# **CS-370 Project Two**

Andrew Emilio DiStefano

SNHU

Dr. Troy Hawk

February 22, 2025

**Analyze the differences between human and machine approaches to solving problems.**

**Describe the steps a human being would take to solve this maze.**

The human approach to solving mazes on a computer screen involves the use of strategic eye movement to identify a viable route (Marquez & Zhao, 2014). The human approach balances exploring the maze environment with one’s eyes and relying on memory to move the mouse cursor through the maze through what one believes is the best route to the goal point (Marquez & Zhao, 2014). One interesting detail documented by Marquez and Zhao in their 2014 paper was the fact that visual exploration and guiding the mouse by memory almost never occurred concurrently (Marquez & Zhao, 2014). In other words, humans usually do not scan the entire maze with their eyes looking for the best route to the exit while moving the mouse toward the end of the maze (Marquez & Zhao, 2014). Human decision-making involves a variety of variables including “available visual information, memory, confidence, the estimated cost in time for exploration, and idiosyncratic tolerance for error” (Marquez & Zhao, 2014).

**Describe the steps your intelligent agent is taking to solve this pathfinding problem.**

The intelligent agent begins knowing nothing about its objective or current environment (Schoberg, 2020). The initial epsilon value before epsilon decay specifies the percentage of times the agent uses exploratory action in an attempt to find the best possible route (Nair, 2020) through the maze during the first epoch. In subsequent epochs, the agent then increasingly uses what it has learned about the maze environment to inform exploitative decision-making (Borrotti & Zangirolami, 2024). Since the agent learns more about its environment each epoch, the agent’s actions become less explorative as the value of epsilon is decomposed exponentially.

The data size is also configured to decompose across epochs since gradually lowering the data size improves learning efficiency by lowering the computational complexity without negatively affecting the agent’s performance (Hu et al., 2025). As the agent becomes more adept at solving the maze, the need for a larger data size becomes less and less significant (Hu et al., 2025). This means that the agent’s process of solving the maze becomes less computationally expensive (Hu et al., 2025) as time goes on.

**What are the similarities and differences between these two approaches?**

There are several striking differences between the way a human would solve this maze and the way our Q-learning agent teaches itself to reach the goal. After making my changes to the original source code, the intelligent agent is usually able to solve the maze in between 30 and 60 epochs. Since each epoch consists of 100 attempts, this means that the agent needs to be exposed to the maze environment 3000 to 6000 times in order for it to solve the maze 100 out of 100 times. A human would of course require much less exposure to such a simple maze, and would almost certainly reach the goal on their first try (for this maze, at least).

In their 2014 study on the human approach to maze completion, Marquez and Zhao found that exploration and exploitation almost never occurred concurrently (Marquez & Zhao, 2014). Hu et al. describe how exploration and exploitation happen simultaneously as Q-learning agents attempt to solve problems (Hu et al., 2025). This interesting difference highlights the fact that humans use a variety of different types of sensors in their everyday decision-making processes, including their eyes, ears, skin, nose, and mouth. It is also important to note that while artificial agents may be able to outperform humans in certain areas, the human mind is significantly more complex than current AI systems and is able to solve more complex problems in a wider variety of situations (Blankendaal et al., 2023).

Despite these significant differences, there are several interesting similarities between how a human being would solve our maze and how our intelligent agent approaches this problem. Humans use a combination of exploitation and exploration (Marquez & Zhao, 2014) the same way that our intelligent agent does in the treasure hunting game. Exploratory behavior tends to increase when individuals of certain human demographics are exposed to new environments (Petzke & Schomaker, 2022), which is reminiscent of the agent’s initial increased experimental action during early epochs. I also find the fact that large data sizes become less relevant as an agent learns more about its environment (Hu et al., 2023) reminiscent of a phenomenon which I have noticed where the mind seems to work harder to learn new skills than it does during the practice of skills which have already been learned. It is evident through these striking similarities between human thought and the decision-making process of artificially intelligent agents that the inspiration behind the logic behind the development of AI technology is a result of careful consideration and study into the inner-workings of the human mind (Cîrstea et al., 2020).

**Assess the purpose of the intelligent agent in pathfinding.**

**What is the difference between exploitation and exploration? What is the ideal proportion of exploitation and exploration for this pathfinding problem?**

Exploration in this context refers to “learning via interactions with an unknown environment” (Borrotti & Zangirolami, 2024). When an agent is exploring, it is improving its current knowledge regarding the potential result of each action (GeeksForGeeks, 2023). This process improves the accuracy of the agent’s estimations and gives the agent the ability to make more informed decisions later on in the learning process (GeeksForGeeks, 2024). Exploitation, on the other hand, refers to the use of learned information to make an informed decision (Borrotti & Zangirolami, 2024). Exploitation involves the agent choosing an action which it estimates will result in the highest reward in the short-term (GeeksForGeeks, 2023).

Effectively balancing exploration and exploitation requires balancing “the benefits of exploring unknown options to learn more about them, with exploiting known options, for immediate reward” (Bonawitz et al., 2021). An agent which overly engages in exploration is likely to waste a lot of time exploring the environment in a way which does not effectively bring the agent closer to achieving its goal (Nair, 2020). The graphs depicted in Figure 3 of Nair’s 2020 article shows that an agent which engages only in exploration is essentially acting randomly and not moving closer to achieving its goals within the system (Nair, 2020). Alternatively, an agent's over-engagement in exploitative action results in what is known as ‘epsilon greed,’ which refers to the suboptimal performance of an agent due to too much focus on short-term rewards (GeeksForGeeks, 2023).

The suggestions included in the constructive criticism of my last milestone submission led me to try a few different methods of epsilon decay in the pathfinding problem. I found that the ideal proportion of exploitation for this problem involved starting with a relatively high epsilon value which would decay exponentially across epochs. The high epsilon value worked well with an exponential decay rate because epsilon would not decay too much in the earliest epochs, but would begin to reduce drastically when the system had accumulated a significant amount of information relating to its environment and its possible choices from each state.

**How can reinforcement learning help to determine the path to the goal (the treasure) by the agent (the pirate)?**

Reinforcement learning algorithms use reward and punishment (negative reward) as they process data in order to learn how to reach their objective (Amazon Web Services, 2024). In the case of solving a maze, the agent moves in the direction which it estimates will result in the best possible outcome given the current state and the possible actions which it could take in said state (Schoberg, 2020). When training begins, the agent is only aware of its current state and its current earned reward (Schoberg, 2020). “It knows nothing about how many states there are, what the rewards are, or where the end states are” (Schoberg, 2020). By optimizing certain factors such as exploration factor (epsilon), epsilon decay rate, and data size reduction, a balance can be found between exploration and exploitation to yield the best possible results in terms of agent behavior in a given amount of time or over a given number of epochs (GeeksForGeeks, 2023). By experimenting with different values for these factors and variables, and by researching different approaches to similar problems, developers can continue to use this program to further optimize the agent’s solving of the maze problem for the best possible performance.

**Evaluate the use of algorithms to solve complex problems.**

**How did you implement deep Q-learning using neural networks for this game?**

During my research for the completion of this project, I took a deep dive into some of the different ways in which epsilon can be configured to decay over time. I paid close attention to the recommendations included in the constructive criticism of my last milestone for this project, and decided to implement all of the recommended changes in order to increase performance and efficiency.

I began by adding functions for the decay of epsilon over time. I decided to include three different decay functions in my program which would allow developers to choose between the use of linear decay, exponential decay, or discrete interval decay (also known as step decay). As the name implies, linear decay involves the linear decomposition of epsilon over a series of epochs (Bowyer, 2022). Exponential decay involves a rate of decay which increases exponentially across epochs (Bowyer, 2022). Discrete interval decay (or step decay) involves defining a number of steps at which epsilon will be decayed by a certain predefined value (Bowyer, 2022). For example, step decay could be configured so that epsilon is decreased by 0.05 every 10 epochs. I added the printing of each epoch’s epsilon value to the printed output for every epoch so that changes in epsilon using different decay methods could be monitored more easily.

Next, I decided to adjust the data size parameter dynamically based on the epoch number or win rate to potentially improve learning efficiency as recommended. Decreasing the data size after the agent has had a chance to learn about its environment reduces complexity while leaving performance relatively unaffected (Hu et al., 2025), as long as the data size is not reduced too quickly when not enough learning has taken place. In order to implement this optimization method, I added a function meant to decrease the data size every *n* epochs by a predefined discrete value. I used the modulo operator to determine whether dividing the current epoch number by a specific number would yield a remainder. If the remainder is zero, then the rest of the function will execute, which reduces the data size by a predefined value. I included separate variables for each of these values in order to facilitate adjustment. The docstrings and comments which I included in this function makes it clear which variable values must be changed to adjust the rate at which the data size is decreased. This way, colleagues can easily adjust the configuration of this function without needing to understand how the desired goal is achieved mathematically.

I then decided to implement another recommendation by programming an early stopping method to prevent overfitting (GeeksForGeeks, 2024). Before implementing this change, I noticed that the model would sometimes begin to overfit rather than reaching the terminal state. I would often reach a 100% win rate which would repeat over thirty or forty epochs before lowering significantly. The program would then continue into a sort of ‘limbo’ where no improvement in performance was being made. In order to address this issue, I added some code into the training loop so that the model would stop learning after achieving a 100 out of 100 success rate. This of course meant that overfitting would not be an issue as long as the model was able to reach the 100% success rate.

I then decided to focus on Dr. Hawk’s industry standard best practices recommendations. I modularized the qtrain function into smaller, more modular functions to improve readability, reusability, and maintainability. I created separate functions for the printing of each epoch and for the execution of each training loop. I added more comprehensive documentation, not only to my code, but to the original source code within the provided IPYNB file. Even someone with a relatively elementary understanding of reinforcement learning with TensorFlow, or of the Python language in general, would be able to develop an understanding of how the program works by reviewing my comprehensive series of comments and docstrings.

I adjusted the formatting to a consistent coding style to ensure readability. Lines with new comments are separated from the previous lines, and docstrings are used to inform the user of which values can be changed, commented, or uncommented in order to adjust the learning process. All variable and function names are in snake case, as was the case with the original code provided to us.

I then added robust error handling to gracefully inform the user of which function caused an error, and of the likely cause of specific errors. For example, the throwing of a TypeError or a ValueError alerts the user that they should check that the variables, parameters, and the values passed into them are of compatible data types. I also have a more generalized error-catching method for other errors which might occur. These changes certainly made the program easier to read, understand, and modify for optimal performance.

Finally, I began searching for an optimal configuration which would yield the best possible performance by the agent. After a comprehensive series of runs with a wide variety of different configurations, I found that the use of exponential epsilon decay with a relatively high decay rate of 0.98 resulted in the best performance, along with an initial epsilon value of 0.45 to assure that enough experimentation was taking place in early epochs. With these specifications, having the data size reduced by 2 every 7 epochs allowed me to assure a 100% success rate after between 30 and 60 epochs. The most typical performance resulted in 100% accuracy after between 35 and 50 epochs. While the best observed performance of the agent used in my previous milestone submission was 100% success after 53 epochs, the best observed performance of the agent used in my current submission was a 100% success rate after only 30 epochs. The changes recommended by Dr. Hawk, along with the implementation of strategies which I discovered through intense research, improved the agent’s best observed performance by over 43 percent.

**Sources:**

Amazon Web Services. (2024). *What is Reinforcement Learning?*

<https://aws.amazon.com/what-is/reinforcement-learning>

Blankendaal, R. A. M., van de Boer-Visschedijk, G. C., Boonekamp, R. C.,

Eikelboom, A. R., & Korteling, J. E. (2021, March 25). Human-

versus Artificial Intelligence. *Frontiers in Artificial Intelligence*, *4*,

622364. <https://doi.org/10.3389/frai.2021.622364>

Bonawitz, E., Costa, V. D., Ebitz, R. B., & Wilson, R. C. (2021). Balancing

exploration and exploitation with information and randomization.

*Current Opinion in Behavioral Sciences, 38*, 49-56.

<https://doi.org/10.1016/j.cobeha.2020.10.001>

Borrotti, M., & Zangirolami, V. (2024, June 7). Dealing with

uncertainty: Balancing exploration and exploitation in deep

recurrent reinforcement learning.

*Knowledge-Based Systems, 293*.

<https://doi.org/10.1016/j.knosys.2024.111663>

Bowyer, C. M. (2022, September 20). *Strategies for Decaying Epsilon*

*in Epsilon-Greedy.* Medium.

[https://medium.com/@CalebMBowyer/strategies-for-decaying-](https://medium.com/@CalebMBowyer/strategies-for-decaying-epsilon-in-epsilon-greedy-9b500ad9171d)

[epsilon-in-epsilon-greedy-9b500ad9171d](https://medium.com/@CalebMBowyer/strategies-for-decaying-epsilon-in-epsilon-greedy-9b500ad9171d)

Chia, R., Dekkers, R., & Zhou, Q. (2023). Are James March's

‘exploration’ and ‘exploitation’ separable? Revisiting

the dichotomy in the context of innovation management.

*Technological Forecasting and Social Change, 192.*

<https://doi.org/10.1016/j.techfore.2023.122592>

Cîrstea, B., Cuzzolin, F., Morelli, A., & B J Sahakian. (2020, May 7).

Knowing me, knowing you: theory of mind in AI.

*Psychological medicine*, *50*(7), 1057–1061.

<https://doi.org/10.1017/S0033291720000835>

GeeksForGeeks. (2023, January 10). *Epsilon-Greedy Algorithm in*

*Reinforcement Learning.*

[https://www.geeksforgeeks.org/epsilon-greedy-algorithm-in-](https://www.geeksforgeeks.org/epsilon-greedy-algorithm-in-reinforcement-learning/)

[reinforcement-learning/](https://www.geeksforgeeks.org/epsilon-greedy-algorithm-in-reinforcement-learning/)

GeeksForGeeks. (2024, February 15). *In Which Epoch Should I Stop*

*the Training to Avoid Overfitting?*

[https://www.geeksforgeeks.org/in-which-epoch-should-i-stop-](https://www.geeksforgeeks.org/in-which-epoch-should-i-stop-the-training-to-avoid-overfitting/)

[the-training-to-avoid-overfitting/](https://www.geeksforgeeks.org/in-which-epoch-should-i-stop-the-training-to-avoid-overfitting/)

### [Hu](https://openreview.net/profile?id=~Hao_Hu3), H., [Jiang](https://openreview.net/profile?id=~Yuhua_Jiang1), Y., [Li](https://openreview.net/profile?id=~Chenghao_Li1), C., [Wang](https://openreview.net/profile?id=~Quanwei_Wang1), Q., [Yang](https://openreview.net/profile?id=~Yiqin_Yang1), T., [Wu](https://openreview.net/profile?id=~Chengjie_Wu1), C.,

### [Zhang](https://openreview.net/profile?id=~Chongjie_Zhang1), C., [Zhang](https://openreview.net/profile?id=~Ziyou_Zhang1), Z., [Zhao](https://openreview.net/profile?id=~Qianchuan_Zhao1), Q., & [Zhong](https://openreview.net/profile?id=~Dianyu_Zhong1), D.

### (2025, February 11). Fewer May Be Better: Enhancing Offline

### *Reinforcement Learning with Reduced Dataset. International*

### *Conference on Learning Representations, 2025* (Under review).

<https://openreview.net/pdf?id=zqtql1YmlS>

Marquez, A. G., & Zhao, M. (2014). *Understanding humans’ strategies*

*in maze solving from eye-hand coordination* (Working Paper)*.*

Rutgers University. <https://doi.org/doi:10.7282/T38W3FX5>

Nair, S. (2020, April 8). *RL Series#3: To explore or not to explore,*

*that is the question.* Analytics Vidhya.

[https://medium.com/analytics-vidhya/rl-series-3-to-explore-or-](https://medium.com/analytics-vidhya/rl-series-3-to-explore-or-not-to-explore-1ff88e4bf5af)

[not-to-explore-1ff88e4bf5af](https://medium.com/analytics-vidhya/rl-series-3-to-explore-or-not-to-explore-1ff88e4bf5af)

Petzke, T. M., & Schomaker, J. (2022). A bias toward the unknown: individual

and environmental factors influencing exploratory behavior.

*Annals of the New York Academy of Sciences*, *1512*(1), 61–75.

<https://doi.org/10.1111/nyas.14757>

Schoberg, S. (2020, April 9). *Introduction to Reinforcement Learning*

*(Q-Learning) by Maze Solving Example.* Analytics Vidhya.

[https://medium.com/analytics-vidhya/introduction-to-reinforcement-](https://medium.com/analytics-vidhya/introduction-to-reinforcement-learning-q-learning-by-maze-solving-example-c34039019317)

[learning-q-learning-by-maze-solving-example-c34039019317](https://medium.com/analytics-vidhya/introduction-to-reinforcement-learning-q-learning-by-maze-solving-example-c34039019317)