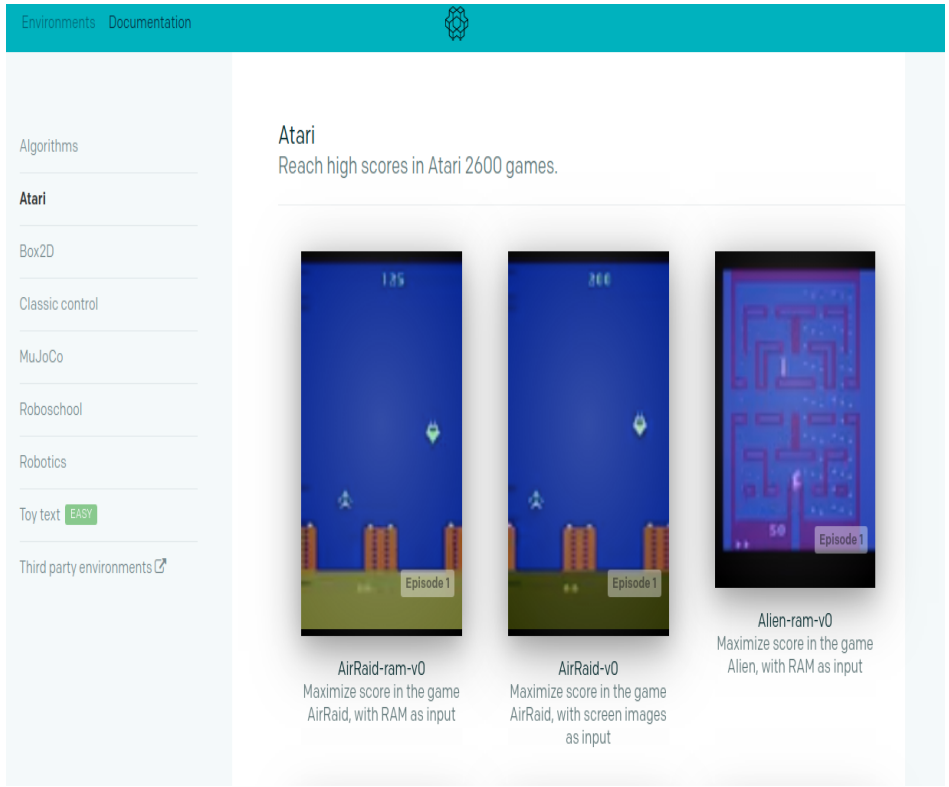
Table of Contents

介绍gym

- https://gym.openai.com/



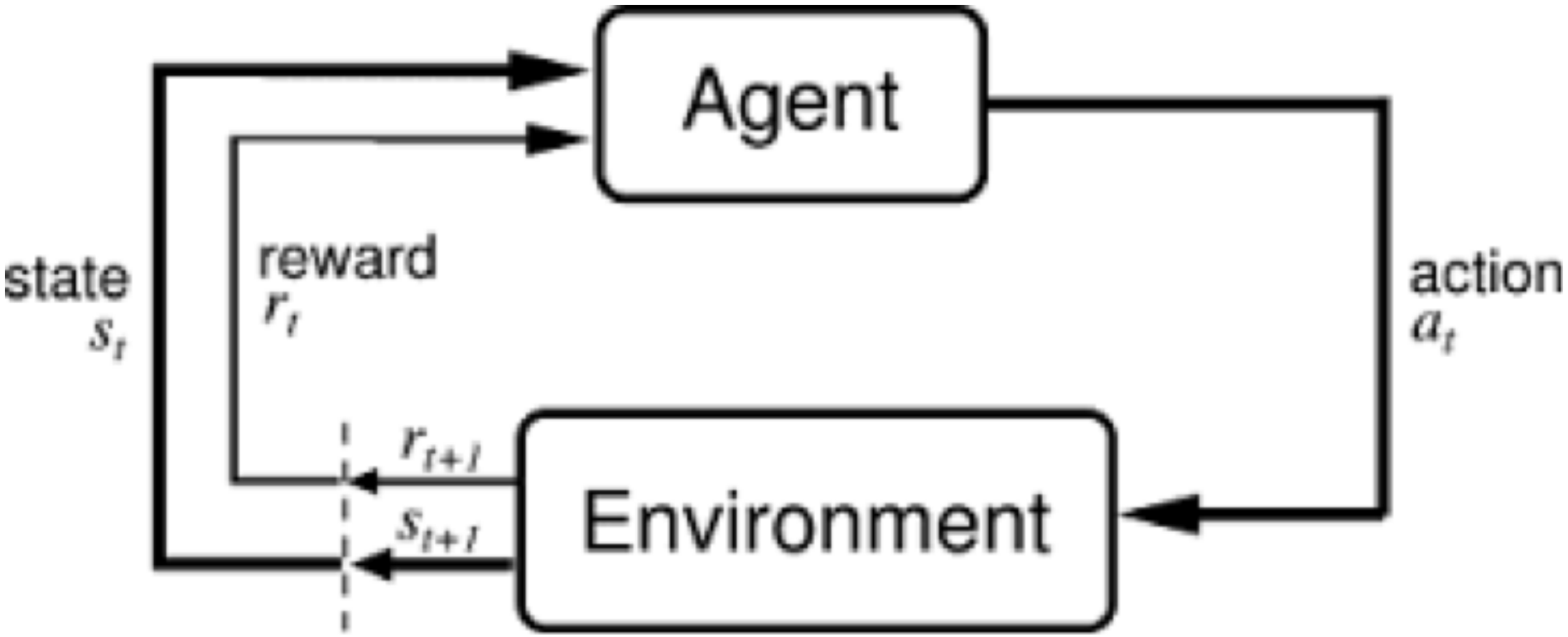
安装gym

- python 3.5.2 安装 gym
  sudo pip3 install gym matplotlib==2.0 pandas==0.23.0
- 其它
  sudo apt-get install python3-tk
- 如果上面报错呢？一般原因是少了一些必要的模块
  sudo apt-get install golang python3-dev python-dev libcupti-dev libjpeg-turbo8-dev make tmux htop chromium-browser git cmake zlib1g-dev libjpeg-dev xvfb libav-tools xorg-dev python-opengl libboost-all-dev libsdl2-dev swig

window10 安装gym

- 下载anaconda (推荐)
- 在base环境中直接pip install gym
  网速不好可以接-i https://pypi.tuna.tsinghua.edu.cn/simple

如何使用gym环境



如何使用gym环境

- 创建一个environment
  - $env = gym.make(game\_name)$
- 初始状态s_0
  - $s_0 = env.reset()$
- agent和environment之间的交互
  - $next\_state, reward, done, \_ = env.step(action)$

如何使用gym环境

- 显示画面
  - *env.render()*
- spaces
  - *env.action_space*
  - *env.observation_space*

Table of Contents

black jack环境

- reinforcement learning,sutton p_{74}
- black jack 也叫21点游戏
  - 目标：游戏者的目标是使手中的牌的点数之和不超过21点且尽量大,本次实验只有你和庄家玩
  - action：叫牌(hit)和停止叫牌(stick)
  - 计算：2至9牌，按其原点数计算；K、Q、J和10牌都算作10点；A 牌既可算作1点也可算作11点

black jack环境

- 浏览环境

```
plf@plf-pc:~/Desktop/exp1/exp1/assignment1/mc$ python3 BlackjackEnv.py
Player Score: 17 (Usable Ace: False), Dealer Score: 1
Taking action: Hit
Player Score: 20 (Usable Ace: False), Dealer Score: 1
Taking action: Stick
Player Score: 20 (Usable Ace: False), Dealer Score: 1
Game end. Reward: 1.0
```

mc算法

- 在mc.py中实现函数mc(env, num_episodes, discount_factor, epsilon)

On-policy first-visit MC control (for ε-soft policies), estimates $\pi \approx \pi_*$

Algorithm parameter: small $\varepsilon > 0$

Initialize:

$\pi \leftarrow$ an arbitrary ε-soft policy

$Q(s,a) \in \mathbb{R}$ (arbitrarily), for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

$Returns(s,a) \leftarrow$ empty list, for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$

Repeat forever (for each episode):

Generate an episode following π: $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:

$G \leftarrow G + R_{t+1}$

Unless the pair S_t, A_t appears in $S_0, A_0, S_1, A_1 \ldots, S_{t-1}, A_{t-1}$:

Append G to $Returns(S_t, A_t)$

$Q(S_t, A_t) \leftarrow \text{average}(Returns(S_t, A_t))$

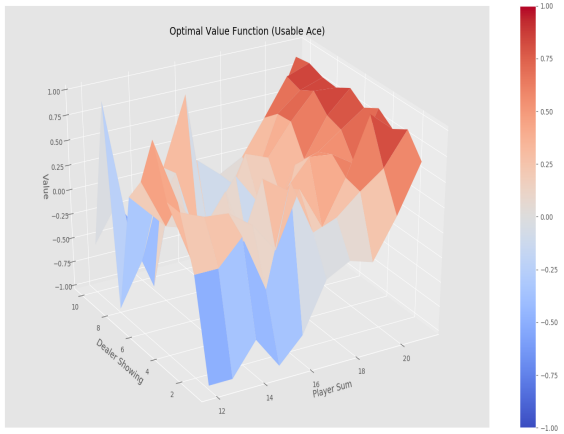
$A^* \leftarrow \arg\max_a Q(S_t, a)$ (with ties broken arbitrarily)

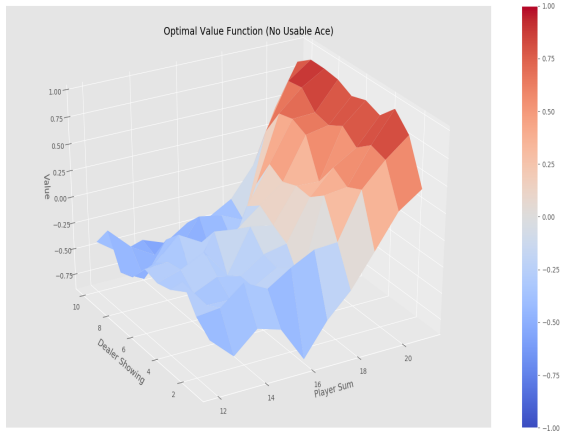
For all $a \in \mathcal{A}(S_t)$:

$$\pi(a|S_t) \leftarrow \begin{cases} 1 - \varepsilon + \varepsilon/|\mathcal{A}(S_t)| & \text{if } a = A^* \\ \varepsilon/|\mathcal{A}(S_t)| & \text{if } a \neq A^* \end{cases}$$

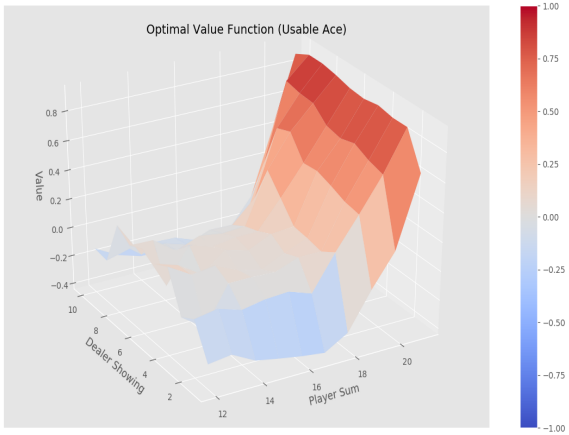
mc算法结果

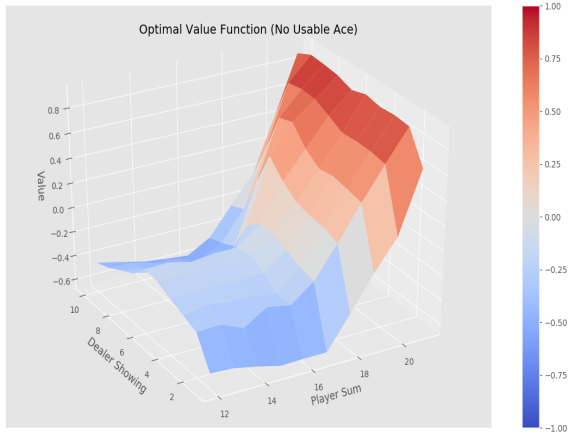
- 10000 episodes





- 500000 episodes



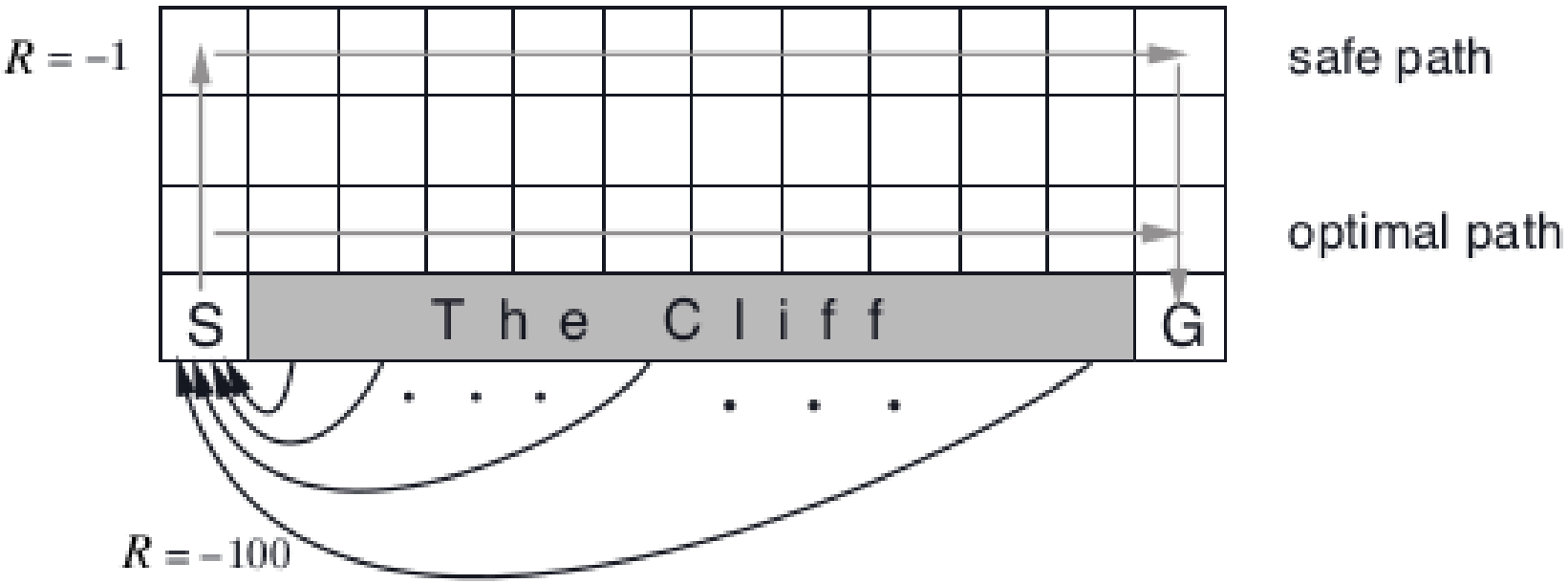


可选做

- 额外加分
  实现every-visit 版本，并且对比 first-visit 和 every-visit

Table of Contents

cliff walk环境



cliff walk环境

- 运行td文件下的cliff_walk.py 浏览环境

```
plf@plf-pc:~/Desktop/exp1/exp1/assignment1/td$ python3 cliff_walk.py
36
o  o  o  o  o  o  o  o  o  o  o  o
o  o  o  o  o  o  o  o  o  o  o  o
o  o  o  o  o  o  o  o  o  o  o  o
x  C  C  C  C  C  C  C  C  C  C  T

(24, -1.0, False, {'prob': 1.0})
o  o  o  o  o  o  o  o  o  o  o  o
o  o  o  o  o  o  o  o  o  o  o  o
x  o  o  o  o  o  o  o  o  o  o  o
o  C  C  C  C  C  C  C  C  C  C  T
```

reward

- reinforcement learning,sutton p_{106}

sarsa算法

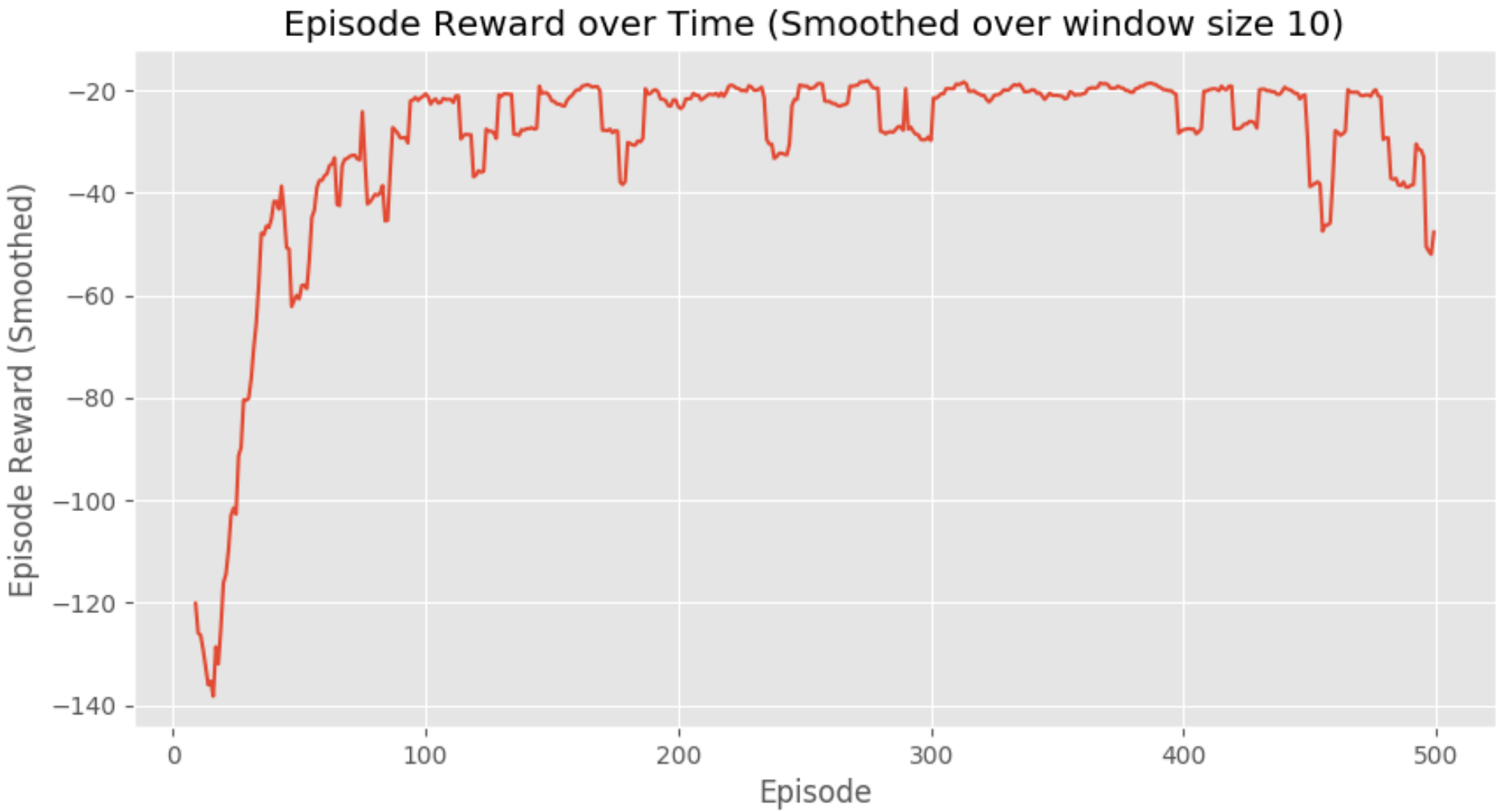
- 在sarsa.py中实现函数sarsa(env, num_episodes, discount_factor, alpha, epsilon)

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize S
  Choose A from S using policy derived from Q (e.g., ϵ-greedy)
  Loop for each step of episode:
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ϵ-greedy)
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma Q(S', A') - Q(S, A)\big]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  until S is terminal

sarsa算法


Episode Reward over Time (Smoothed over window size 10)
Episode Reward (Smoothed)
−20
−40
−60
−80
−100
−120
−140
0
100
200
300
400
500
Episode

q learning算法

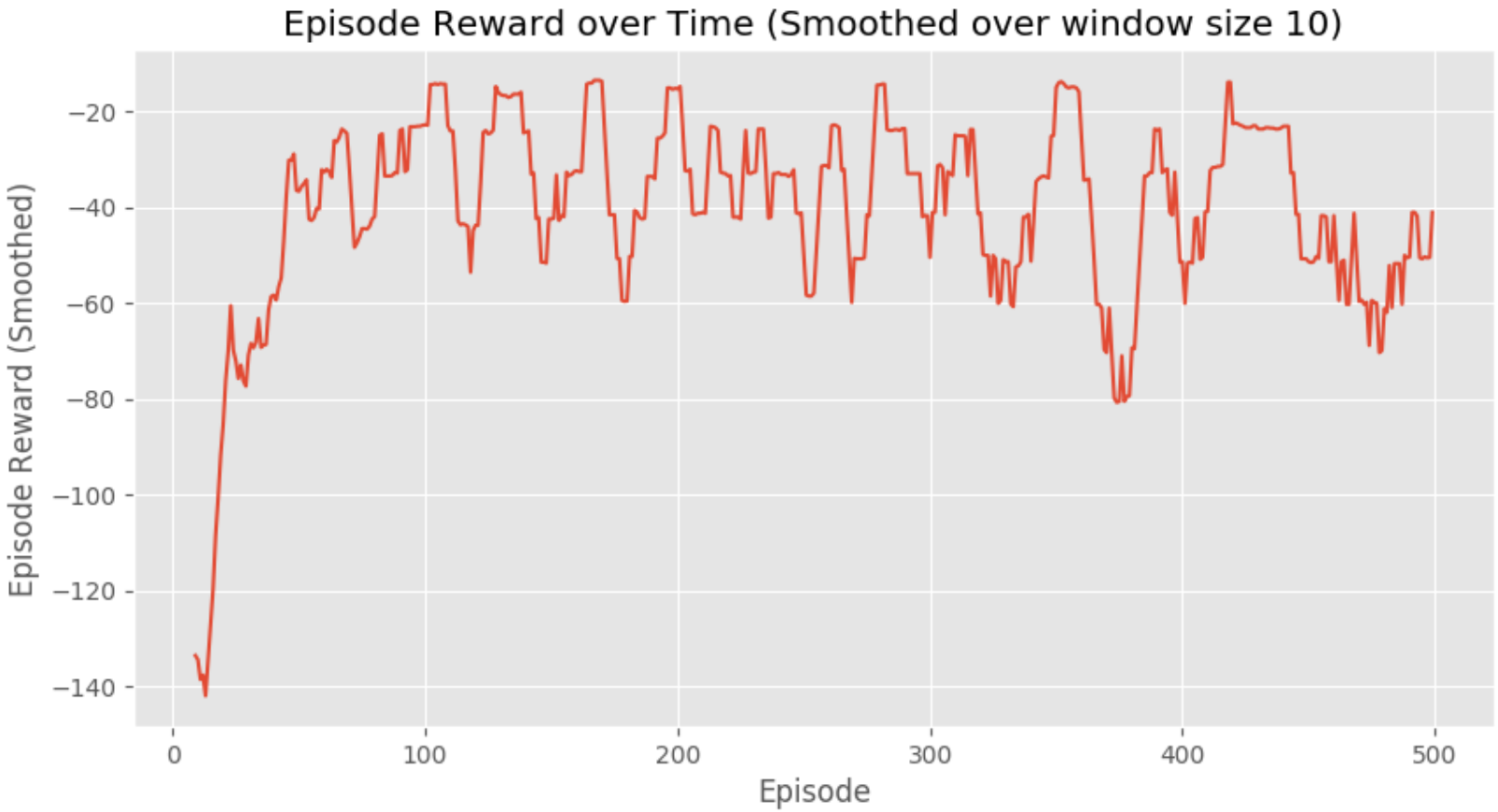
- 在qlearning.py中实现函数q_learning(env, num_episodes, discount_factor, alpha, epsilon)

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
    $S \leftarrow S'$
  until S is terminal

qlearning算法



可选做

- 额外加分
  实现double q-learning,对比double q-learning和q-learning

Double Q-learning, for estimating $Q_1 \approx Q_2 \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q_1(s, a)$ and $Q_2(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, such that $Q(terminal, \cdot) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    Choose A from S using the policy ε-greedy in $Q_1 + Q_2$
    Take action A, observe R, S'
    With 0.5 probabilility:

$$Q_1(S, A) \leftarrow Q_1(S, A) + \alpha\Big(R + \gamma Q_2\big(S', \operatorname{argmax}_a Q_1(S', a)\big) - Q_1(S, A)\Big)$$

    else:

$$Q_2(S, A) \leftarrow Q_2(S, A) + \alpha\Big(R + \gamma Q_1\big(S', \operatorname{argmax}_a Q_2(S', a)\big) - Q_2(S, A)\Big)$$

    $S \leftarrow S'$
  until S is terminal

提交要求

- 邮件标题: 实验一-姓名-学号
- 压缩文件命令格式: 实验一-姓名-学号
  - 源码
  - 报告(pdf格式)
- 截止日期: 2021.11.16
- 提交邮箱地址: ustcrl2021@163.com

联系

- 课程qq群:902870137
- 邮箱:zhsh1@mail.ustc.edu.cn