Chad Netwig

Professor Susalla

CS-370-H5862

16 June 2022

Project Two: Design Defense

1. **Analyze Human vs. Machine Intelligence**

There are fundamental differences between human and machine approaches to solving problems, such as the pathfinding problem found in the *Treasure Hunt* game. Humans use real world problem-solving (RWPS) that involves a systematic interaction with the environment while they are attempting to solve a given problem (Sarathy, 2018). Humans use past experiences from their memory that they have acquired through experience either directly or from other people when trying to solve problems (Hanan, 2020). They also use trial-and-error if the problem presented in the environment is something completely new to them.

Machines solve problems differently than humans in that they use various well-defined Machine Learning (ML) algorithms to process information and learn about a given environment and estimate the accuracy of its learning, all in the attempt to solve a particular problem, such as the pirate agent finding the treasure in the *Treasure Hunt* pathfinding game (Hanan, 2020). For instance, to solve the pathfinding problem, a Model-Free Reinforcement Learning (RL) algorithm like the Q-Learning algorithm is commonly used by developers where an agent learns to behave in an environment within which it interacts by being rewarded for desired behaviors and punished for negative ones to the end of incentivizing the agent to maximize its overall rewards that push it iteratively toward the solution in an episodic task (Carew, 2021).

In the case of solving the *Treasure Hunt* pathfinding problem, humans would quickly identify a path to the treasure by holistically viewing the maze with cells that the pirate can and cannot use to reach the treasure. However, the machine would use an RL algorithm along with a deep neural network (DNN) that would first explore the cells by trial-and-error, map and record the rewards and states, and recursively exploit the environment until it reaches the treasure cell.

A key similarity shared between the human and machine approaches is that both will exploit past experiences from memory and use trial-and-error to the end of solving the pathfinding problem.

1. **Purpose of Intelligent Agent in pathfinding**

The difference between exploration and exploitation is that exploration focuses on the agent randomly searching the environment to find out new things about it by mapping states to rewards, while exploitation involves the agent exploiting knowledge that it already knows about the environment with the goal of maximizing its rewards, such as the agent utilizing an epsilon-greedy policy to exploit the agent's current estimated values to get the most reward. (Kiit, 2022).

However, there is a trade-off between exploration and exploitation. Too much exploration gives too many random predictions which gives rise to poor algorithmic performance, though, it is necessary for the agent to initially explore the environment in order to build data that can later be used for exploitation (Gulli & Pal, 2017). With the Q-learning RL algorithm, for example, the epsilon-greedy approach is used to balance exploration and exploitation by randomly choosing between the two, where epsilon is a value that the algorithm uses to choose the proportion of times the agent explores versus exploits.

The ideal epsilon to solve the pathfinding problem is approximately epsilon = 0.1, which instructs the agent to explore the environment 1 time out of 10 exploits so that the Q-function is optimized efficiently in terms of returning more consistent Q-values, which are values that instruct the agent to take the action from its current state that will yield it the highest future rewards (Surma, 2018).

In the case of the pirate (agent) determining the path to the treasure (goal) in the *Treasure Hunt* game, the Q-learning RL algorithm can be used for experience replay where the algorithm builds a Q-table to keep track of the mappings between the agent’s states, actions, and its expected rewards (Q-values) that it can iteratively exploit to find the treasure. Specifically, after a large number of agent actions via the Q-learning algorithm, the Q-table becomes efficient enough to be utilized directly by the pirate to optimize its moves in order to quickly find the treasure (Beysolow, 2019).

1. **Use of Algorithms to Solve Complex Problems**

In the past decade, algorithms used in ML have made great strides in the direction of solving complex problems. One area in particular that exemplifies this progress is a subfield of ML called Deep Reinforcement Learning (Deep RL) that “can solve a wide range of complex decision-making tasks that were previously out of reach for a machine to solve real-world problems with human-like intelligence” (Torres, 2021).

In the attempt to approach human-like intelligence, Deep RL marries the strengths of artificial neural networks and reinforcement learning algorithms to solve complex problems, such as DeepMind’s AlphaZero algorithm that has reached superhuman levels of *Go* gameplay (Silver et al., 2017). Essentially, Deep RL algorithms combine deep neural networks (DNNs) to predict the best reward outcome probabilities for an agent with the advantages of reinforcement learning that, together, train a model to improve the actions an agent takes within any given stage in an environment to the end of maximizing its rewards.

Another example of a Deep RL algorithm is a Deep Q-Network (DQN) that combines Q-learning with a DNN and was the algorithm that I used to solve the pathfinding problem presented in the *Treasure Hunt* game. The DNN was implemented with three *Keras Dense* layers, each using Parametric Rectified Linear Unit (PReLU) activation functions, an *Adam* optimizer, and a mean squared error (MSE) loss function. The Q-learning functionality was implemented using a *for loop* iteration of 500 epochs and a *while loop* to train the agent on the maze with the following pseudocode:

While GAME\_STATUS = NOT\_OVER

DO:

Set VALID\_ACTIONS

Set PREVIOUS\_ENVIRONMENT\_STATE = ENVIRONMENT\_STATE

Get NEXT\_ACTION using Exploration (random VALID\_ACTIONS) or Exploitation

* (Use EXPERIENCE REPLAY data prediction) based on EPSILON

Increment EPISODE + 1

Call ACT function to Set ENVIRONMENT\_STATE, REWARD, and

* GAME\_STATUS

Set EPISODE = ARRAY with PREVIOUS\_ENVIRONMENT\_STATE,

* ACTION, REWARD, ENVIRONMENT\_STATE, GAME\_STATUS

Call EXPERIENCE.REMEMBER() function to store EPISODE (experience)

Call EXPERIENCE.GET\_DATA() function to get EXPERIENCE REPLAY data

Call Keras MODEL.FIT() function to train the neural network

IF GAME\_STATUS = WIN or LOSE Break

IF WIN\_RATE > EPSILON and PASSES GAME Break

References

Beysolow, T. I. (2019). *Applied Reinforcement Learning with Python: With OpenAI Gym, Tensorf low and Keras.* San Francisco: Apress.

Carew, J. M. (2021, March 29). *reinforcement learning*. SearchEnterpriseAI. <https://www.techtarget.com/searchenterpriseai/definition/reinforcement-learning#:%7E:text=Reinforcement%20learning%20is%20a%20machine,learn%20through%20trial%20and%20error>.

Gulli, A., & Pal, S. (2017). Deep Learning with Keras. Birmingham, UK: Packt Publishing.

Hanan, J. (2020, April 2). *Human Learning and Machine Learning – How they differ ?* Data Science Central. <https://www.datasciencecentral.com/human-learning-and-machine-learning-how-they-differ/#:%7E:text=Humans%20acquire%20knowledge%20through%20experience,not%20mean%20you%20are%20intelligent>.

Kiit, T. (2022, January 2). *Reinforcement Learning – Exploration vs Exploitation Tradeoff*. AI ML Analytics. https://ai-ml-analytics.com/reinforcement-learning-exploration-vs-exploitation-tradeoff/#:%7E:text=Exploration%20is%20more%20of%20a,lead%20to%20long%20term%20benefit.&text=Exploitation%20basically%20exploits%20the%20agent’s,to%20get%20the%20most%20reward.

Sarathy, V. (2018). *Real World Problem-Solving*. Frontiers. <https://www.frontiersin.org/articles/10.3389/fnhum.2018.00261/full>

Silver, D., & Hubert, T. (2018, December 7). *A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play*. Science.Org. Retrieved May 24, 2022, from https://www.science.org/doi/10.1126/science.aar6404

Surma, G. (2018, September 26). *Cartpole - Introduction to Reinforcement Learning (DQN - Deep Q-Learning)*. Medium.Com. Retrieved May 30, 2022, from https://gsurma.medium.com/cartpole-introduction-to-reinforcement-learning-ed0eb5b58288

Torres.Ai, J. (2021, December 14). *A gentle introduction to Deep Reinforcement Learning*. Medium. https://towardsdatascience.com/drl-01-a-gentle-introduction-to-deep-reinforcement-learning-405b79866bf4