ROB 537 Homework 3: Reinforcement Learning

Nathan Butler | butlnath@oregonstate.edu

# Overview

The objective of this assignment was to train an agent to explore a 5x10 gridworld with the objective of finding a door. The environment includes:

* a door initialized at coordinates (9,1) with a reward of 20;
* a solid wall (gray) the agent cannot move across;
* a reward of -1 for every time the agent is in a state other than the red door state.

The agent starts at a random location and has four actions (move in four directions). The state of the system is the location of the agent (x,y), and an episode is 20 time steps.

Unless otherwise specified, the models trained in this assignment were trained over 200 epochs with a step size of 0.85, a discount factor of 0.95, and an epsilon value of 0.9 per the training setup discussed in [1]. Throughout testing the epsilon value was modified to examine the results of emphasizing exploration vs exploitation.

# Academic Statement

Attach your code as an appendix and write a statement at the beginning of the assignment asserting that all work is your own work (including code (minus the code we provide)). Adding to this, cite any outside information you reference while working on this assignment and the people you work with.

Noah Boehme

# SARSA Performance

*Implement SARSA to solve this problem. How did the algorithms perform? Include learning curves and plots of the learned value tables.*

Figure 1 displays the learning curves and value tables gained by varying SARSA training configurations. Learning success metrics were calculated every 5 epochs by conducting 20 tests with the agent initialized in random positions. The average final reward of these tests is plotted in Figure 1. During training, an epsilon-greedy approach was taken to encourage exploration [1]. However, during testing epsilon was disabled so that the algorithm always selected the action with the highest reward. The percentage of successful results gained during testing is also displayed in Figure 1.

The performance of the SARSA learning approach relied largely upon the balance of exploration and exploitation captured by the epsilon parameter. A larger epsilon encouraged the model to explore less optimal actions, but as a result it seemed unable to capture meaningful reward values. Lower epsilons produced behaviors in which the model prioritized higher-reward actions more frequently during training.

A graph and a chart

Description automatically generated

(a)

A screenshot of a graph

Description automatically generated

(b)

Figure 1. Learning curves and value tables for SARSA models with (a) representative example of epsilon = 0.9 and (b) representative example of epsilon = 0.1

As observed in Figure 1, prioritizing exploitation during training yielded a value table capable of producing correct solutions 100.0% of the time, as compared to the higher epsilon values that yielded correct solutions only around 20% of the time.

# Q-Learning Performance

*Implement a Q-learning algorithm to solve this problem. How did the algorithms perform? How did solution compare to the SARSA solution? Discuss the implications of your results.*

Figure 1 displays the learning curves and value tables gained by varying Q-learning training configurations, generated using the same process as SARSA. Overall, the trends observed were …

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Figure 2. Learning curves and value tables for Q-learning methods

The results for Q-learning indicate much improved performance over the SARSA algorithm. This is because …

# Moving Door Evaluation

*3 - Now consider the environment where the red door moves randomly by 1 cell every time step. Keep the initial starting location of the door the same as before. Use the EXACT same algorithms from problems 1 and 2 to solve this problem. How does the performance of the agent compare to problems 1 and 2? Does the agent learn a good policy? Describe your results and hypothesize why your agent performs the way it does. Speculate on how you may improve the performance of the agent. Again, plot learning curves and value tables.*

A screenshot of a graph

Description automatically generated

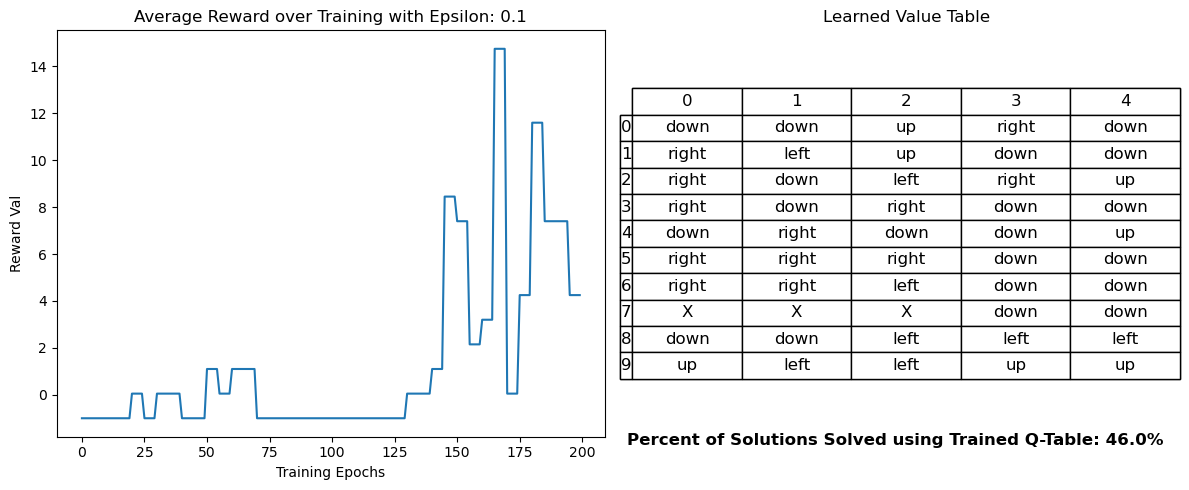


Figure 3. SARSA

A graph and a chart

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

Figure 4. Q-learning

# References

[1] Geeks for geeks

# Appendix

Link to github code: <INSERT LINK>