Project 4

A. Burak Gulhan

April 13, 2023

Contents

1 Baseline Investigation **2**
- 1.1 Summary . . . 2
- 1.2 Results . . . 2
  - 1.2.1 Classic Q-learning . . . 2
  - 1.2.2 Deep Q-learning . . . 2
- 1.3 Observations and Reasoning . . . 3

2 Exploring Regularity in RL-based Learning **9**
- 2.1 New SOTA Method for Maze Solving . . . 9
- 2.2 Results on new mazes . . . 10
  - 2.2.1 Classic Q-learning . . . 10
  - 2.2.2 Deep Q-learning . . . 10
- 2.3 Observations and Reasoning . . . 10
- 2.4 Comparisons . . . 19

1 Baseline Investigation

1.1 Summary

Q-learning is a reinforcement learning method that trains a Q-function, Q(s, a) which represents the expected cumulative reward obtained by taking action a in state s, by finding a policy for each state and action pair. The Q-function has a Q-table where each cell corresponds to a state and action pair value. In our maze algorithm we have only 3 states, move North, South, East or West. This table in randomly initialized and is updated as the agent explores the environment.

The Q-learning algorithm updates each state-action pair in the Q-table, which is visited during the agent's interacting with the environment (ie maze). The agent uses an exploration and exploitation strategies to select actions during this process. Meaning that the agent sometimes chooses actions randomly (exploration) in order to discover new states and actions, and sometimes the agent chooses actions based on the current estimation of the Q-function (exploitation).

Deep Q-learning, replaces the Q-table with a neural network that learns and predicts values for each action at a state. Deep Q-learning may be used in environments where the action/state combinations are very large, making classical Q-learning infeasible to use.

1.2 Results

Due to having many different types of mazes (both random and sample) I did not put the all of the results into the report. I put the ones that I thought were representative of Q-learning behavior.

1.2.1 Classic Q-learning

Figure 1 shows the steps per episode and rewards per episode figures of a random 3x3 maze.

Figure 2 shows the steps per episode and rewards per episode figures of the sample 5x5 maze.

Figure 3 shows the steps per episode and rewards per episode figures of the sample 10x10 maze.

Figure 4 shows the steps per episode and rewards per episode figures of a random 10x10 maze with portals.

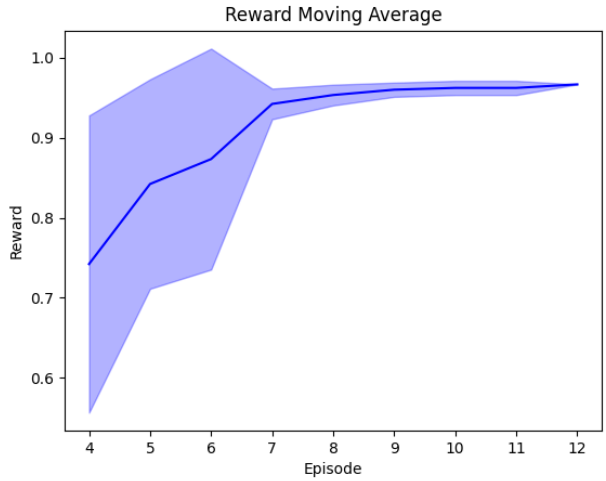
Figure 5 shows the steps per episode and rewards per episode figures of the sample 100x100 maze.

1.2.2 Deep Q-learning

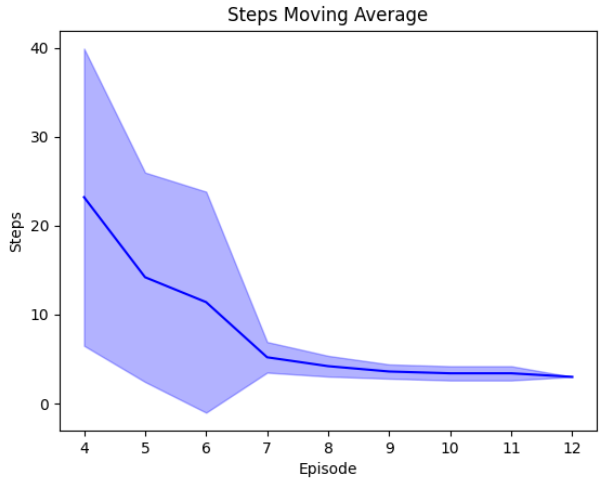
Figures 6 and 7 show the maze layout, visitation frequency, steps per episode, and rewards per episode figures of the sample 3x3 maze.

Figures 8 and 9 show the maze layout, visitation frequency, steps per episode, and rewards per episode figures of the sample 5x5 maze.

Figures 10 and 11 show the maze layout, visitation frequency, steps per episode, and rewards per episode figures of the sample 10x10 maze.

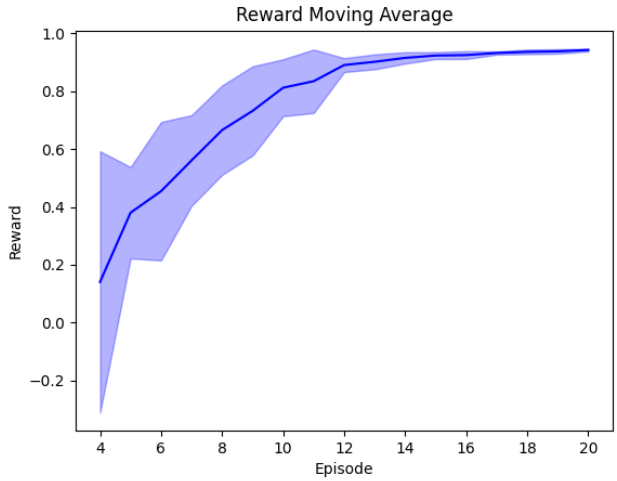


(a) Rewards per episode

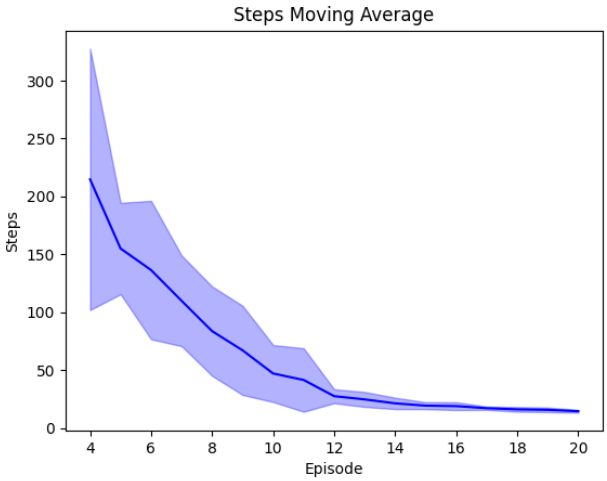


(b) Steps per episode

Figure 1: Q-learning random 3x3 maze



(a) Rewards per episode



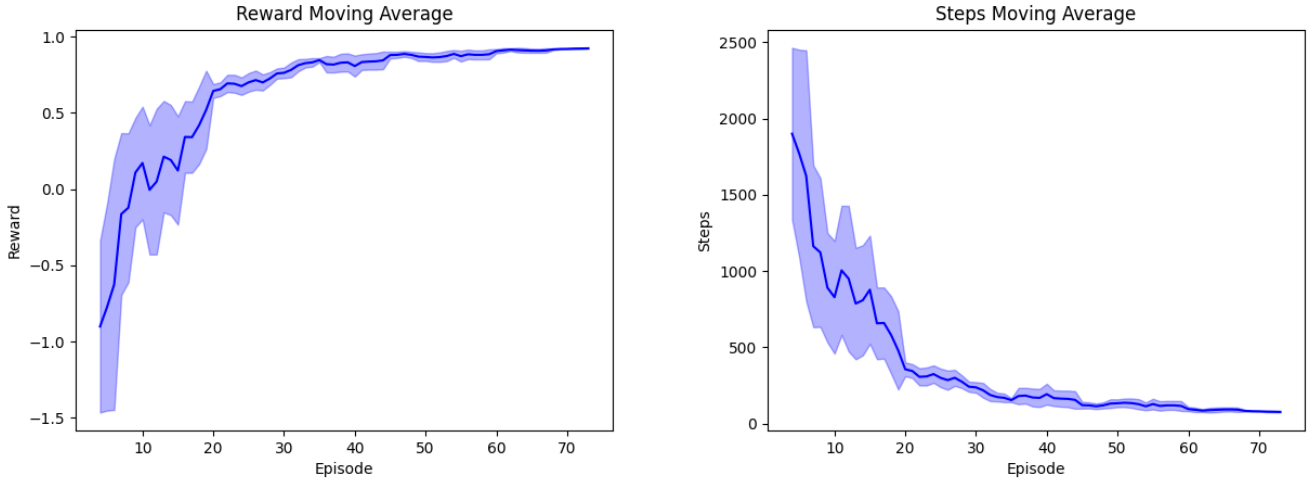
(b) Steps per episode

Figure 2: Q-learning sample 5x5 maze

1.3 Observations and Reasoning

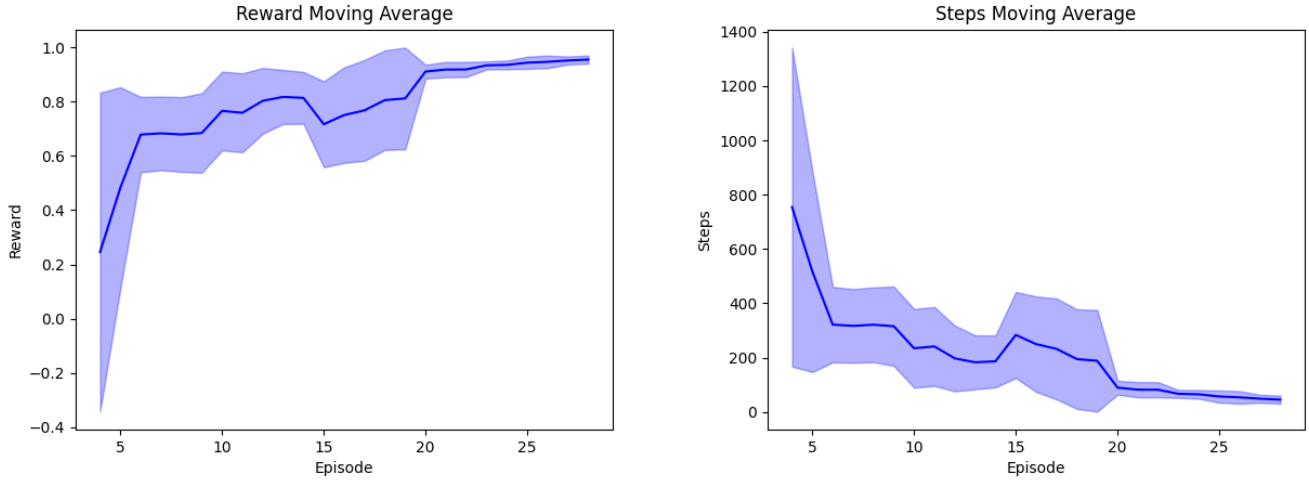
Deep Q-learning takes longer to solve the mazes, both in terms of the number of episodes and in real-time (note that I used CPU for both classic Q-learning and Deep Q-learning).

In the classical Q-learning results we can see that the number of episodes increases as the maze size increases. It is interesting that the plus mazes (that include portals) take less time to solve than an equivalently sized non-plus maze. This may be due to portals



(a) Rewards per episode (b) Steps per episode

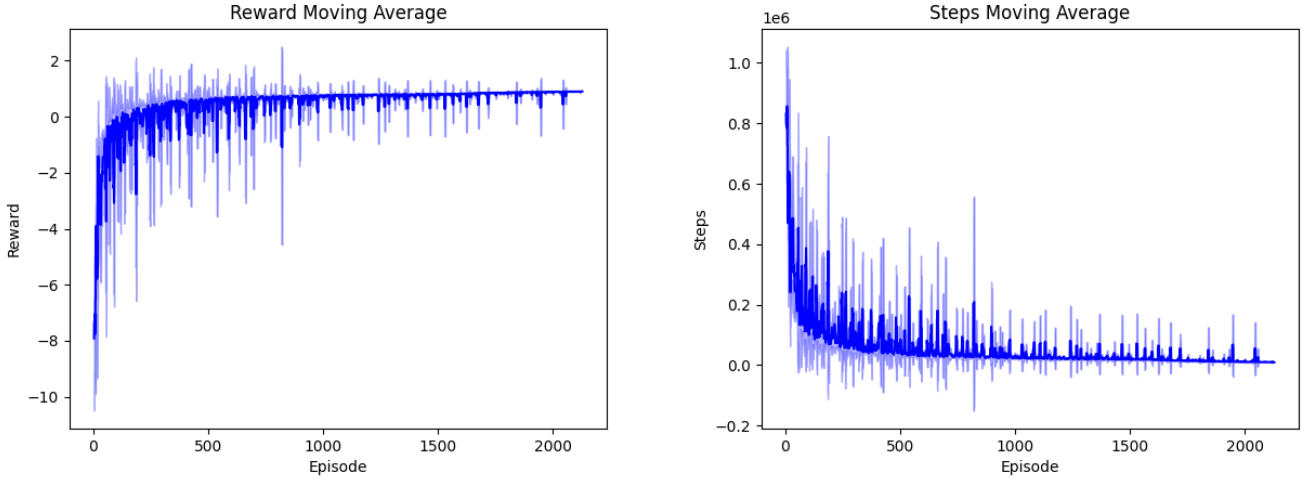
Figure 3: Q-learning sample 10x10 maze



(a) Rewards per episode (b) Steps per episode

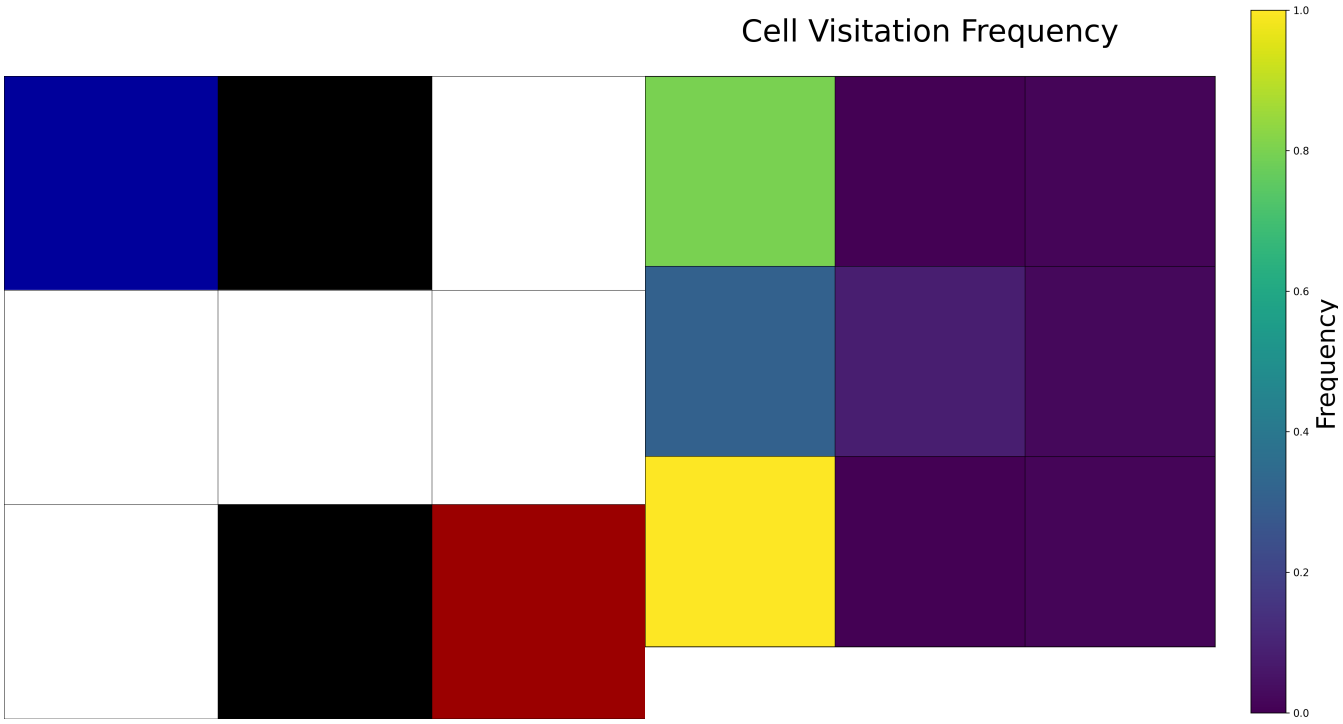
Figure 4: Q-learning random 10x10 maze with portals

0=



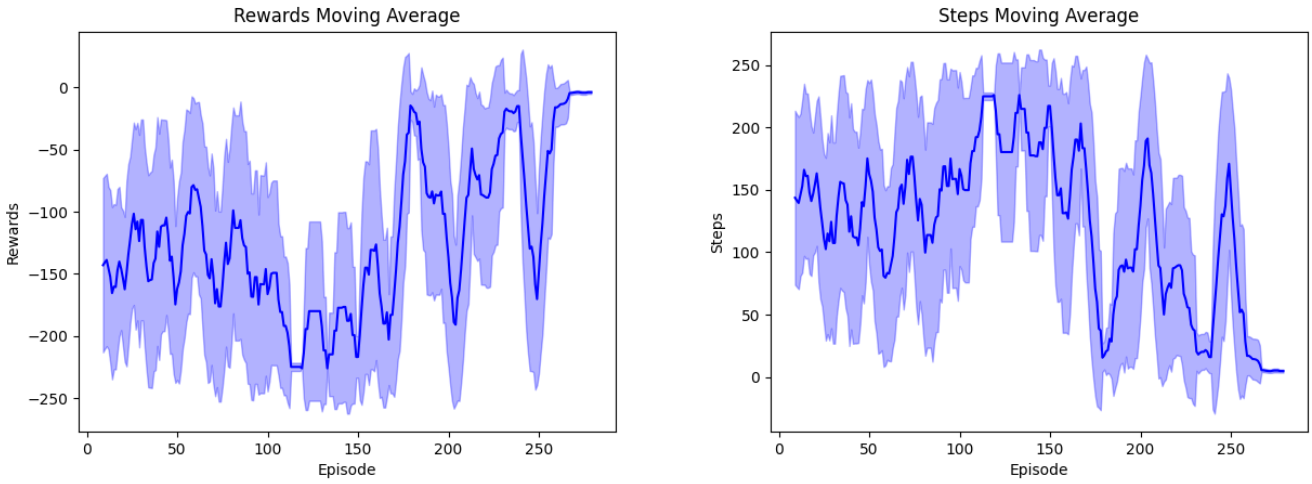
(a) Rewards per episode (b) Steps per episode

Figure 5: Q-learning sample 100x100 maze



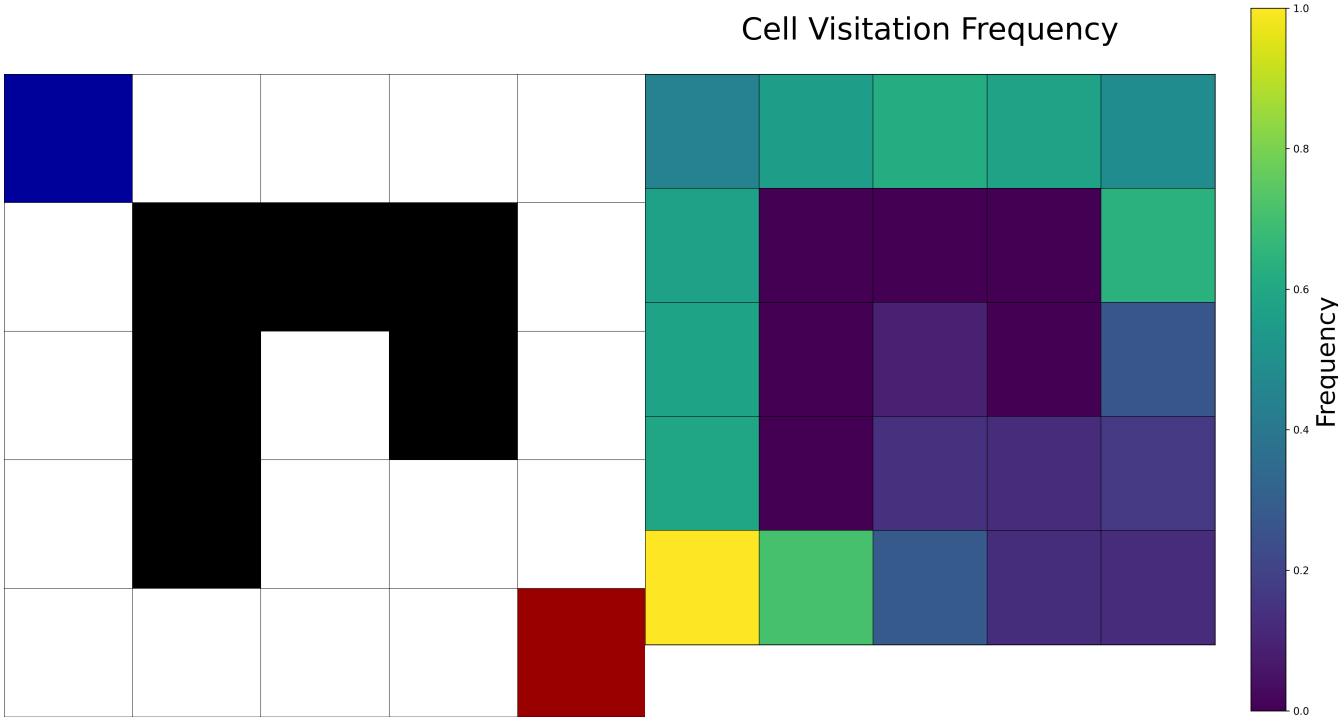
(a) Maze Layout (b) Visitation Frequency

Figure 6: Deep Q-learning 3x3 maze



(a) Rewards per episode (b) Steps per episode

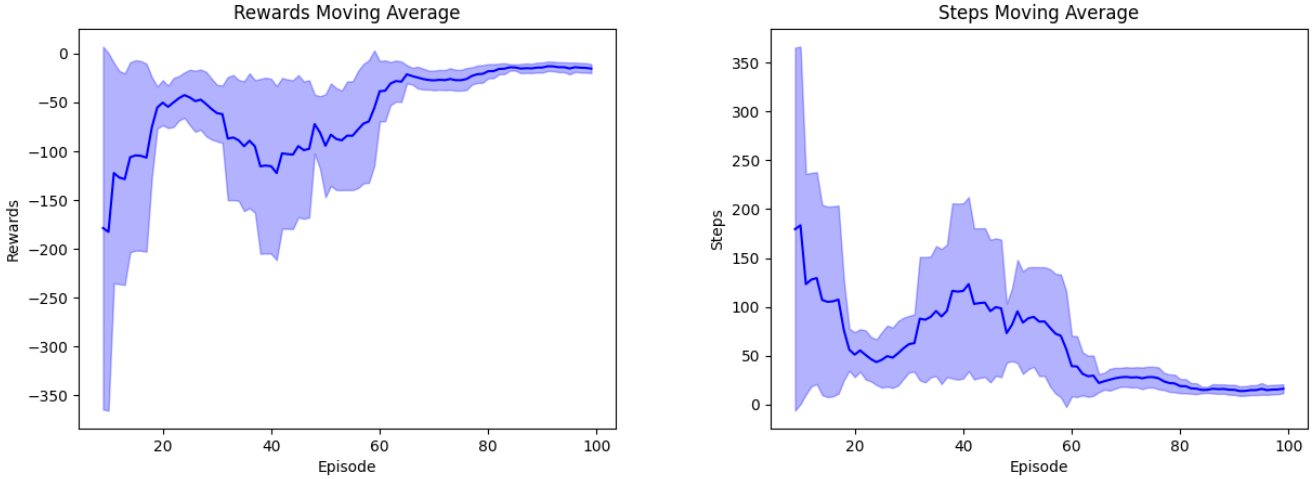
Figure 7: Deep Q-learning 3x3 maze



(a) Maze Layout (b) Visitation Frequency

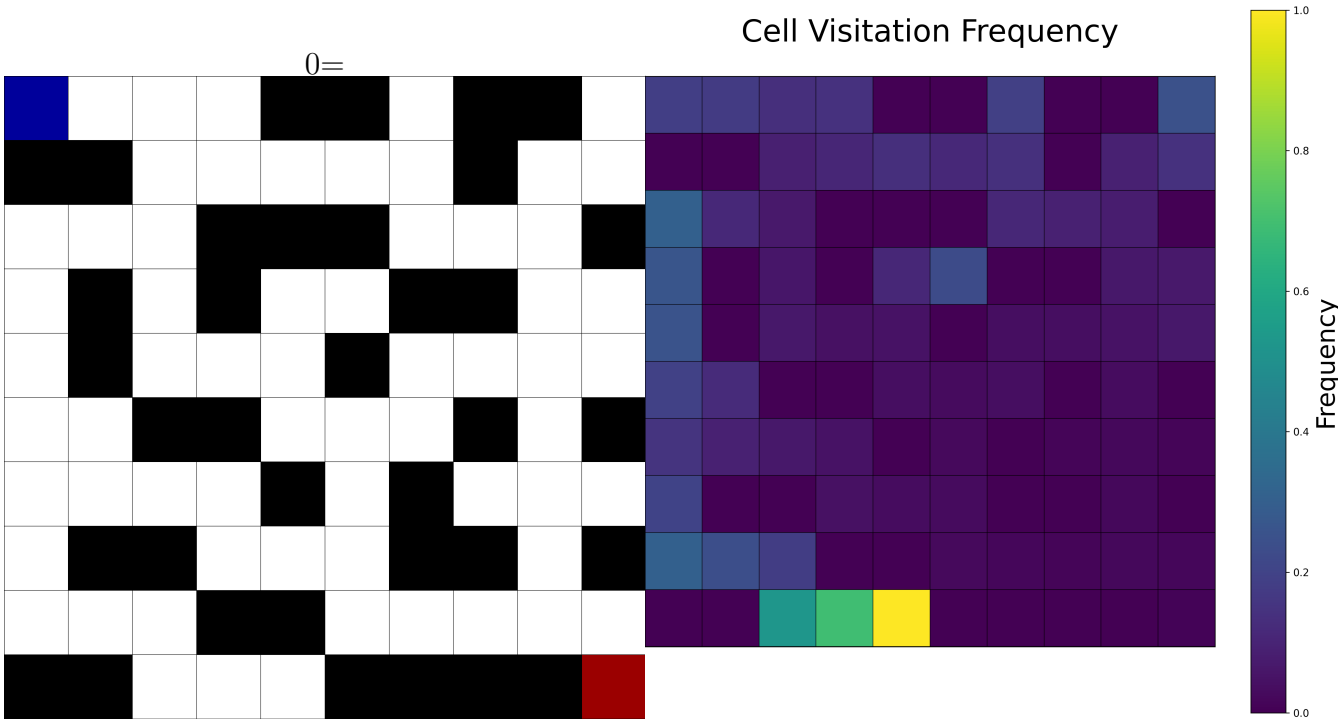
Figure 8: Deep Q-learning 5x5 maze

0=



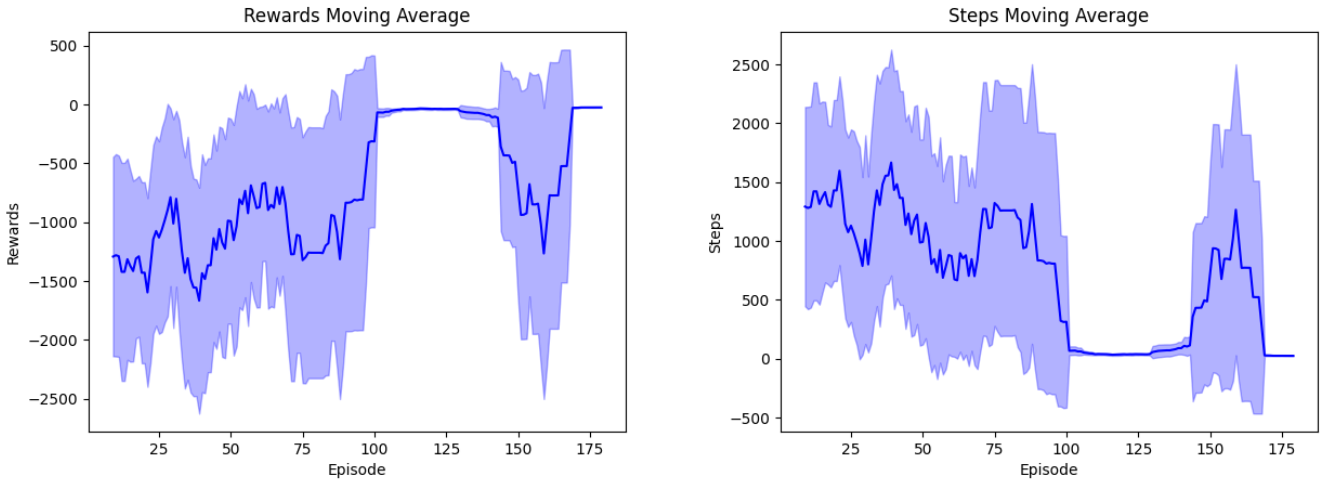
(a) Rewards per episode (b) Steps per episode

Figure 9: Deep Q-learning 5x5 maze



(a) Maze Layout (b) Visitation Frequency

Figure 10: Deep Q-learning 10x10 maze



(a) Rewards per episode

(b) Steps per episode

Figure 11: Deep Q-learning 10x10 maze

allowing finding a shorter solution quicker, that is finding a local optimum. However, these are probably not the global optimal solutions.

An interesting observation is that deep Q-learning seems to get stuck sometimes, even for very simple mazes. For example the 3x3 maze 6 took over 250 Episodes 7 to solve in deep Q-learning but only took 12 episodes in classic Q-learning 1. Looking at the visitation frequency graph of deep Q-learning 6b we can see that the model gets stuck on the left side of the maze.

Another interesting observation is that in Deep Q-learning the steps and rewards graphs are very noisy compared to classic Q-learning. With classic Q-learning, we can easily identify that the curves are asymptotic (especially in the 100x100 maze 5), but this is not very clear in deep Q-learning figures.

Lastly, in deep Q-learning, the model converging does not mean that the optimal was found. For example, in the 10x10 maze 10, we can see both in the steps and rewards graphs 11 that around 100 to 125 episodes, there is a local minimum solution, however, this is one step off from the optimal. After around 70 iterations the deep Q-learning model manages to exit this local optimum. This may be due to either the drop rate or the random factor hyperparameter. Note that I changed the default solved threshold value in config.cfg, since with the 10x10 maze, the default threshold value was never reached and thus continued running for 5000 episodes. I set the threshold to 1 more than the optimal solution. If I had not set this value to a tight bound, then the model would have stopped running at the local optimal solution.

I do not know if the classic Q-learning model can exit local minima. The given Q-learning code does not have a threshold parameter like in deep Q-learning, it instead returns after the same solution was found consecutively 10 times. So I was not able to test this.

2 Exploring Regularity in RL-based Learning

In this section I test 2 types of mazes with regularity. One is a maze that traverses entire rows at a time such as in Fig. 22a. The second type of maze is the Hilbert space filling curve [1] maze – an example is shown in Fig. 12. The Hilbert curve is a recursive pattern, and depending on the size of the maze, the level of recursion differs (for example in a 15x15 maze, the level of recursion is 3). If the deep Q-learning model achieves good results with Hilbert curves, then this would indicate that the model can learn recursive patterns.

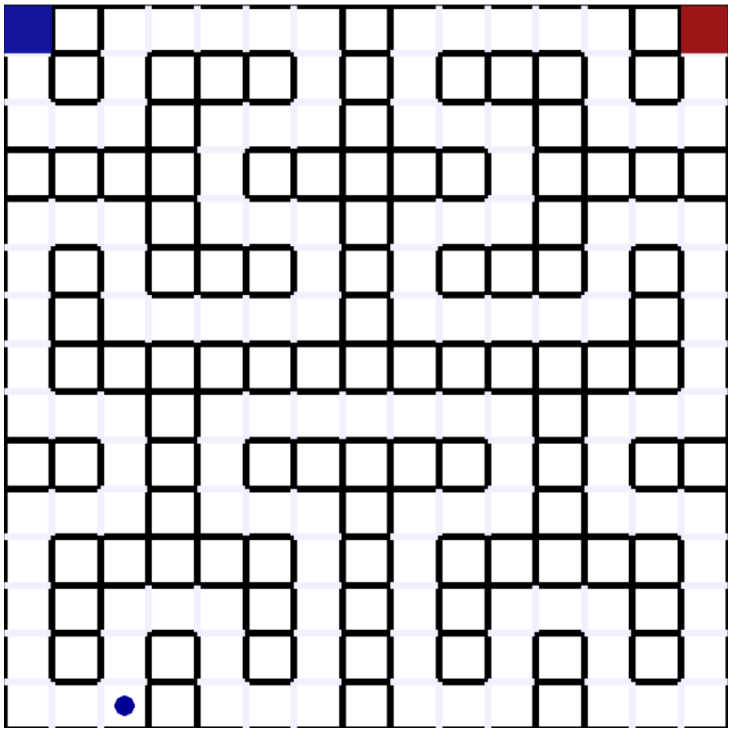


Figure 12: 15x15 Hilbert Curve Maze

2.1 New SOTA Method for Maze Solving

One relatively new method for maze solving is called "PlaNet: Learning Latent Dynamics for Planning from Pixels" [2] which was developed in collaboration with Google Deepmind and published in 2019. PlaNet is primarily used for solving reinforcement learning tasks where the environment is unknown. Also PlaNet uses images of the environment to learn it, so it can be shown and learn a variety of environments including maze solving and various grid environments. Instead of directly predicting from one image to the next, PlaNet instead compresses these images, which is also called a latent dynamics model, so that it can extract abstract information such as velocities of objects, positions, etc.

PlaNet differs from classical Q-learning in that instead of using a Q-Table to predict the future, it insread uses the aforementioned learned latent dynamics model (eg. the learned enviroment model) to simulate possible future states of the agent in the environment and plan future actions accordingly. The planning algorithm used is Monte Carlo

tree search. In addition PlaNet can capture uncertainty in environments, making better at stochastic enviroments compared to Q-learning.

2.2 Results on new mazes

2.2.1 Classic Q-learning

Figure 13 shows the steps per episode and rewards per episode figures of a row major 7x7 maze.

Figure 14 shows the steps per episode and rewards per episode figures of a row major 10x10 maze.

Figure 15 shows the steps per episode and rewards per episode figures of a row major 15x15 maze.

Figure 16 shows the steps per episode and rewards per episode figures of a Hilbert curve 7x7 maze.

Figure 17 shows the steps per episode and rewards per episode figures of a Hilbert curve 15x15 maze.

2.2.2 Deep Q-learning

Figures 18 and 19 show the maze layout, visitation frequency, steps per episode, and rewards per episode figures of a row major 7x7 maze.

Figures 20 and 21 show the maze layout, steps per episode, and rewards per episode figures of a row major 10x10 maze.

Figures 22 and 23 show the maze layout, steps per episode, and rewards per episode figures of a row major 15x15 maze.

Figures 24 and 25 show the maze layout, visitation frequency, steps per episode, and rewards per episode figures of a Hilbert curve 7x7 maze.

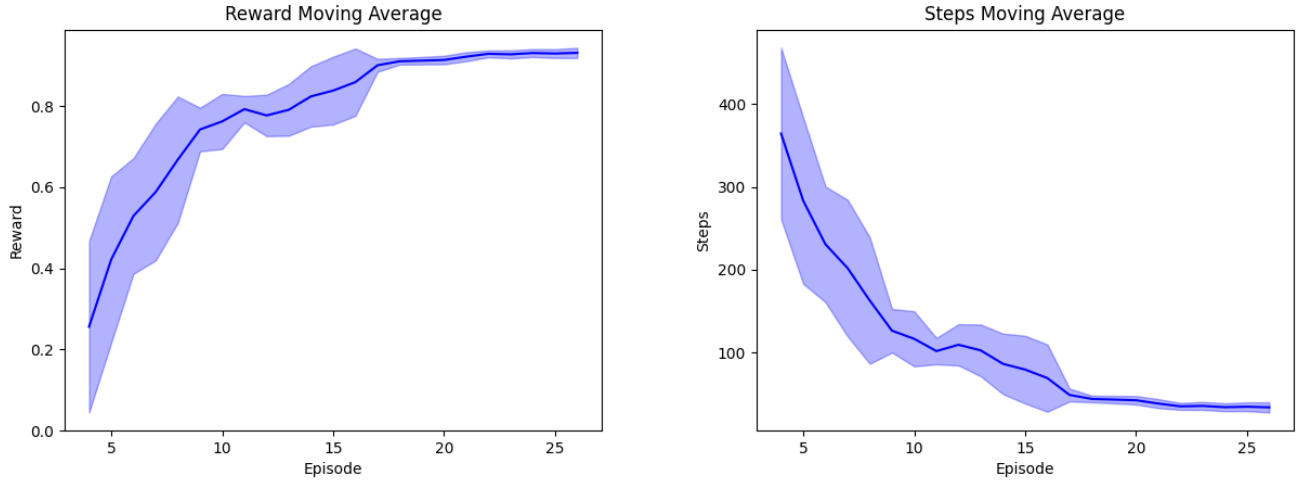
Figures 26 and 27 show the maze layout, steps per episode, and rewards per episode figures of a Hilbert curve 15x15 maze.

Note: some mazes were not able to be solved in 5000 episodes using deep Q-learning with default parameters. Therefore the visitiation frequency for these mazes is not shown. These unsolved mazes are: Row major 10x10, Row major 15x15, Hilbert curve 15x15 mazes

2.3 Observations and Reasoning

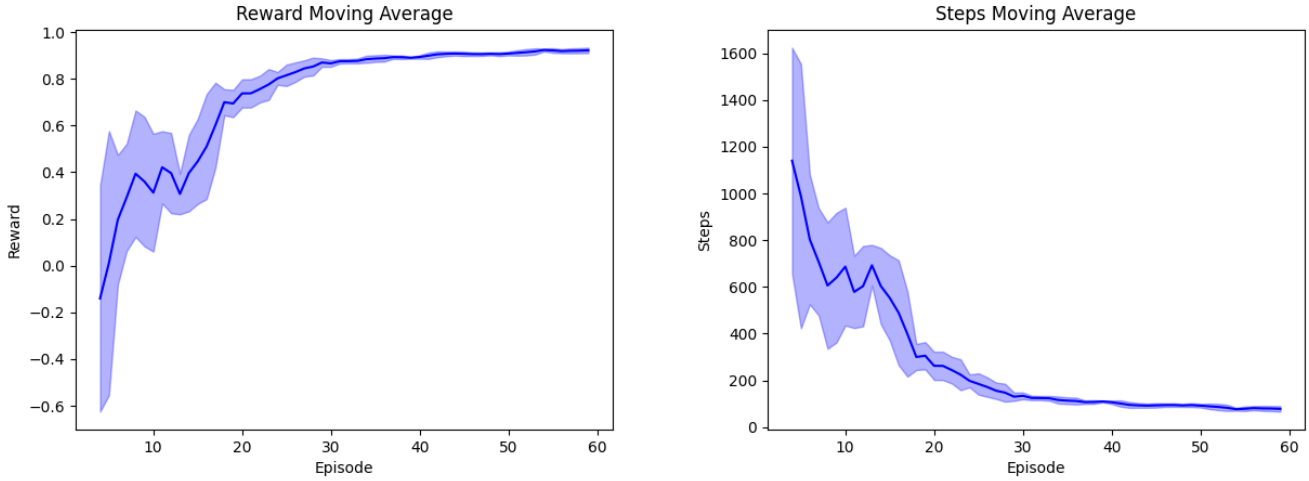
Deep Q-learning seems to have trouble solving large mazes and/or mazes with a relatively high number of steps needed to solve.

For example deep Q-learning was able to solve the 7x7 Hilbert curve maze. Looking at the cell visitation frequency 24b it seems that the agent gets stuck in the upper right corner, but once it learns this part of the maze, it quickly finishes the rest. This may indicate that the deep learning model is able to deal with low-depth recursion (2 level recursion in this case), or it may be that the maze is small enough to solve without learning the recursion.



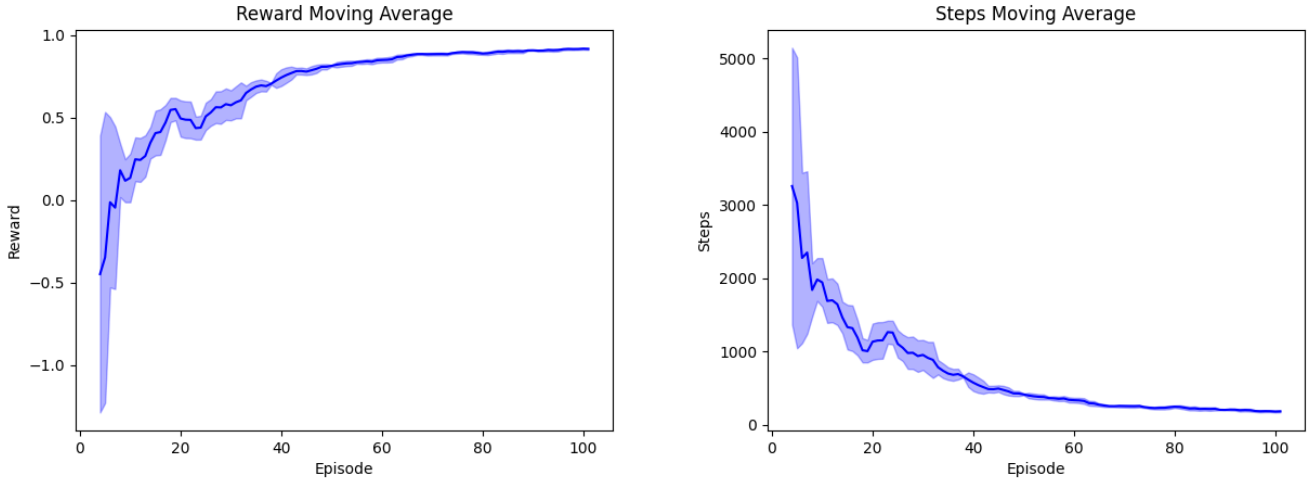
(a) Rewards per episode (b) Steps per episode

Figure 13: Q-learning Row Major 7x7 maze



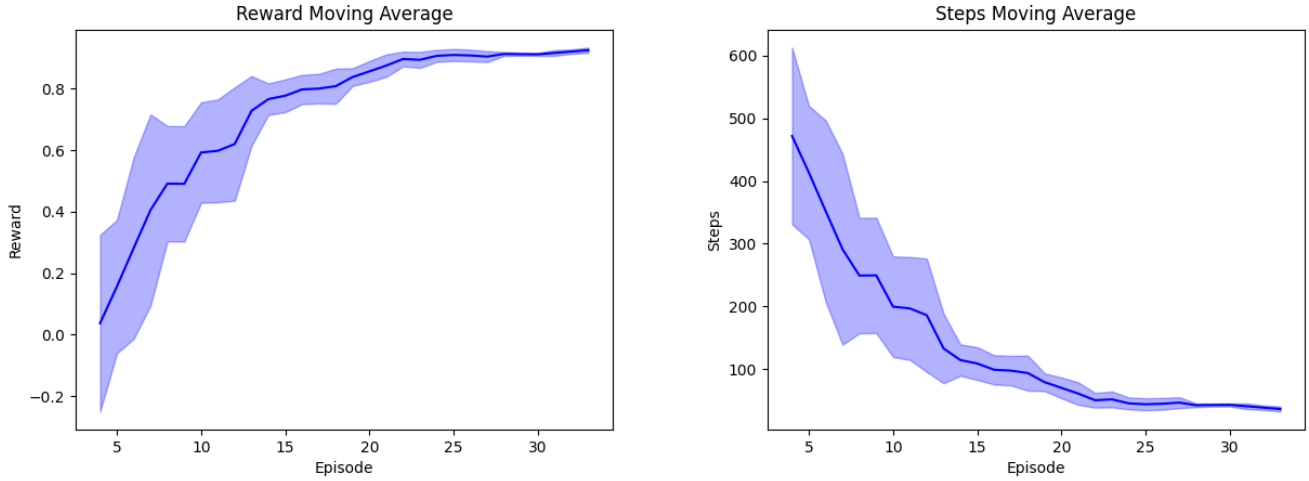
(a) Rewards per episode (b) Steps per episode

Figure 14: Q-learning Row Major 10x10 maze



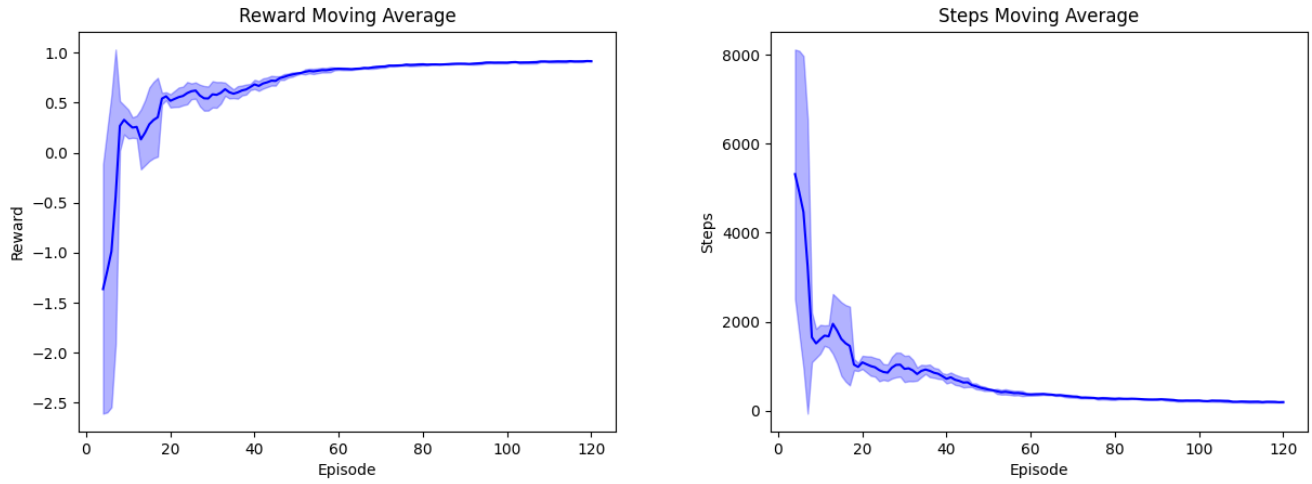
(a) Rewards per episode (b) Steps per episode

Figure 15: Q-learning Row Major 15x15 maze



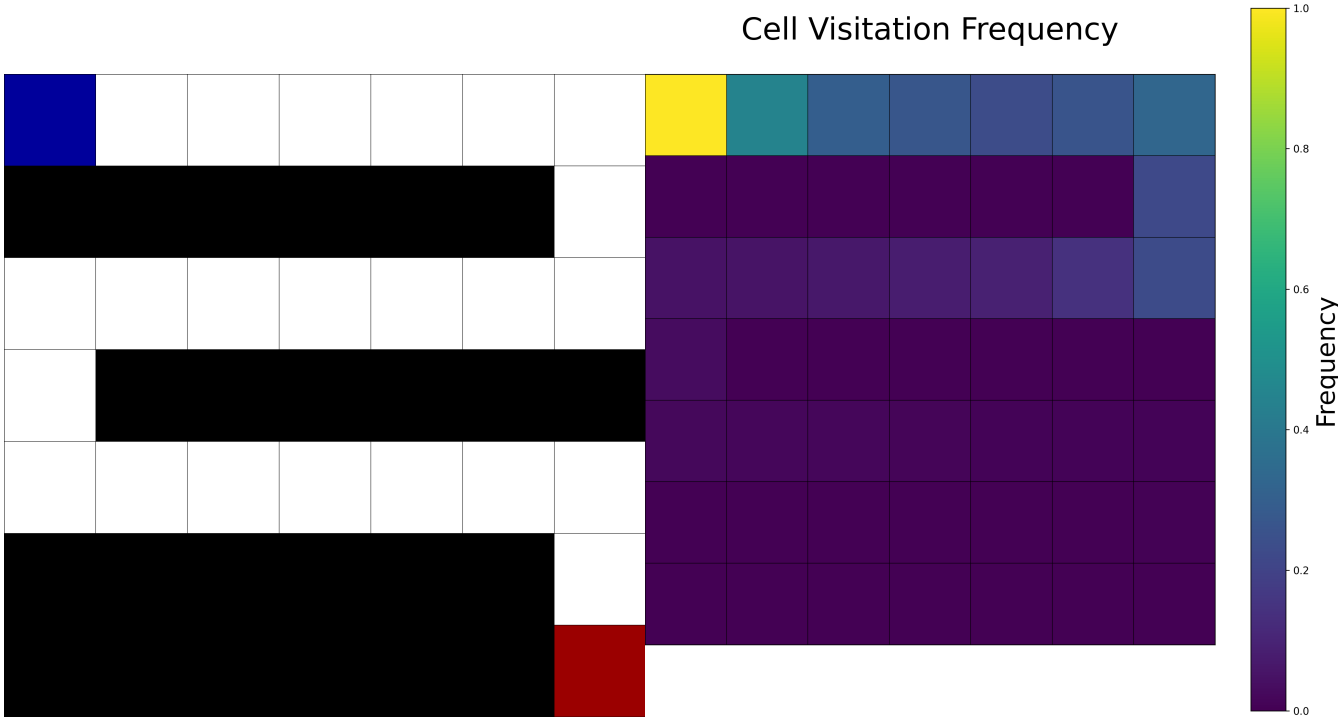
(a) Rewards per episode (b) Steps per episode

Figure 16: Q-learning Hilbert Curve 7x7 maze



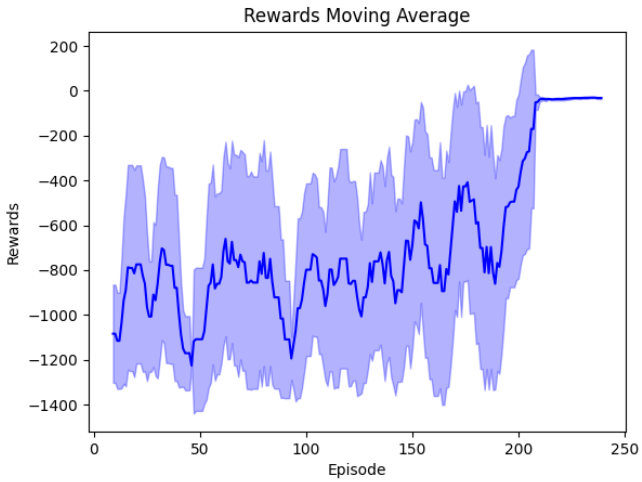
(a) Rewards per episode (b) Steps per episode

Figure 17: Q-learning Hilbert Curve 15x15 maze

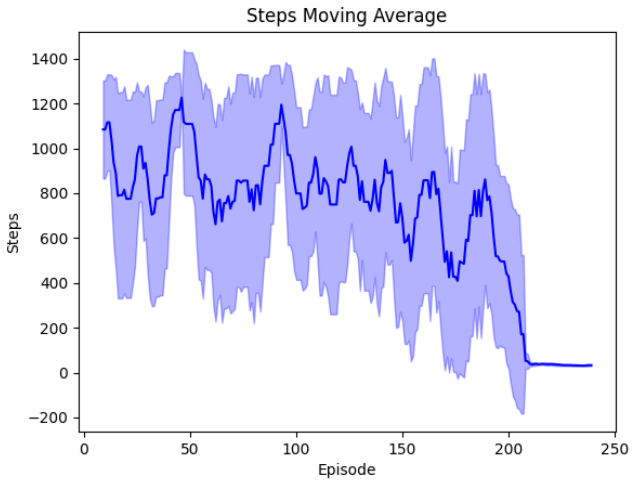


(a) Maze Layout (b) Visitation Frequency

Figure 18: Deep Q-learning Row Major 7x7 maze

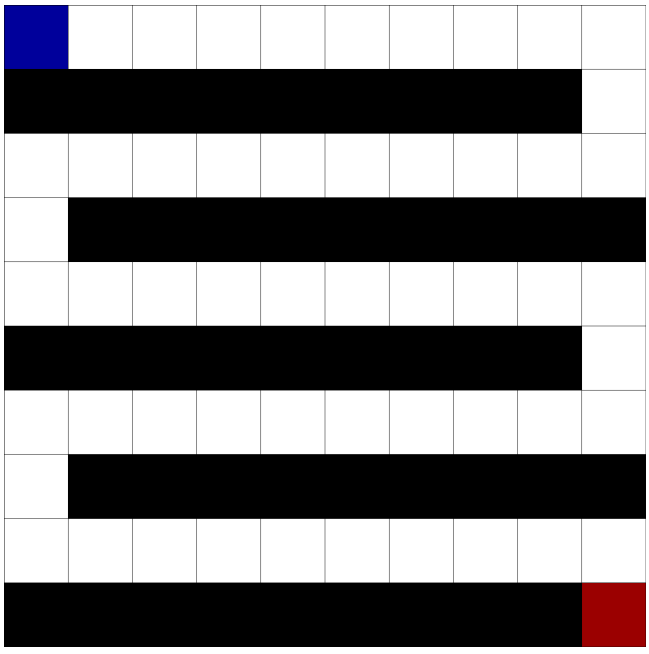


(a) Rewards per episode



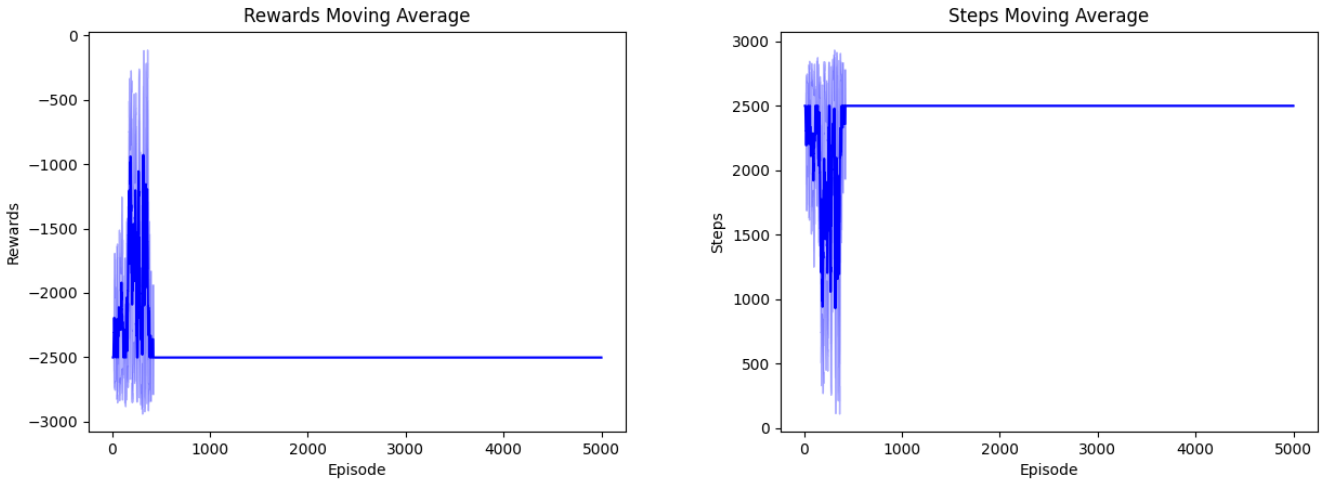
(b) Steps per episode

Figure 19: Deep Q-learning Row Major 7x7 maze



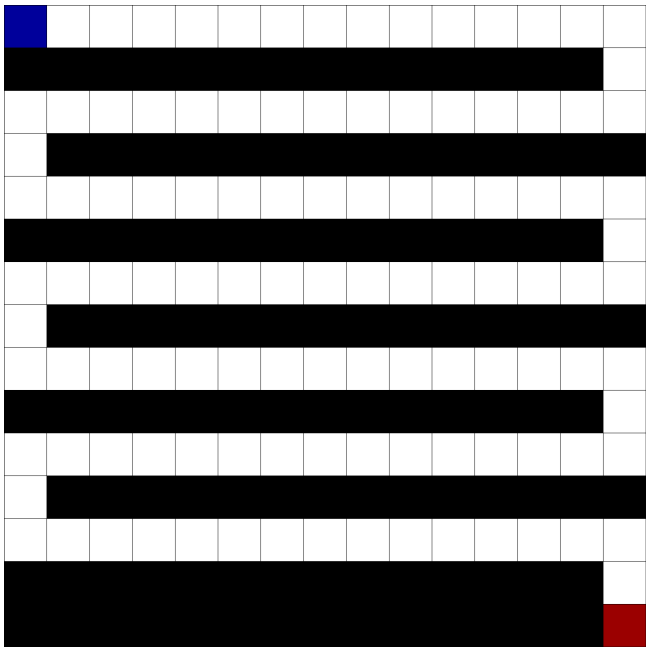
(a) Maze Layout

Figure 20: Deep Q-learning Row Major 10x10 maze



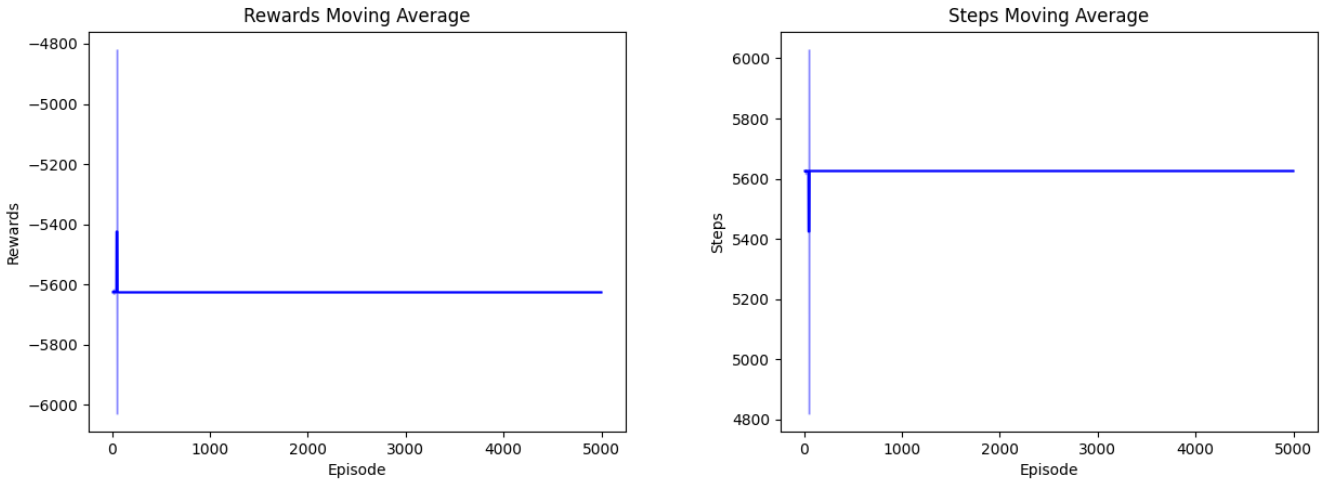
(a) Rewards per episode (b) Steps per episode

Figure 21: Deep Q-learning Row Major 10x10 maze



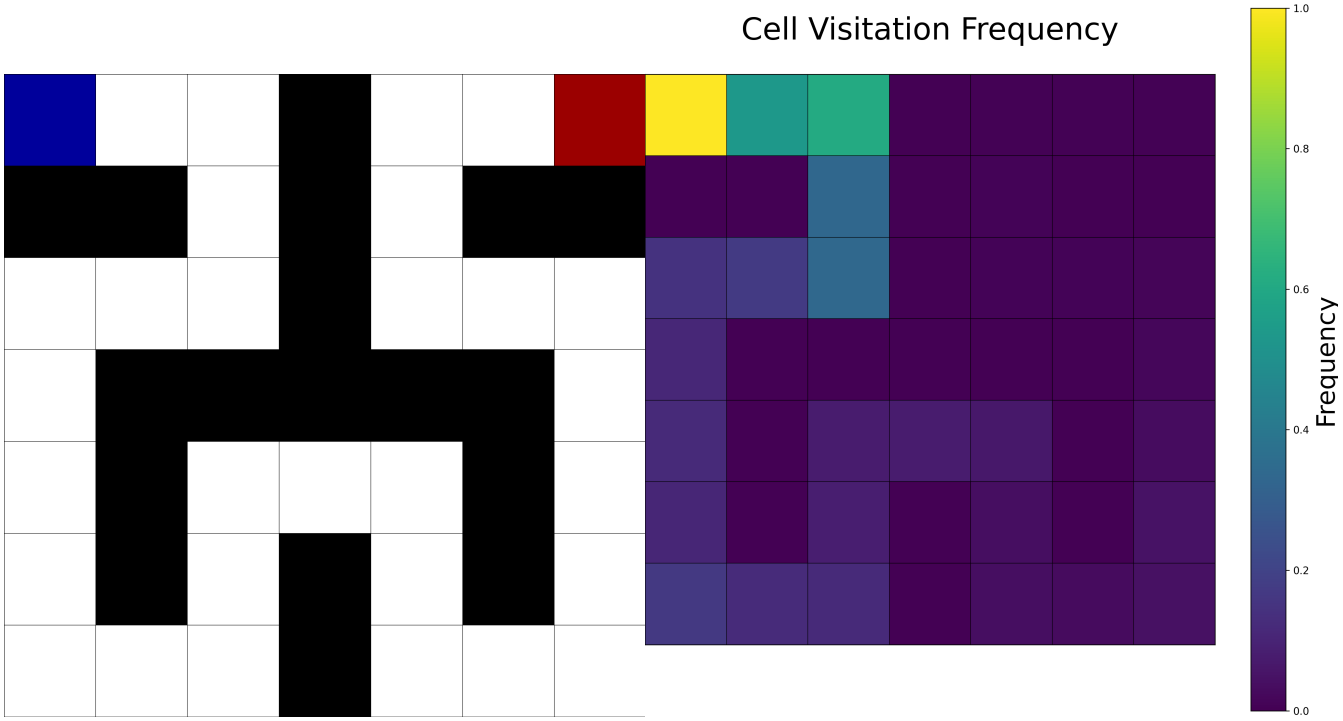
(a) Maze Layout

Figure 22: Deep Q-learning Row Major 15x15 maze



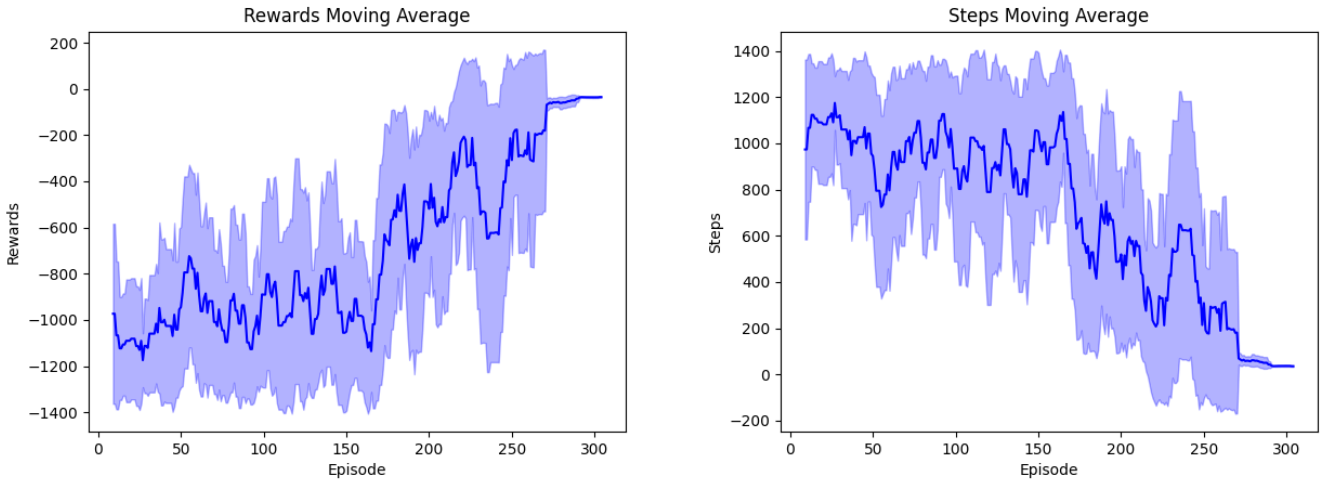
(a) Rewards per episode (b) Steps per episode

Figure 23: Deep Q-learning Row Major 15x15 maze



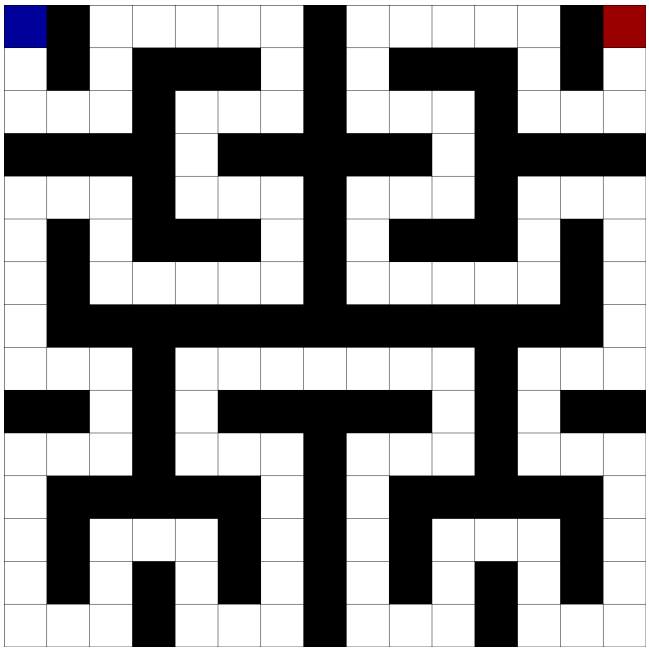
(a) Maze Layout (b) Visitation Frequency

Figure 24: Deep Q-learning Hilbert Curve 7x7 maze



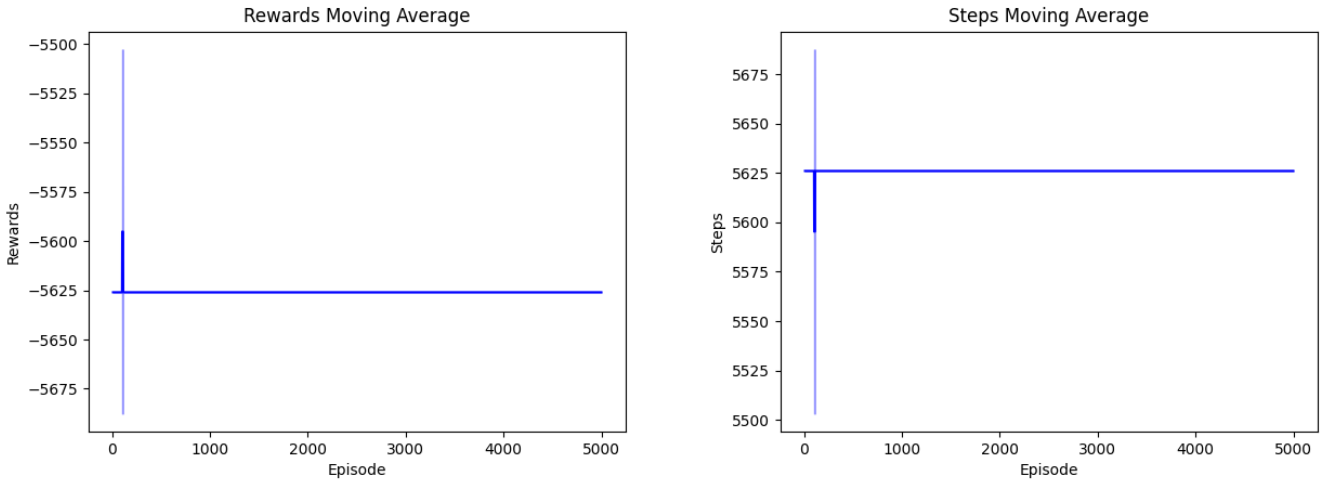
(a) Rewards per episode (b) Steps per episode

Figure 25: Deep Q-learning Hilbert Curve 7x7 maze



(a) Maze Layout

Figure 26: Deep Q-learning Hilbert Curve 15x15 maze



(a) Rewards per episode

(b) Steps per episode

Figure 27: Deep Q-learning Hilbert Curve 15x15 maze

However, Deep Q-learning was not able to solve the 15x15 Hilbert curve maze 27 (4 hours and 5000 episodes with no solution found), but this was solved by classical Q-learning 17. This indicates that the deep learning model used in the given code is not able to learn 4 levels of recursion required to solve the Hilbert curve maze or that the model is not large enough to remember the required 127 steps to solve the Hilbert curve maze.

Since the model was not able to solve a 15x15 Hilbert curve maze, I did not try larger mazes. Although, the code I wrote for generating mazes can generate an arbitrarily large Hilbert curve maze.

As for the row major mazes, deep Q-learning was able to solve the 7x7 row major maze shown in figures 18 and 19, but was unable to solve larger versions of this maze such as the 10x10 row major maze (20 and 21) and the 15x15 row major maze (22 and 23). The solver stopped after 5000 episodes, which is the default given parameter. One interesting observations we can observe in the step and reward plots is that after a certain number of episodes in these large mazes, the agent reverts back to its initial performance and stops improving. For example, in the 10x10 row major maze (figures 21 and 20) the model was able to improve from solving in 2500 steps to 1000 steps in around 500 to 600 episodes, but later it reverted fell back to 2500 steps and did not improve for the remaining 4500 episodes. I could not find out the reason for this.

In contrast to Deep Q-learning, Classical Q-learning was able to solve every maze faster, both in terms of episodes and real time. It seems that Deep Q-learning worse than classical q-learning for maze solving (since the state-action space is small, making it suitable for having a Q-table) or the default neural network used for deep Q-learning is not suitable for maze solving (maybe the default parameters are not optimal, or the neural network is too small, needs more layers, etc.)

2.4 Comparisons

Looking at the results in table 1, we can see that classic Q-learning performs much better in all mazes, taking around 9 times less episodes compared to Deep Q-learning.

An interesting observation is that the number of episodes needed for classic Q-learning to solve the given mazes is slightly more than the minimum number of steps needed to solve the maze in for **most** cases. For example Row-major 7x7 takes 24 steps to solve and Q-learning solved it in 27 steps; Row-major 10x10 takes 54 steps to solve and was solved in 60 episodes; Hilbert 7x7 takes 30 steps to solve, and was solved in 35 episodes

However, interestingly Row-major 15x15 takes 112 steps to solve, but is only solved in 100 episodes and Hilbert 15x15 takes 128 steps to solve and was solved in 120 steps. It seems that for larger mazes, the number of episodes needed is less than the minimum number of steps needed to solve the maze. Also note that these mazes have only 1 path from start to end and this behavior may not hold for mazes where there are multiple paths.

Table 1: Number of episodes to solve new mazes

Maze Name	Classic Q-learning	Deep Q-learning
Row-major 7x7	27	240
Row-major 10x10	60	Failed to solve
Row-major 15x15	100	Failed to solve
Hilbert curve 7x7	35	300
Hilbert curve 15x15	120	Failed to solve

References

[1] "Hilbert curve," Jan 2023. [Online]. Available: https://en.wikipedia.org/wiki/Hilbert_curve 9

[2] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 2555–2565. [Online]. Available: https://proceedings.mlr.press/v97/hafner19a.html 9