Exploration of an Agent

Joshua Flores

CS370

Professor, Smith

**Key Differences Between Human and Machine Maze-Solving**

Humans and machines approach maze-solving with distinct strategies.

* **Human Approach:** Humans solve mazes by gathering information and learning through trial and error. They identify undesirable directions and adjust their path to reach the goal. This process often involves retracing steps and continuous refinement towards the desired outcome.
* **Machine Approach (Q-learning):** A Q-learning agent assigns values (rewards) to directions within a sequence of steps. It analyzes its position and the weights of potential actions, seeking the path with the highest cumulative reward.

**The Agent's Purpose:**

The intelligent agent in this pathfinding algorithm aims to find the optimal route to the goal. It gathers environmental information, prioritizes actions based on rewards, and continuously refines its path selection.

**Exploitation vs. Exploration:**

* **Exploitation:** The agent uses existing knowledge to choose actions that have yielded positive results in the past.
* **Exploration:** The agent ventures into uncharted territory, seeking new information and potentially better paths.

**Reinforcement Learning:** The agent learns by evaluating the outcomes of its actions. Good actions increase a path's reward value, while poor actions decrease it. This feedback loop allows the agent to optimize its decision-making.

**Deep Q-Learning:** This technique leverages a deep neural network within the Q-learning framework. The neural network approximates the Q-values (rewards associated with actions), enabling the agent to handle more complex mazes.

def build\_model(maze):

    model = Sequential()

    model.add(Dense(maze.size, input\_shape=(maze.size,)))

    model.add(PReLU())

    model.add(Dense(maze.size))

    model.add(PReLU())

    model.add(Dense(num\_actions))

    model.compile(optimizer='adam', loss='mse')

    return model

The following are what constitutes the creation of a deep Q-learning network (Juliani, 2017):

* **Multi-layer Convolutional Network:** Replacing a single-layer network with a more complex structure enhances the agent's ability to process intricate maze layouts.
* **Experience Replay:** Storing past experiences allows the agent to learn from previous successes and failures, improving its decision-making over time.
* **Target Network:** Using a separate network to calculate target Q-values stabilizes the training process, leading to more reliable results.

**Implementation**

Our Python code incorporates these DQN aspects:

* **Experience Replay:** The actor's experiences are saved and leveraged during training.
* **Multi-layer Network:** The model compilation function demonstrates the use of a multi-layered network architecture.

These implementations align with the description of Deep Q-learning as a process of "stochastic gradient descent on the Bellman equation, backpropagating the reward through the state space (or episode) and averaging over many trials (or epochs)" (Gulli et Al.). It is these implementations done throughout the code that allowed for deep Q-learning, however as can also be seen from above, as long as we implement the functionality what is practically the bellman equation as described above, then our implementation of the reward functionality is correctly implemented to be used correctly for a Q-learning algorithm neural network.

References

Botvinick, M., Ritter, S., Wang, J. X., Kurth-Nelson, Z., Blundell, C., Hassabis, D. (2019). Reinforcement Learning, Fast and Slow. *Trends in Cognitive Sciences, 23*(5), 408-422. <https://doi.org/10.1016/j.tics.2019.02.006>

Gulli, Antonio, and Sujit Pal. Deep Learning with Keras : Get to Grips with the Basics of Keras to Implement Fast and Efficient Deep-Learning Models, Packt Publishing, Limited, 2017. ProQuestEbookCentral,http://ebookcentral.proquest.com/lib/snhu-ebooks/detail.action?docID=4850536.

Juliani, A. (2017, July 19). *Simple reinforcement learning with tensorflow part 4: Deep Q-Networks and beyond*. Medium. https://medium.com/@awjuliani/simple-reinforcement-learning-with-tensorflow-part-4-deep-q-networks-and-beyond-8438a3e2b8df