Q-learning Vs. SARSA applied to SMART CAB

A PROJECT PRESENTATION

Submitted by

MOHAMMAD WASIL SALEEM

MATRIKEL Nr.: 805779

[MAT-DSAM3A] Advanced Data Assimilation and Modeling A

Reinforcement Learning

MOTIVATION

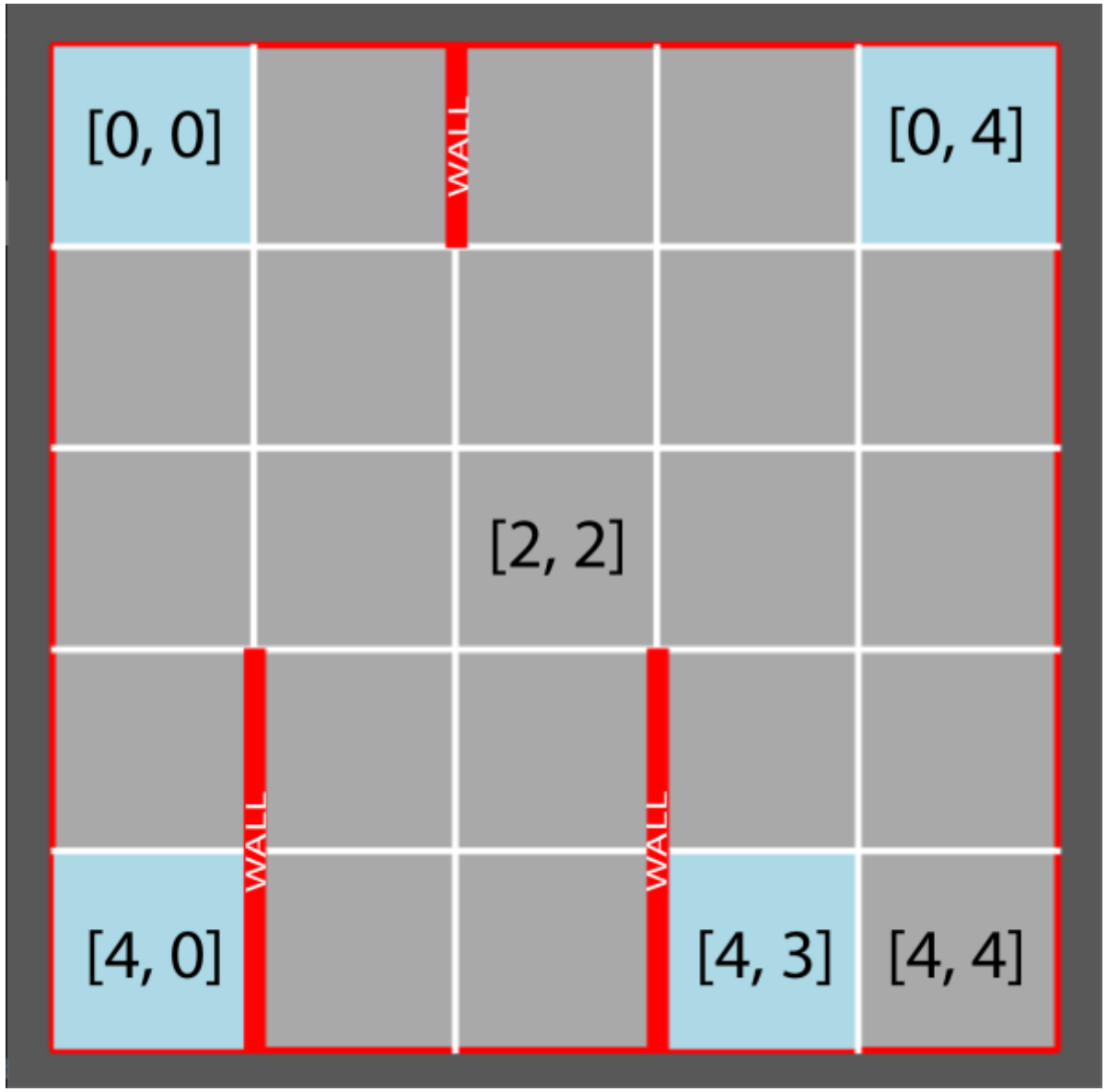
- To study and compare Q-Learning and SARSA algorithm.
- Model free and Model-based RL algorithm.
- Approach - Temporal difference learning - learns how to predict a quantity that depends on future values of a given signal – learns from experience.
- Temporal difference update step:

$$NewEstimate \leftarrow OldEstimate + Stepsize\,[Target - oldEstimate]$$

SMART CAB GAME

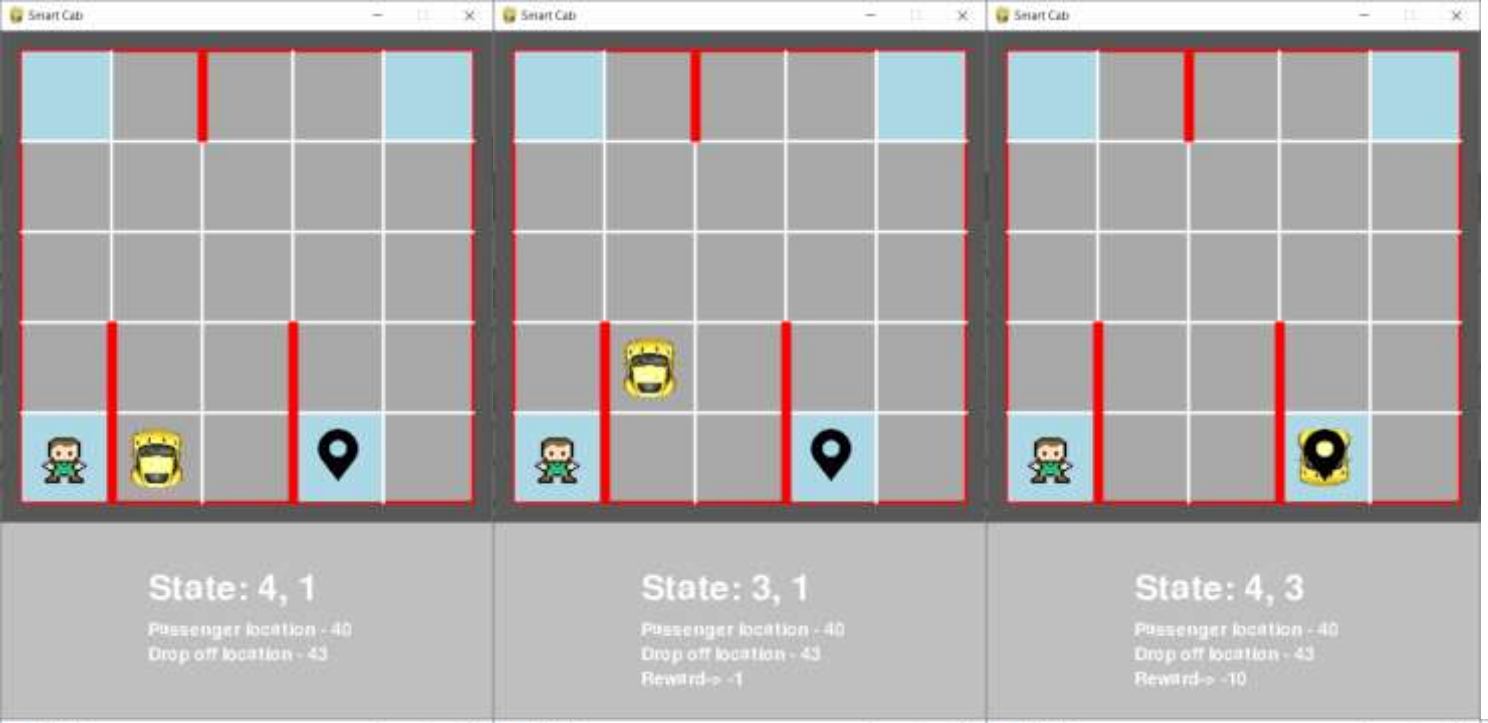
- Inspired from OpenAI gym environment.
- 2D grid 5x5 cells.
- Agent - Cab
- Drop off and pick up locations.
- Objective of the game:
    1. pick up the passenger.
    2. drop off the passenger at the right location.
    3. take as minimum time as possible.
- Coordinate System: See Screenshot.
- Pickup positions: [0, 0], [0, 4], [4, 0] and [4, 3].
- Dropoff positions: [0, 0], [0, 4], [4, 0] and [4, 3].
- Rules:
    1. Drop off location should not be equal to Pickup location in one episode.
    2. Cab cannot go through the walls.
    3. Cab can move “UP”, “DOWN”, “LEFT”, and “RIGHT. No Diagonal movements.
    4. Cannot go beyond the extreme rows and columns.

GRID

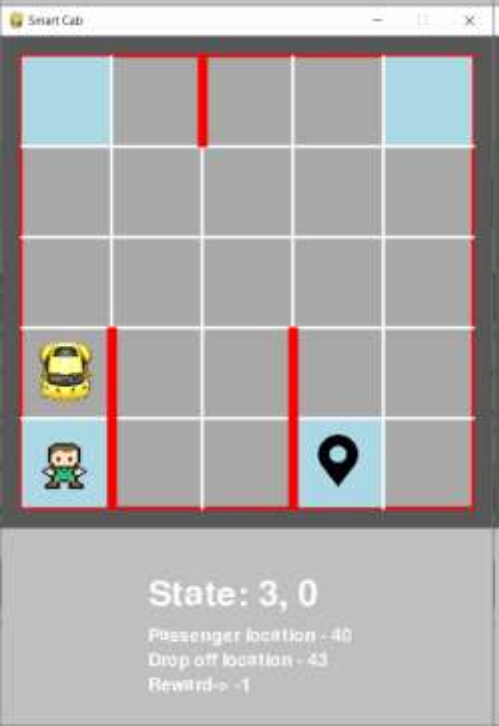


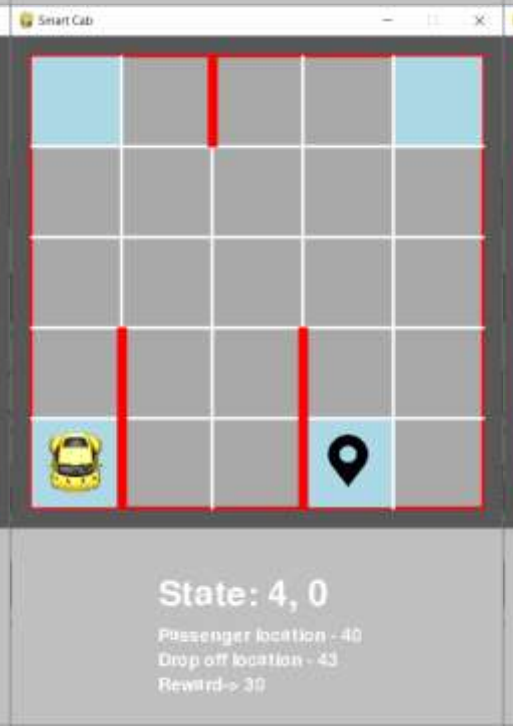
REWARD FUNCTION

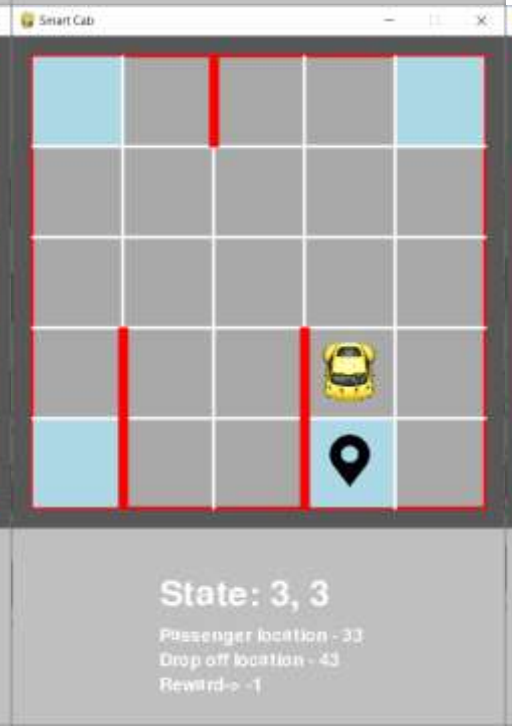
- $Reward(s, a) = \begin{cases} -1 & for\ every\ step \\ -10 & pickup\ from\ wrong\ location \\ -10 & drop\ off\ at\ wrong\ location \\ +30\ or\ 0 & pick\ up\ from\ right\ location \\ +20 & drop\ off\ at\ right\ location \end{cases}$

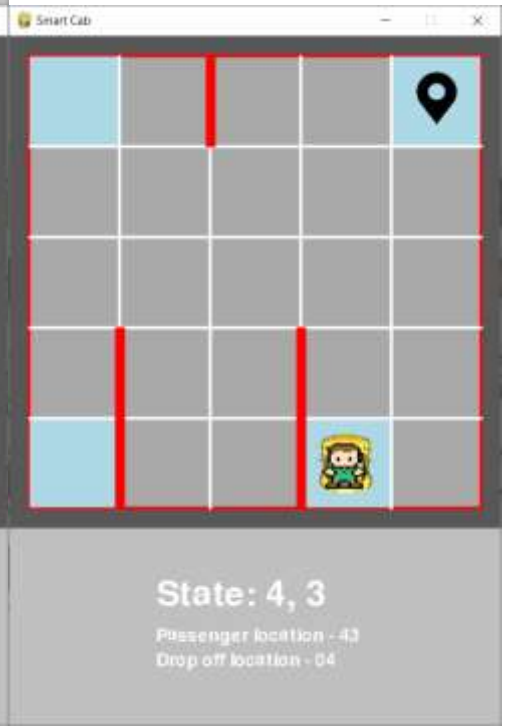


Game work flow…









STATE SPACES

[0, 0] encodes integer ‘0’.
[4, 0] encodes integer ‘1’.
[0, 4] encodes integer ‘2’.
[4, 3] encodes integer ‘3’.

Total number of States = 52 + 336 = **388**

```
'1_2_0_0_1',
'1_2_0_0_2',
'1_2_0_0_3',

'1_2_0_4_0',
'1_2_0_4_1',
'1_2_0_4_3',

'1_2_1_2_0',
'1_2_1_2_1',
'1_2_1_2_2',
'1_2_1_2_3',

'1_2_4_0_0',
'1_2_4_0_2',
'1_2_4_0_3',

'1_2_4_3_0',
'1_2_4_3_1',
'1_2_4_3_2'.
```

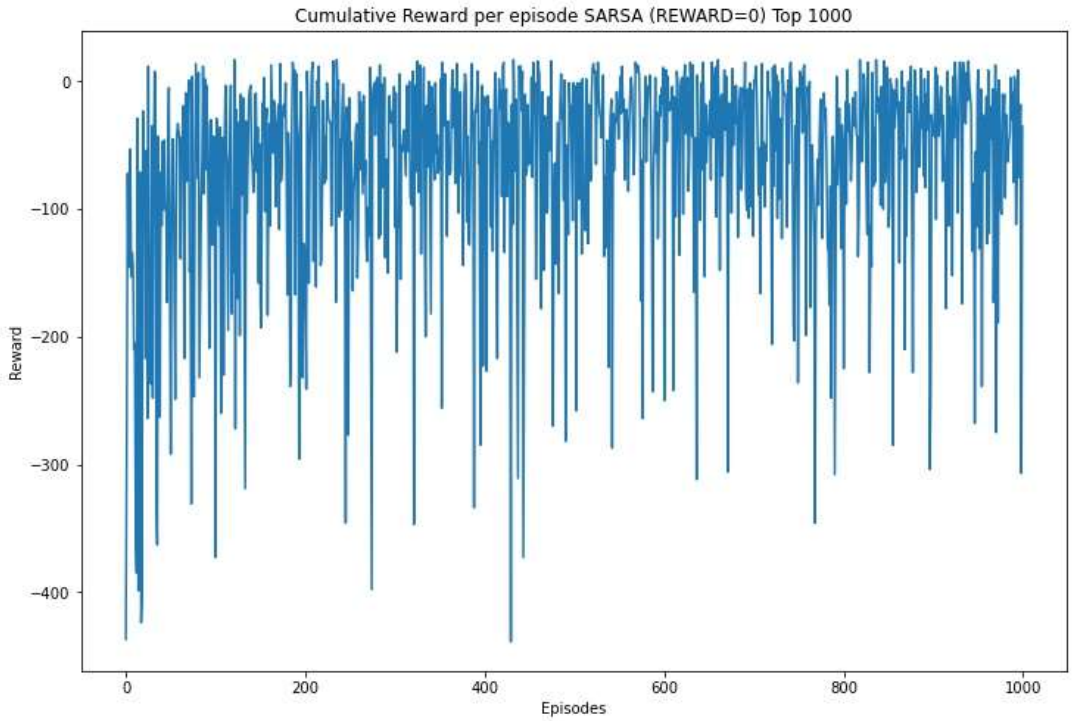
21x16 = 336 States

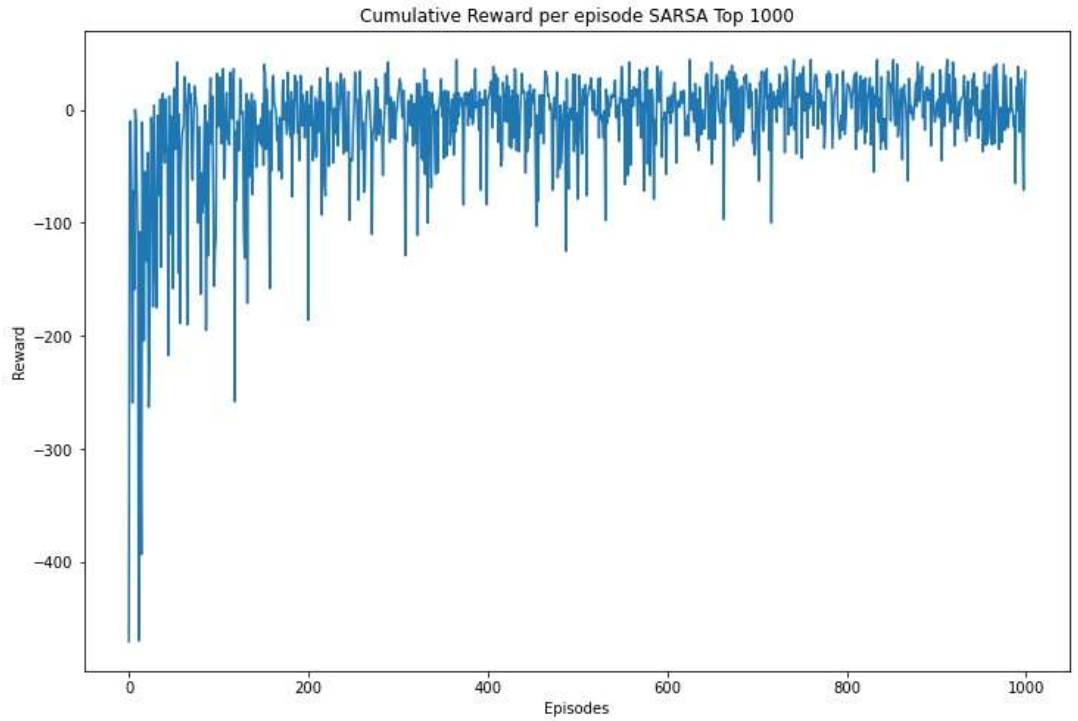
```
'0_0_0_0_0',
'0_0_0_0_1',
'0_0_0_0_2',
'0_0_0_0_3',

'0_0_0_4_0',
'0_0_0_4_1',
'0_0_0_4_3',

'0_0_4_0_0',
'0_0_4_0_2',
'0_0_4_0_3',

'0_0_4_3_0',
'0_0_4_3_1',
'0_0_4_3_2'.
```

13x4 = 52 States

SARSA

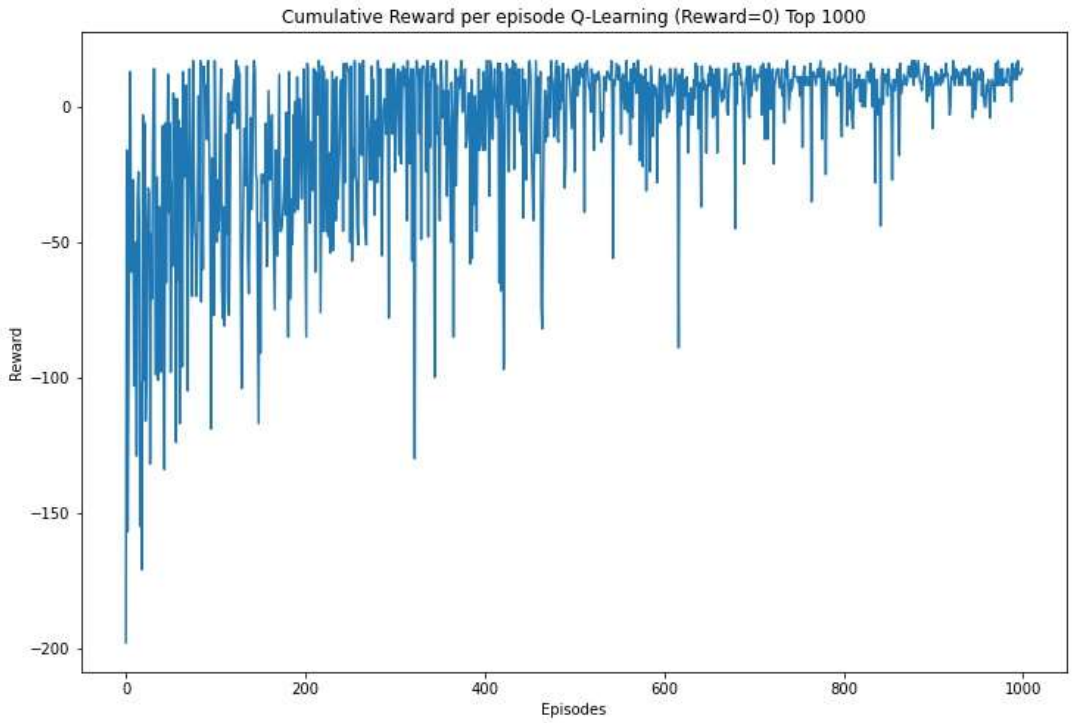
- On-policy learning.
- $\hat{Q}(s,a) \leftarrow \hat{Q}(s,a) + \alpha \left(R + \gamma \hat{Q}(\acute{s},\acute{a}) - \hat{Q}(s,a)\right)$
- Learning rate, $\alpha = 0.1$
- Discount factor, $\gamma = 1$
- ε-greedy algorithm, $\varepsilon = 0.4$ (More chances for exploitation than exploration).
- Balances exploitation and exploration.
- Tries to go to each states.
- Trained for 500, 000 episodes.
- Total average cumulative reward, -25.12, with reward = 0 for picking up from the right location.
- Total average cumulative reward, 5.72, with reward = +30 for picking up from the right location.

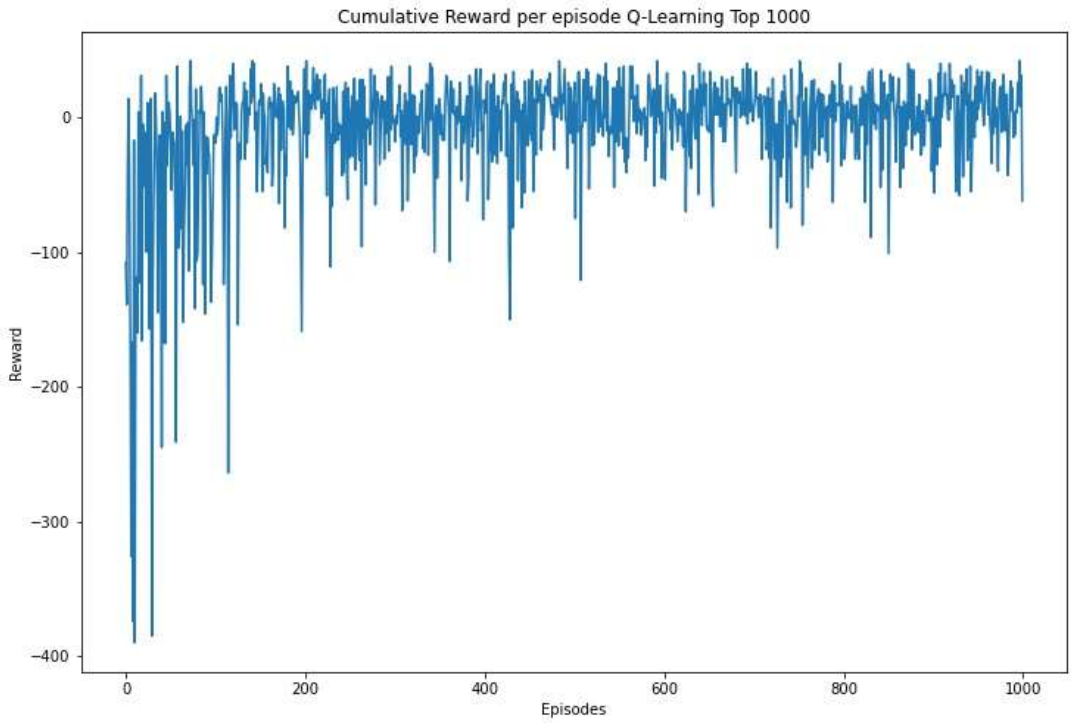

Cumulative Reward per episode SARSA (REWARD=0) Top 1000
Reward
Episodes
0
−100
−200
−300
−400
0
200
400
600
800
1000


Cumulative Reward per episode SARSA Top 1000
Reward
Episodes
0
−100
−200
−300
−400
0
200
400
600
800
1000

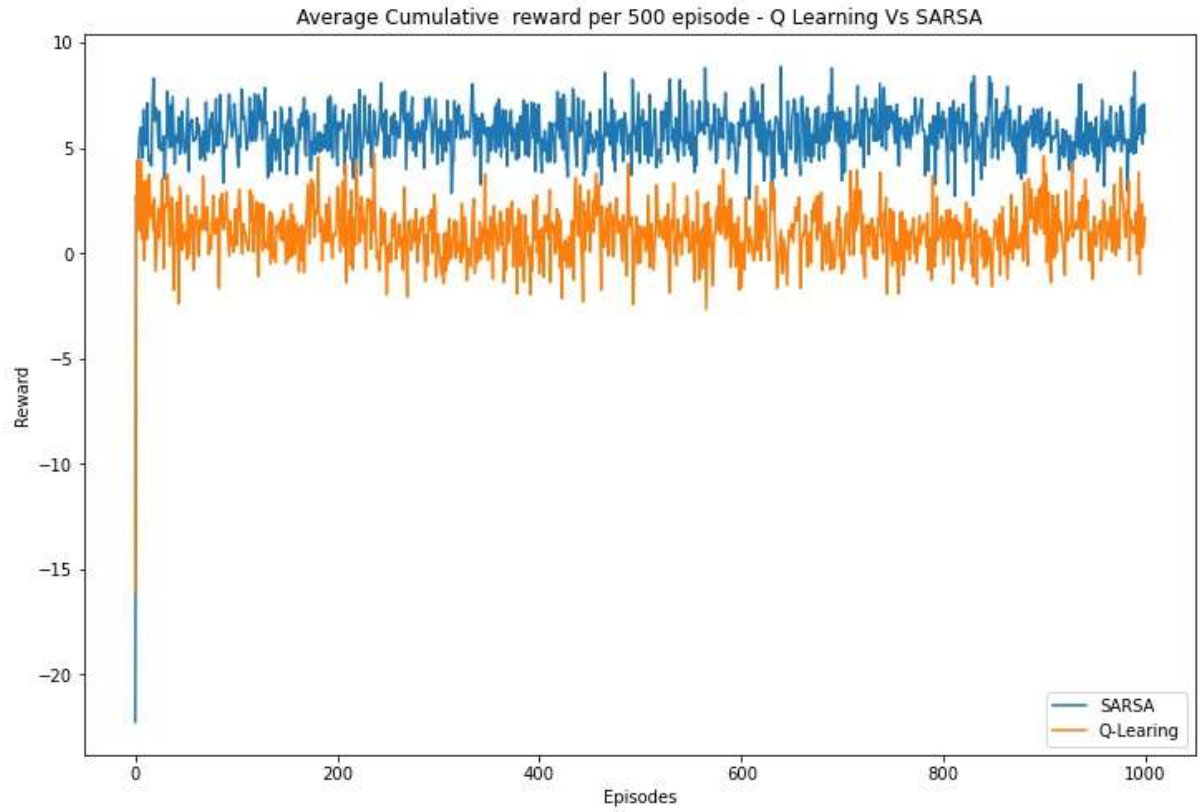
Q-LEARNING

- Off-policy learning.
- $\hat{Q}(s,a) \leftarrow \hat{Q}(s,a) + \alpha \left(R + \gamma \, max_{\acute{a}} \, \hat{Q}(\acute{s},\acute{a}) - \hat{Q}(s,a)\right)$
- Learning rate, $\alpha = 0.1$
- Discount factor, $\gamma = 1$
- ~~ε-greedy algorithm, $\varepsilon = 0.4$~~
- Trained for 500, 000 episodes.
- Total average cumulative reward **10.92** , with reward = 0 for picking up from the right location.
- Total average cumulative reward **0.9812**, with reward = 30 for picking up from the right location.


Cumulative Reward per episode Q-Learning (Reward=0) Top 1000
Reward
Episodes
0
−50
−100
−150
−200
0
200
400
600
800
1000


Cumulative Reward per episode Q-Learning Top 1000
Reward
Episodes
0
−100
−200
−300
−400
0
200
400
600
800
1000

SARSA Vs. Q-LEARNING



CONCLUSION

- Both are excellent approach for RL problems.
- Q-Learning learns optimal policy.
- SARSA learns “near” optimal policy.

REFERENCES

1. https://www.101computing.net/getting-started-with-pygame/
2. http://www.pygame.org/wiki/RotateCenter?parent=CookBook
3. https://gym.openai.com/envs/Taxi-v2/
4. https://www.learndatasci.com/tutorials/reinforcement-q-learning-scratch-python-openai-gym/
5. https://github.com/openai/gym/blob/master/gym/envs/toy_text/taxi.py
6. aakash94.github.io/Reward-Based-Epsilon-Decay/
7. https://stackoverflow.com/questions/53198503/epsilon-and-learning-rate-decay-in-epsilon-greedy-q-learning
8. https://wiki.pathmind.com/deep-reinforcement-learning
9. https://en.wikipedia.org/wiki/Model-free_(reinforcement_learning)
10. https://towardsdatascience.com/reinforcement-learning-temporal-difference-sarsa-q-learning-expected-sarsa-on-python-9fecfda7467e
11. https://medium.com/@violante.andre/simple-reinforcement-learning-temporal-difference-learning-e883ea0d65b0
12. https://towardsdatascience.com/reinforcement-learning-temporal-difference-sarsa-q-learning-expected-sarsa-on-python-9fecfda7467e
13. https://stats.stackexchange.com/questions/184657/what-is-the-difference-between-off-policy-and-on-policy-learning/376830#376830?newreg=703c24a8ebae4e75873f87dbd271717f
14. https://www.geeksforgeeks.org/epsilon-greedy-algorithm-in-reinforcement-learning/
15. https://www.cse.unsw.edu.au/~cs9417ml/RL1/algorithms.html
16. https://towardsdatascience.com/intro-to-reinforcement-learning-temporal-difference-learning-sarsa-vs-q-learning-8b4184bb4978
17. https://datascience.stackexchange.com/questions/9832/what-is-the-q-function-and-what-is-the-v-function-in-reinforcement-learning#:~:text=Q%CF%80(s%2Ca)%20is%20the%20action%2Dvalue,policy%20%CF%80%2C%20taking%20action%20a.

THANK YOU!