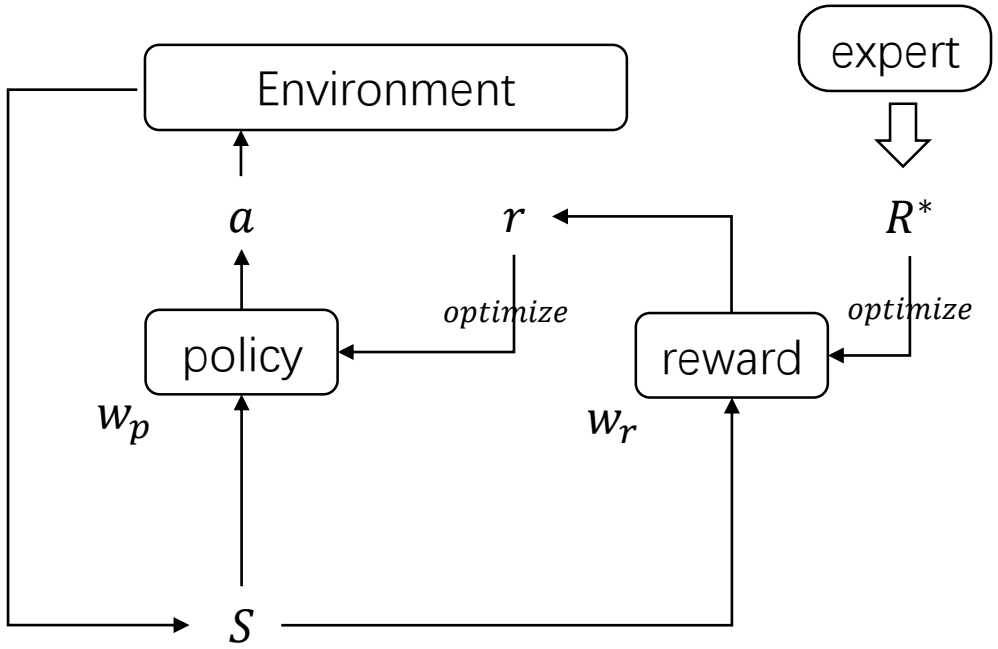
Differences in network

State function Learning



$VS.$

Reward function Learning



Differences in parameters and trajectories

- Parameters
  - $S = f_{w_s}(x)$ $\quad\quad$ $\mathrm{r} = R(S) = w_r^T S$
- Trajectories
  - 在迭代过程中生成范例轨迹数据D'
  - $D' = \{\tau_1', \tau_2', \tau_3', \cdots, \tau_m'\}$
  - $\tau_t' =< x_1^t, a_1^{*t}, r_1^t, x_2^t, a_2^{*t}, r_2^t, \dots >$

范例轨迹数据D提前给出

$D = \{\tau_1, \tau_2, \tau_3, \dots, \tau_m\}$

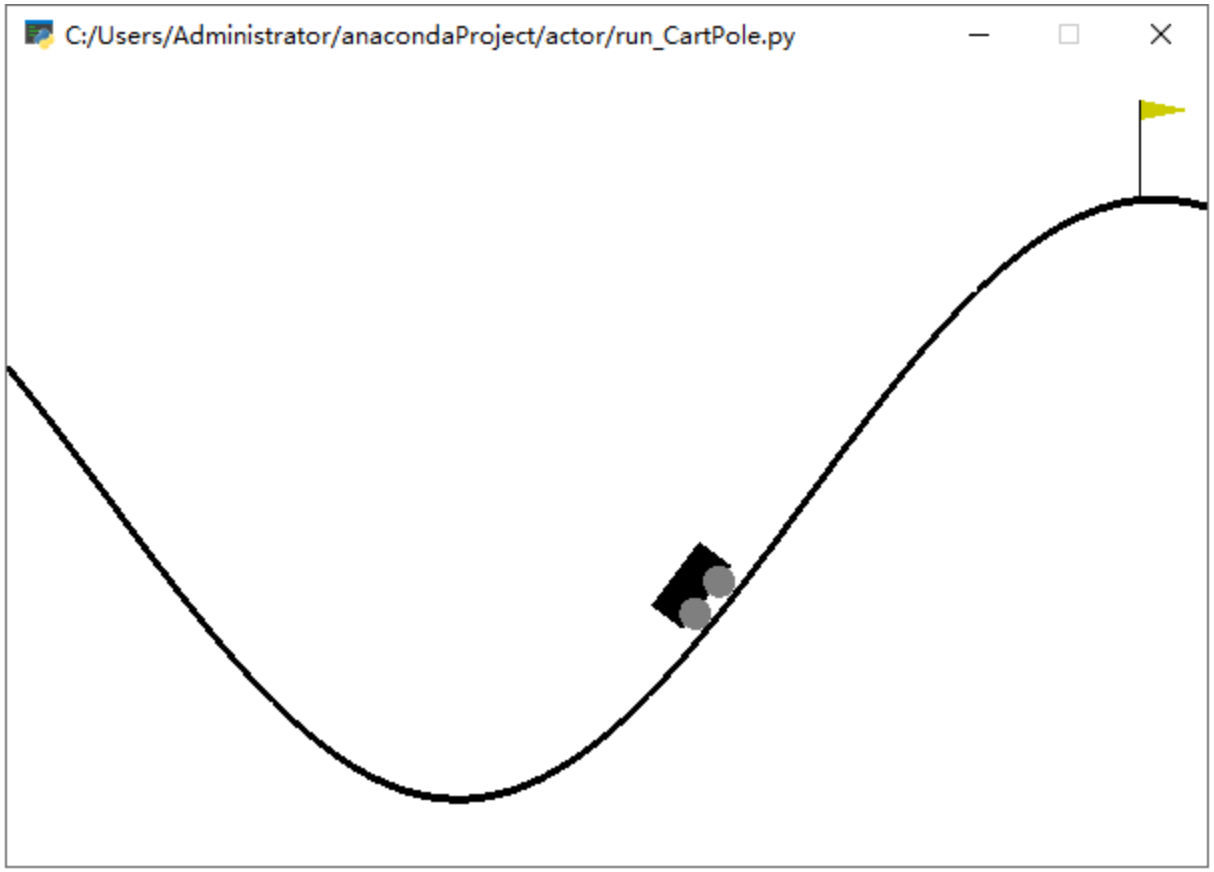
$\tau_t =< s_1^t, a_1^{*t}, x_2^t, a_2^{*t}, \dots >$.

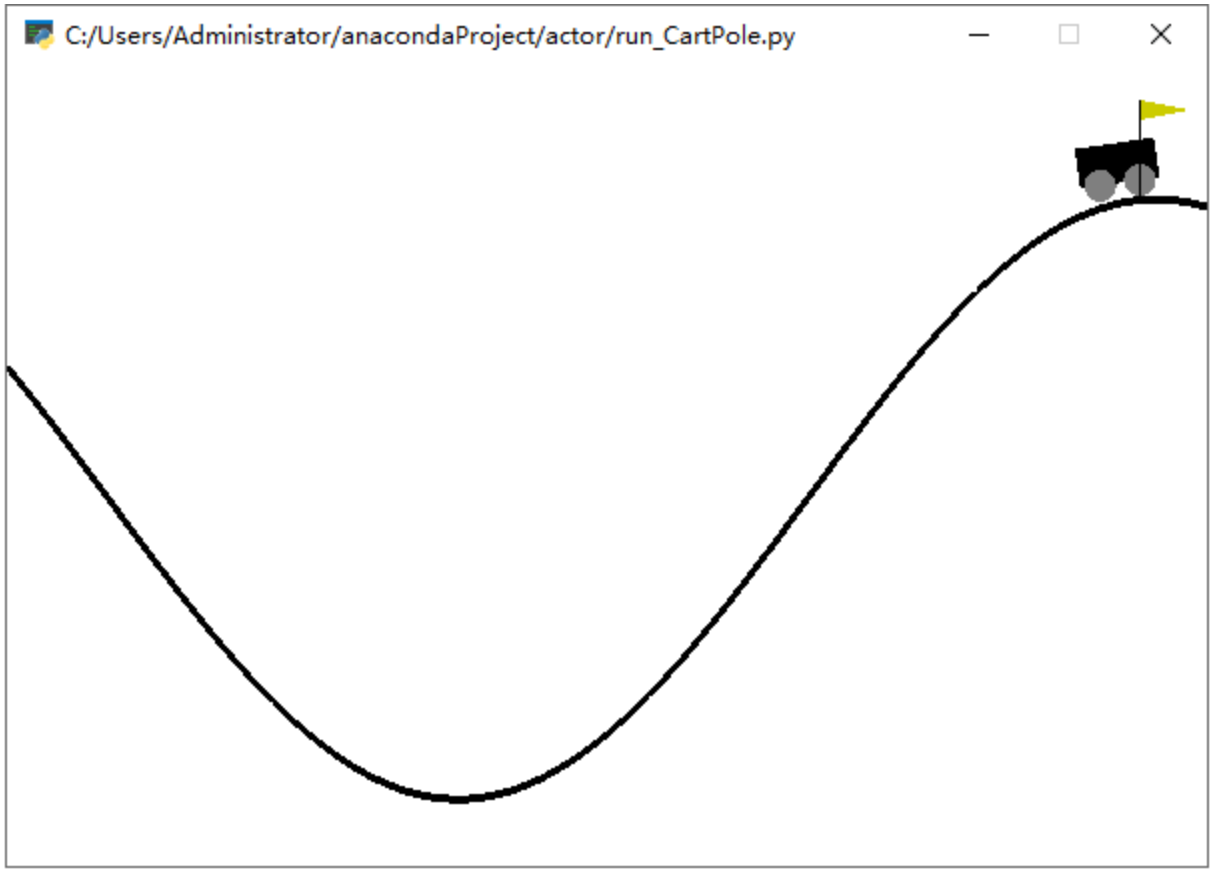
Differences in parameter optimization

- Reward function learning
  - $\bar{X}^* = \frac{1}{m}\sum_{i=1}^{m}\sum_{t=1}^{n_i}\gamma^t S_t^i$
  - $\bar{X}_t^{\pi} = \mathrm{E}[\sum_{i=1}^{n}\gamma^i S_i \,|\pi]^{(t)}$
  - $w_r^* = argmax_{w_r} min_{i=1}^{t} w_r^T(\bar{X}^* - \bar{X}_t^{\pi}) \qquad s.t. \left|\left|w_r\right|\right| \leq 1$

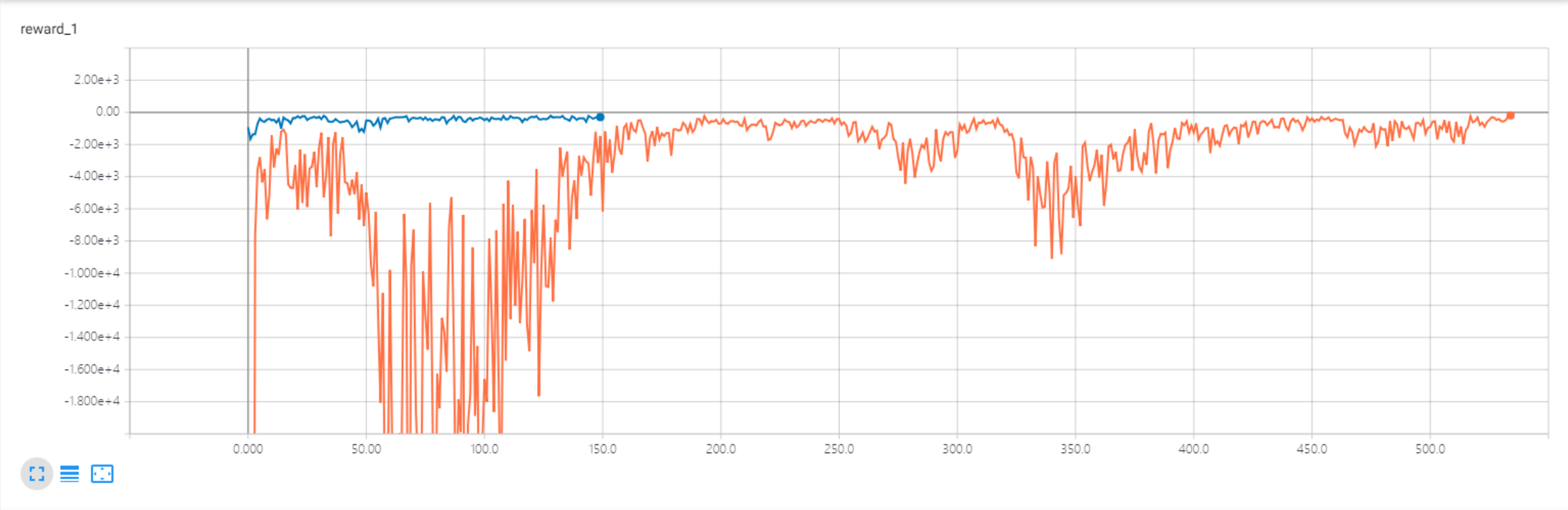
- State function learning
  - $w_s = argmin_{w_s}\frac{1}{n}\sum_{j=1}^{n} a_j^* \log(a_j)$
  - $where\ \ a_j = f_{w_p}(f_{w_s}(x_j))$

Experiments-MountainCar

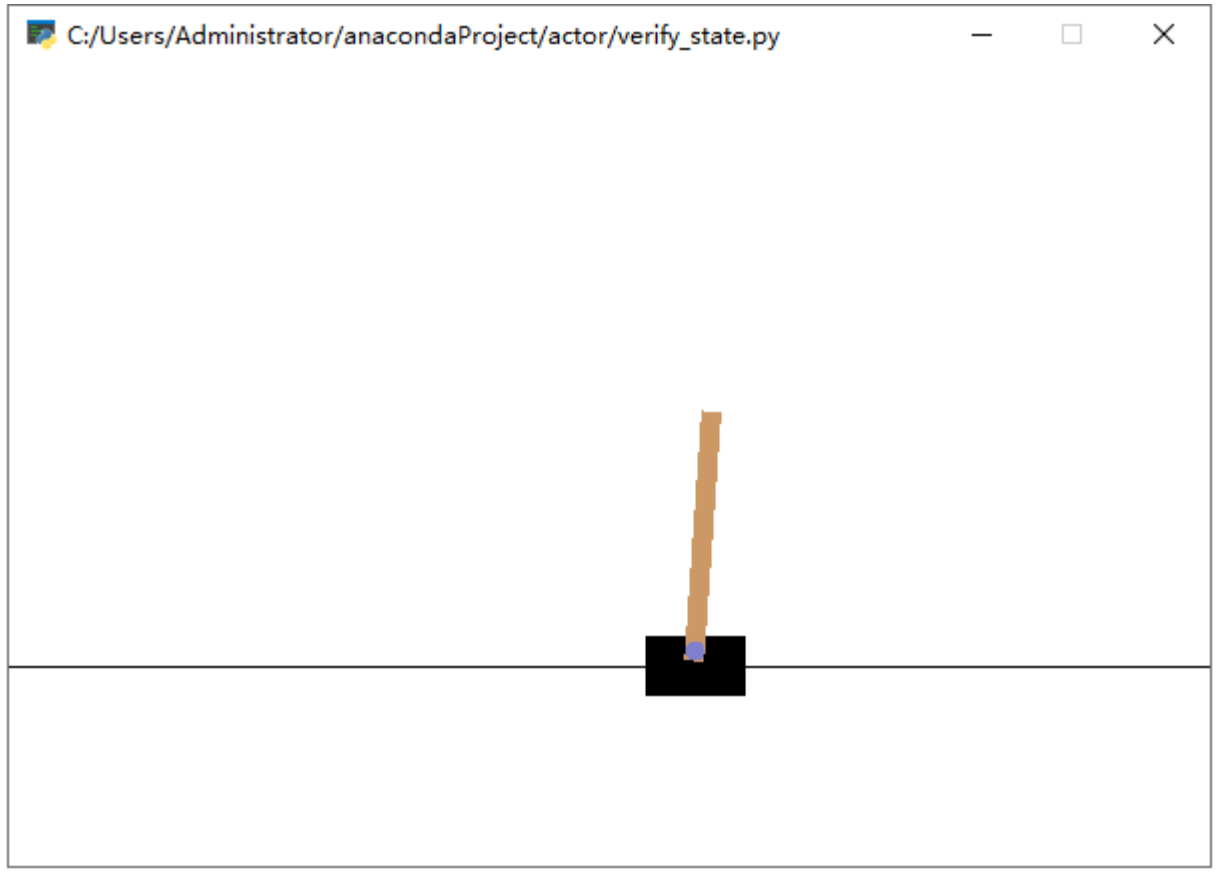


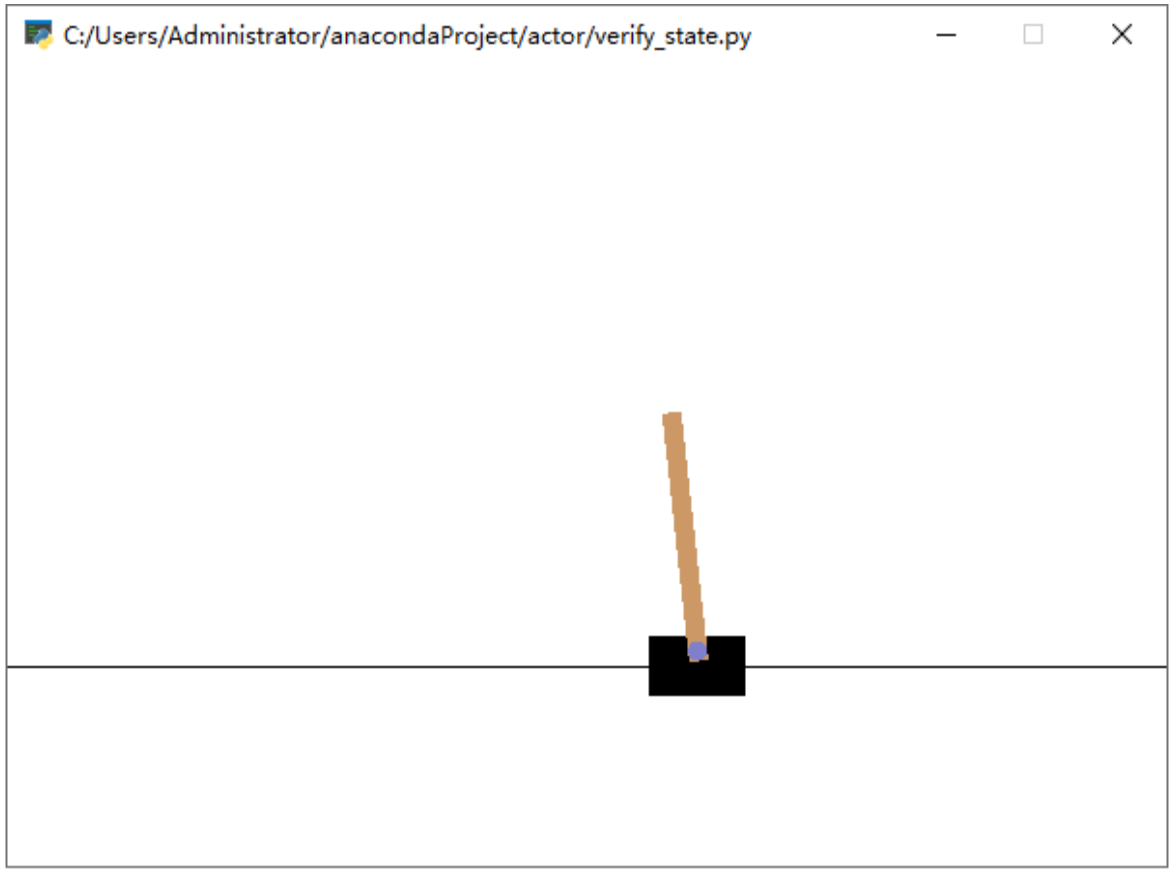


Experiments-MountainCar

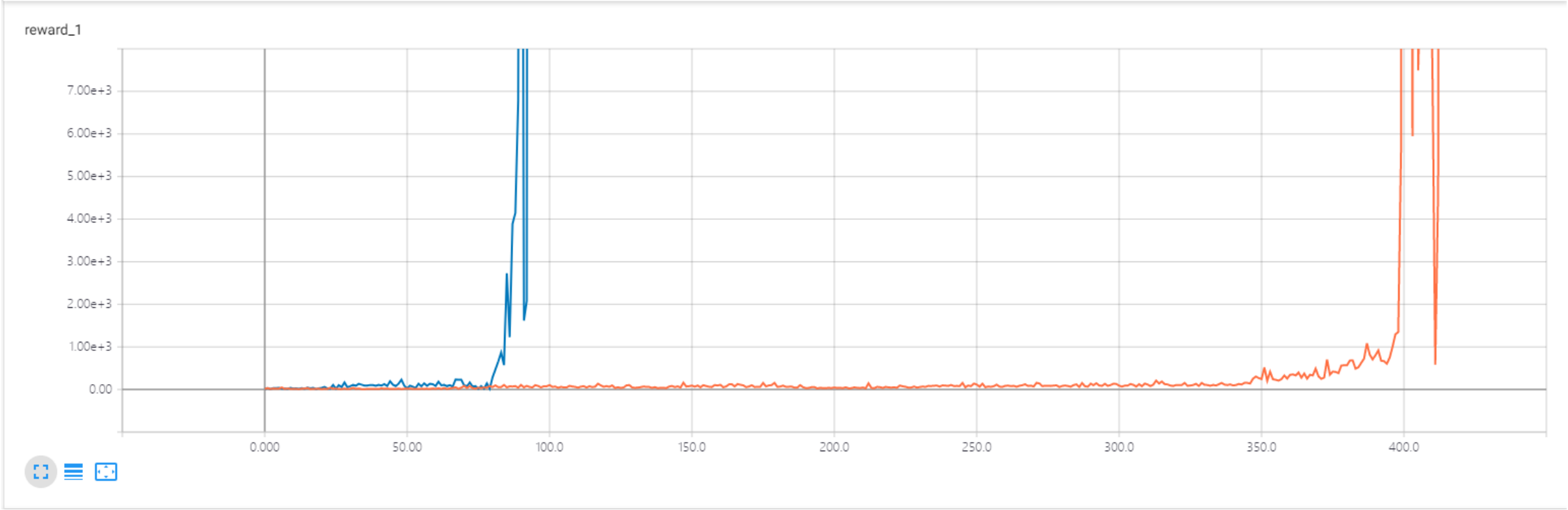


Experiments-CartPole

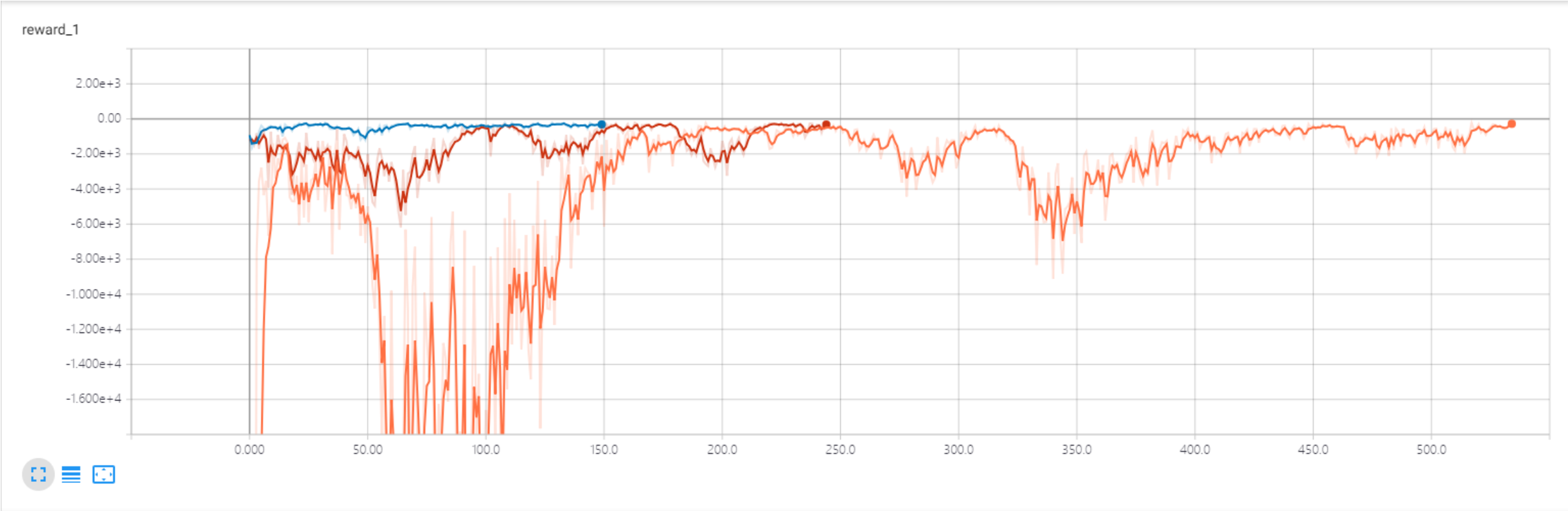




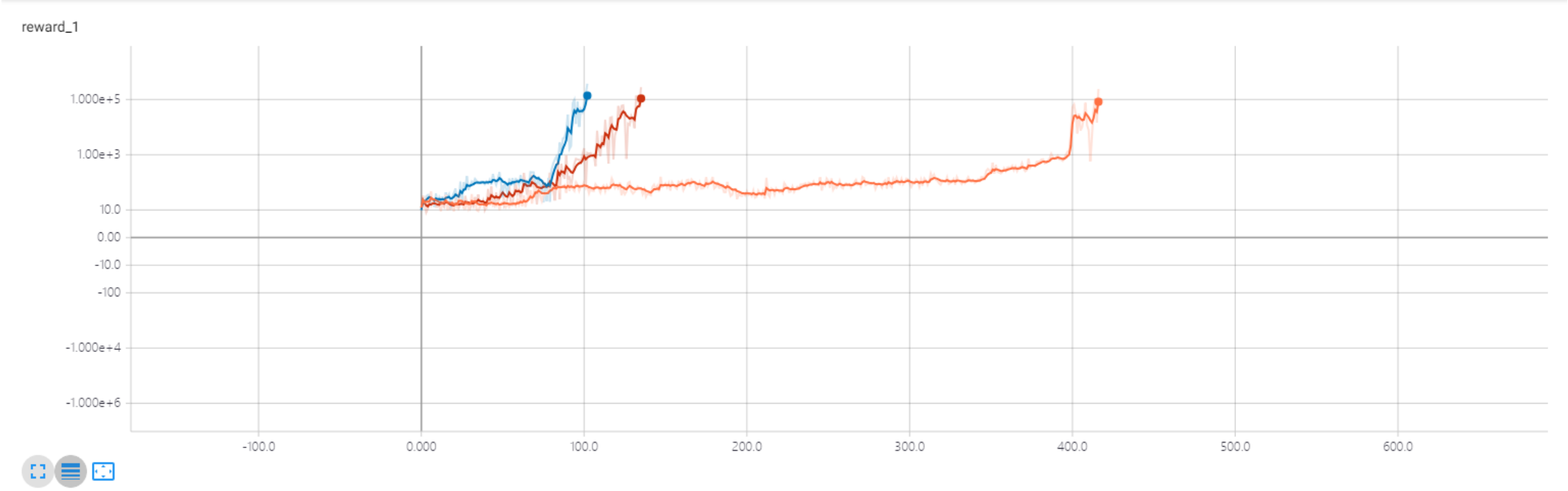
Experiments-CartPole



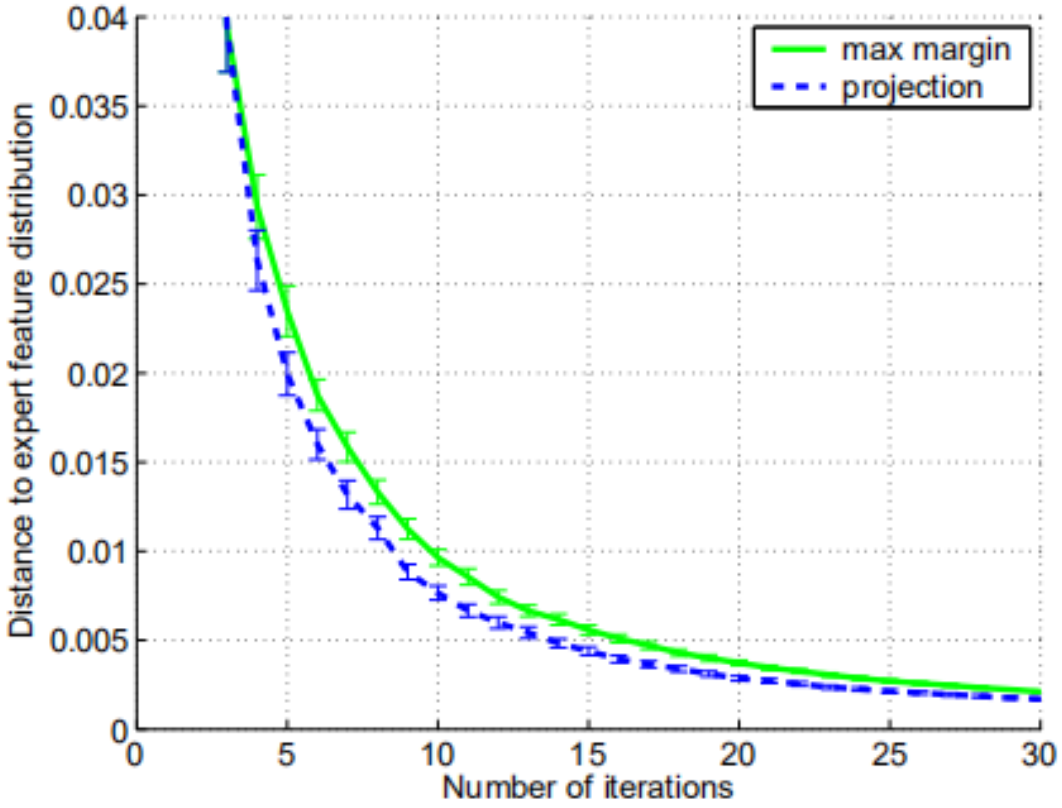
Experiments-MountainCar

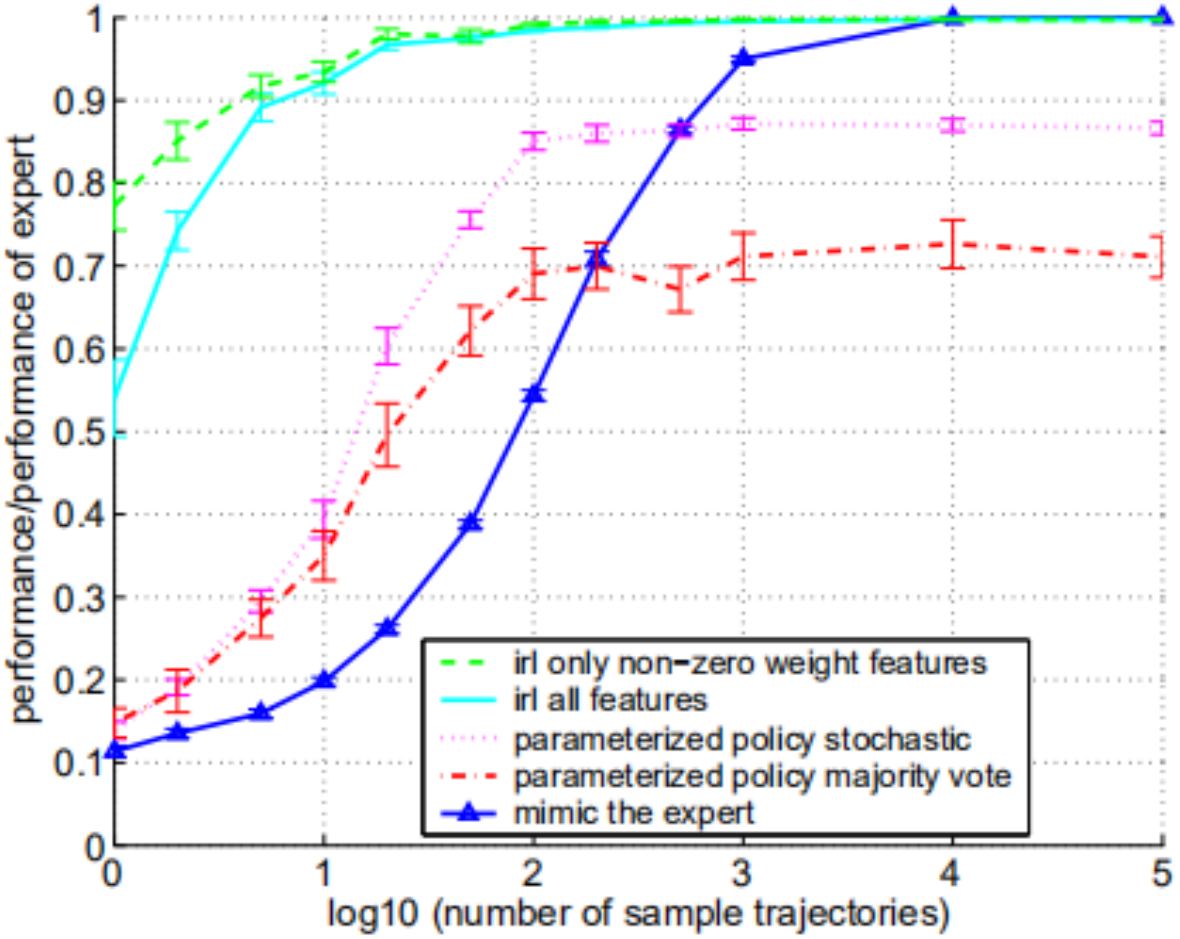


Experiments-CartPole

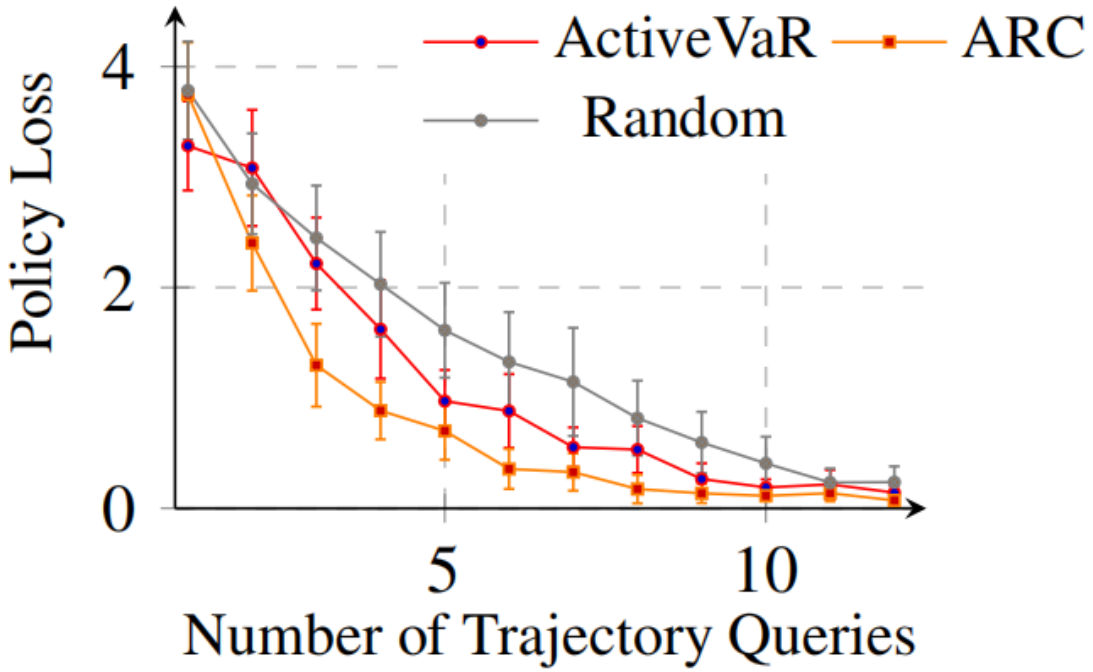


Some Experiments-Grid World(出现在很多论文的实验中)





Some Experiments-Risk Aware Active Inverse Reinforcement Learning



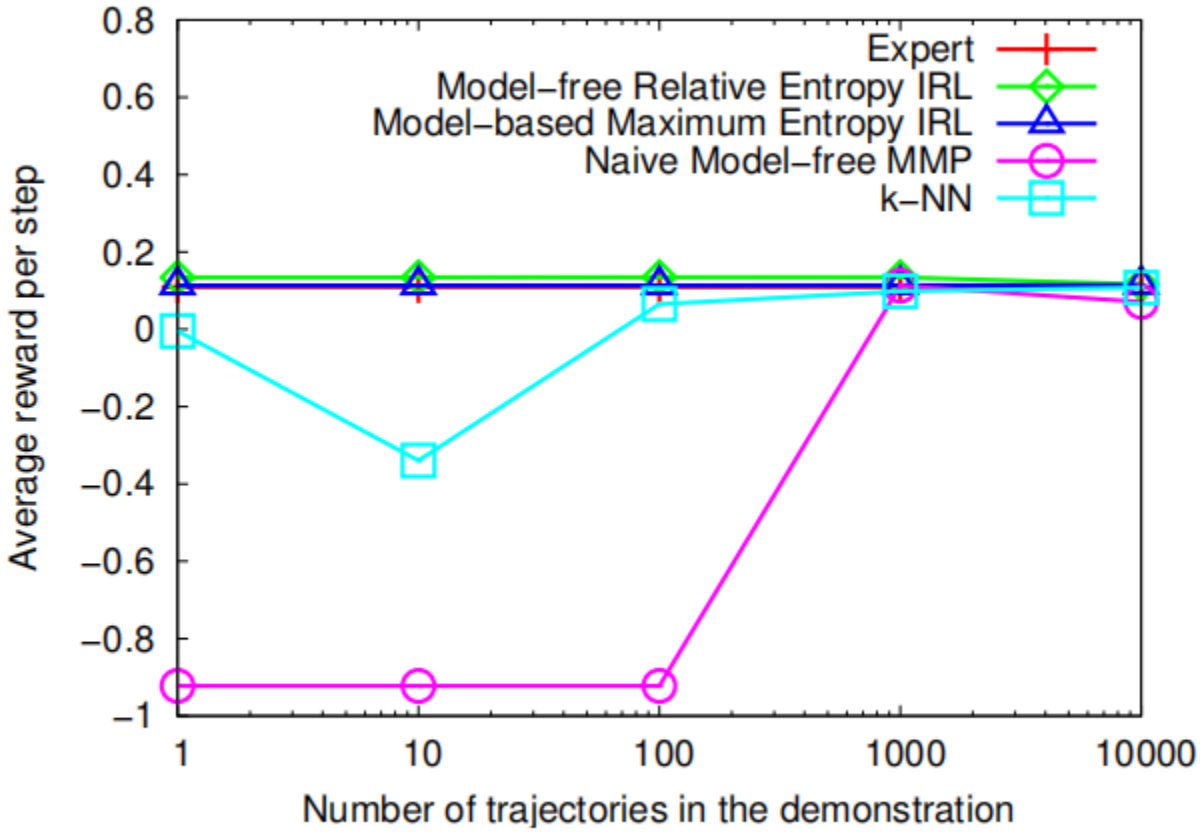
(a) Averaged policy losses

Algorithm	Avg. Time (s)
Random	0.0015
ActiveVaR	0.0101
ARC	865.6993

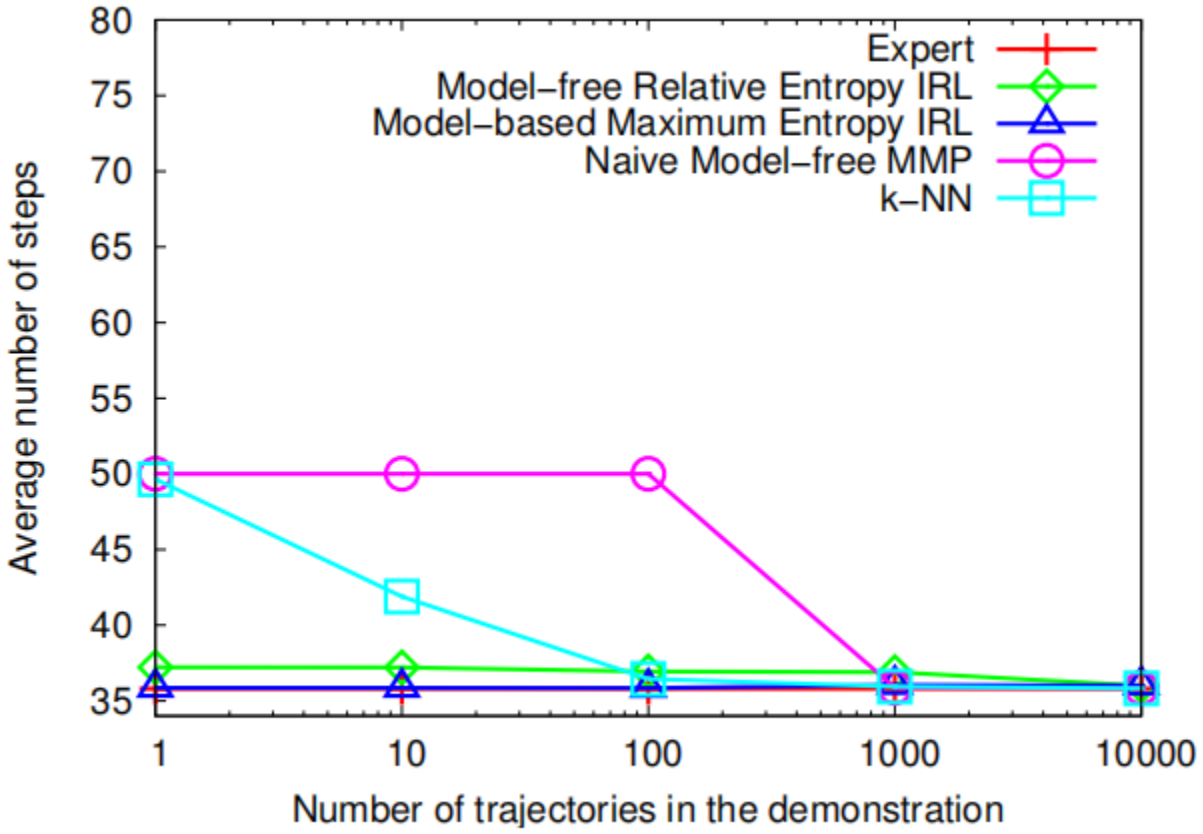
(b) Timing for one iteration of each algorithm

Figure 3: Active critique queries in 8×8 gridworlds with 48 features.

Some Experiments-Relative Entropy Inverse Reinforcement Learning

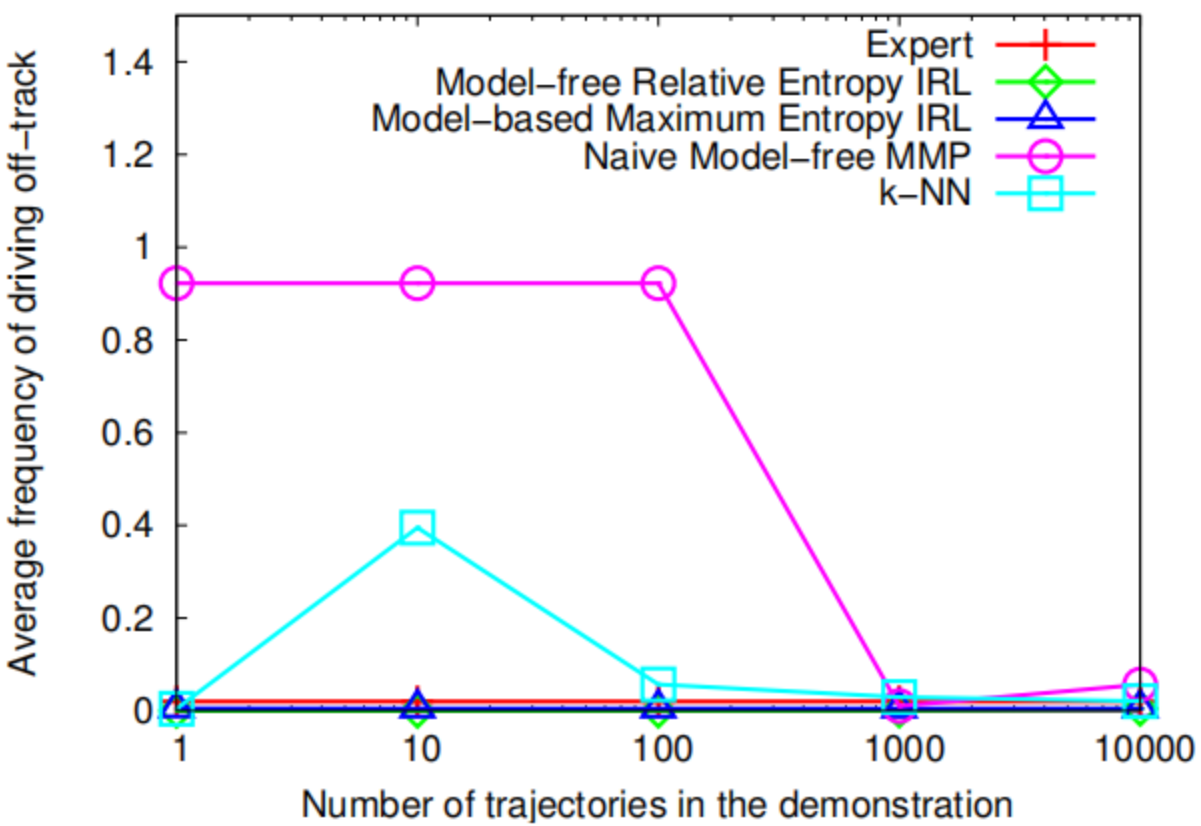


(a) Average reward in the racetrack

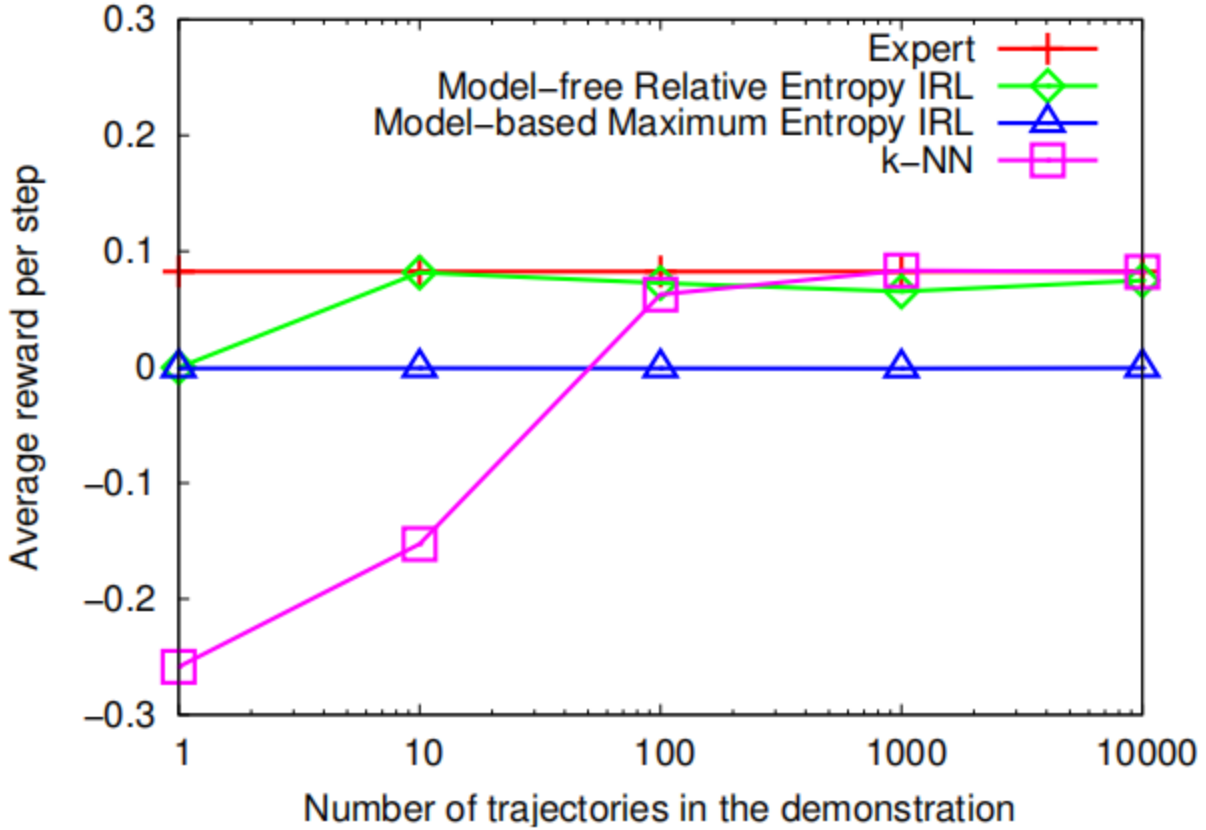


(b) Average number of time-steps per round

Some Experiments-Relative Entropy Inverse Reinforcement Learning



(c) Average frequency of driving off-track



(d) Average reward in the gridworld