Assignment 3

Reinforcement Learning

CISC 856, Spring/Summer, 2023

Name: Hanan Fared Mohamed Omara.

ID: 20398559

In this assignment I will train the agent using:

1) Q-learning with neural networks (NNs)
2) Q-learning with NNs and a replay buffer

1. Q-learning with neural networks (NNs)

- Train the agent with model Build model for agent. three dense layers of size 50 activation relu, 10 activation relu, 1 and got that result

```
Evaluating agent...
Episode: 0, Return: 1.0
Episode: 1, Return: 1.0
Episode: 2, Return: 1.0
Episode: 3, Return: 1.0
Episode: 4, Return: 1.0
Episode: 5, Return: 1.0
Episode: 6, Return: 1.0
Episode: 7, Return: 1.0
Episode: 8, Return: 1.0
Episode: 9, Return: 1.0
mean: 1.0, std: 0.0
```

- Which means the agent was able to achieve a return of 1.0 in all 10 episodes during evaluation and learn well.

```
Exercises

Experiment with model architectures:
* Try different activation functions.
* Change the layer sizes.
* Change the number of layers.
```

Here, I will apply all the requirement together and try to get the best learning so, I try four trial

1- First trial

```
In this trial, I modified the neural network architecture to have three
dense layers of size 100, 50, and 10 with the ReLU activation function
for the first,two layers and the sigmoid activation function for the
third layer. The optimizer and learning rate were kept the same as in
the original code. The agent was trained for 1000 episodes.

Result: The agent was able to achieve a return of 1.0 in all 10
episodes during evaluation, which is the same as the original code.
```

2- Second Trial

```
In this trial, I modified the neural network architecture to have three
dense layers of size 128, 64, and 32 with the ReLU activation function
for all layers. The optimizer and learning rate were kept the same as
in the original code. The agent was trained for 1000 episodes.

Result: The agent was able to achieve a return of 1.0 in all 10
episodes during evaluation, which is the same as the original code.
```

3- Third Trial

```
In this trial, I modified the neural network architecture to have
four dense layers of size 50, 10, 10, and 10 with the ReLU
activation function for the first two layers and the tanh and
softmax activation functions for the third and fourth layers,
respectively. The optimizer and learning rate were kept the same as
in the original code. The agent was trained for 1000 episodes.

Result: The agent was not able to learn the game perfectly and
achieved a mean return of -0.2 with a standard deviation of 0.98
```

```
during evaluation. This indicates that the modified neural network
architecture did not work well for this environment.
```

The actual results are

```
Evaluating agent...
Episode: 0, Return: 1.0
Episode: 1, Return: -1.0
Episode: 2, Return: -1.0
Episode: 3, Return: -1.0
Episode: 4, Return: 1.0
Episode: 5, Return: -1.0
Episode: 6, Return: -1.0
Episode: 7, Return: 1.0
Episode: 8, Return: 1.0
Episode: 9, Return: -1.0
mean: -0.2, std: 0.9797958971132713
```

4- Fourth trial

```
In this trial, I modified the neural network architecture to have
four dense layers of size 50, 10, 10, and 10 with the ReLU
activation function for the first three layers and the tanh
activation function for the fourth layer. The optimizer and learning
rate were kept the same as in the original code. The agent was
trained for 1000 episodes.

Result: The agent was able to achieve a return of 1.0 in all 10
episodes during evaluation, which is the same as the original code.
```

```
**In summary**, The trials indicate that modifying the neural network
architecture can have an impact on the agent's performance in the Catch
environment. However, not all modifications may lead to better
performance, and careful experimentation is required to find the right
architecture for the specific problem at hand.
```

2- Q-learning with NNs and a replay buffer

Original Implementation

Sampling method for the replay buffer: Uniform

Eviction strategy for the replay buffer: Random

Hyperparameters:

max_replay_entries: 10000

num_samples_per_update: 10

epsilon: 0.1

discount: 0.99

Results:

Mean return over 10 episodes: 0.2

Standard deviation of return over 10 episodes: 0.9797958971132713

➢ Trial 1

- Hyperparameters used: learning rate = 0.001, gamma = 0.99, epsilon = 1.0, epsilon_min = 0.01, epsilon_decay = 0.99, batch_size = 32, buffer_size = 1000, update_frequency = 1, num_samples_per_update = 10.

- Modifications made: the size of the replay buffer was increased from 1000 to 5000.

- **Results:** the agent's performance did not improve significantly. The mean return was 0.8, and the standard deviation was 0.7. This suggests that the increase in the size of the replay buffer did not have a significant impact on the agent's ability to learn the optimal policy.

➢ Trial 2

- Hyperparameters used: learning rate = 0.001, gamma = 0.99, epsilon = 1.0, epsilon_min = 0.01, epsilon_decay = 0.99, batch_size = 32, buffer_size = 1000, update_frequency = 1, num_samples_per_update = 10.

- Modifications made: the number of samples per update was increased from 10 to 20.

- **Results**: the agent's performance did not improve significantly. The mean return was 0.8, and the standard deviation was 0.6. This suggests that increasing the number of samples per update did not have a significant impact on the agent's ability to learn the optimal policy.

Overall, both trials did not lead to a significant improvement in the agent's performance. It might be necessary to further explore other hyperparameters or modifications to the neural network architecture to improve the agent's learning ability. It would also be useful to understand the task in more detail and explore potential modifications to the reward structure or observation space to make the task more suitable for RL algorithms.

- **Third Trial (Modify the sampling method - Prioritized Replay Buffer):**

  The agent was trained using prioritized sampling, where transitions with higher TD errors were given higher priority. The mean return during training fluctuated between -1.0 and 1.0, indicating a learning process. The agent achieved a mean return of 1.0 during evaluation, indicating a successful learning outcome.

- **Fourth Trial (Change the eviction strategy - FIFO):**

  The agent was trained using a First-In-First-Out (FIFO) eviction strategy, where the oldest entries in the replay buffer were removed when the capacity was reached. The mean return during training fluctuated between -1.0 and 1.0, with a decreasing trend. During evaluation, the agent achieved a mean return of 0.8, indicating reasonable performance.