**Deep Reinforcement Learning for Playing Snake Game**

Name: Low You Jun | Student ID: 28696492

***Keywords***— **Deep Q-network (DQN), Reinforced Learning (RL)**

1. **Introduction**

“The Snake” is a popular game created in 1997 being first introduced in Nokia Phones. The snake begins with a length 1 and increases its length by 1 after eating a “fruit”. The problem lies with the limitation of the human as it requires a combination of spatial and mechanical skills.

RL is a subset of machine learning that involves an agent to take an action in a deterministic environment (state) and has been a popular algorithm used in playing games to outperform humans. In RL, the agent aims to maximize the rewards with the best action by employing trials and errors in a given current state [1]. As the learning progresses, the agent will find the best action based on the Q-table consisting of a map of action-state pairs to rewards after accumulating short-term rewards to a long-term reward. However, Q-table is no longer feasible for infinite spaces. This has posed a challenge on traditional reinforcement learning. A deep reinforcement learning (DRL) is introduced to tackle this issue as the policy decision is evaluated through the deep neural network instead of a table.

A related work of using DQN to play Atari 2600 games has demonstrated its outperforming results over all the previous reinforcement learning algorithms by adopting a deep convolutional neural network along with experience replay mechanism, successfully controlling the policies from images frame from video games [2]. The proposed method has successfully adopted the DQN algorithm on playing Snake game with a high score of 83 [3]. The states are in an array of Boolean variables consisting of immediate dangers, snake movements and the directions of the fruit.

1. **Methodology**

This section discusses the DQN dataset and design of model architecture. The coding of the model is deeply inspired by Patrick Loeber [4].

***A. Datasets***

The dataset of the model are the states in the game. Table 1 shows the 11 states and final output of the DQN model

Table 1: State boolean

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **States** | | | |
| **Danger Direction** | Straight | Right | Left | |
|
|
| **Last Movement** | Up | Down | Left | Right |
|
|
|
| **Food Direction** | Up | Down | Left | Right |
|
|
|

***B. Model***

The following Deep Q-Network (DQN) adopted:

The Q value is updated according to the Bellman’s equation, Q value is initialised with random weights. Initialize memory, *D* with capacity of *N.*

Epsilon-greedy, initialised = 1.

For episode = [1 , M]:

For *i* = [0 , *T*]:

1. Get the current state, (observation space).
2. With epsilon-greedy, , select random action, . Otherwise, So as the training goes on, the random action would minimize and relies more on its learnt network. (Exploration or Exploitation)
3. Execute and and snake will move according to the . Reward, is given and the new state, is captured.
4. Stores the into the memory, *D*. Update the stack of the last 4 frames.
5. Sample random minibatch of transition from *D*. (Replay Memory)
6. Then update Q-value, according to the Bellman’s equation. But if the game ended at , .
7. Back propagation: where

As for Step 5, after storing the experiences into the memory buffer, *D* with enough experiences, then sample a random batch of experiences from the memory buffer (1-6).

Figure 1 below shows the block diagram of the model architecture aforementioned.

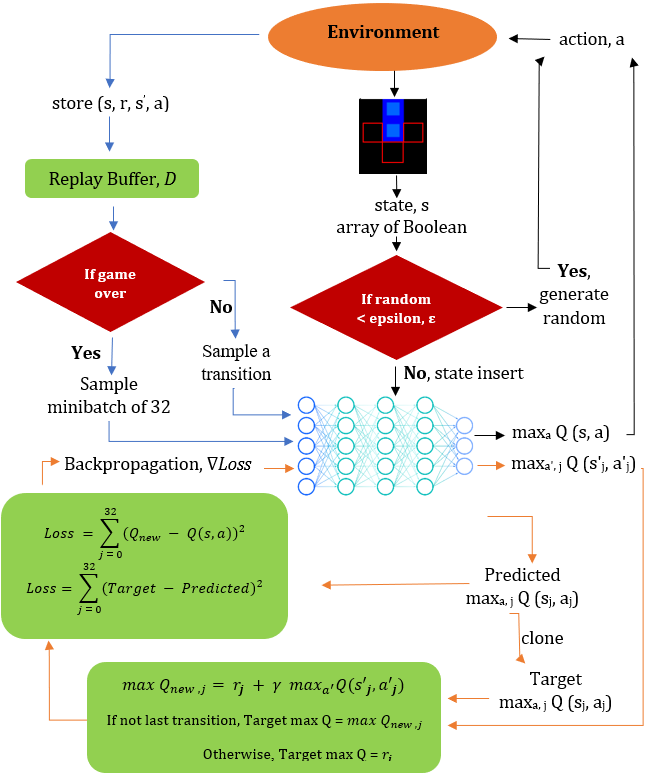
****

Figure 1: Block diagram of model architecture

1. **Results and discussion**

This section discusses the performance and the results from the model. Table 2 shows the case studies of the experiment done.

Table 2: Case Study for DQN

|  |  |
| --- | --- |
| Model | Hyperparameters |
| A | 2 linear layers with 1 hidden ReLu layer in between. 11 input nodes, 256 hidden nodes and 3 output nodes. |
| B | 3 linear layers with 2 hidden ReLu layers in between. 11 input nodes, 256 first hidden layer nodes, 128 second hidden layer nodes and 3 output nodes. |
| C | 4 linear layers with 3 hidden ReLu layers in between. 11 input nodes, 256 first hidden layer nodes, 128 second hidden layer nodes, 64 third hidden nodes layer and 3 output nodes. |

After running the case studies for \_\_\_\_ epochs, Figure 2, 3 and 4 below shows the plot for all the aforementioned case studies

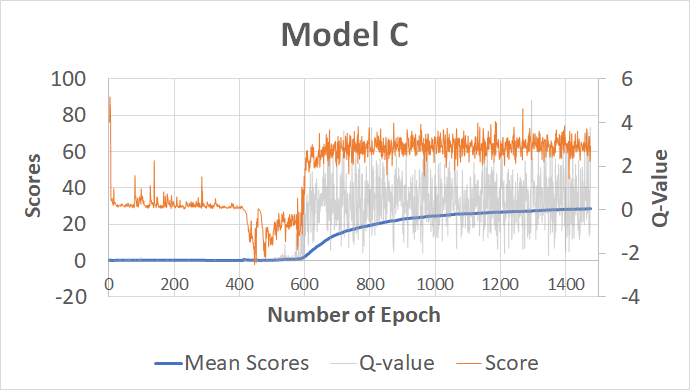


Figure 2: Plot of score, mean score and Q-value of model A

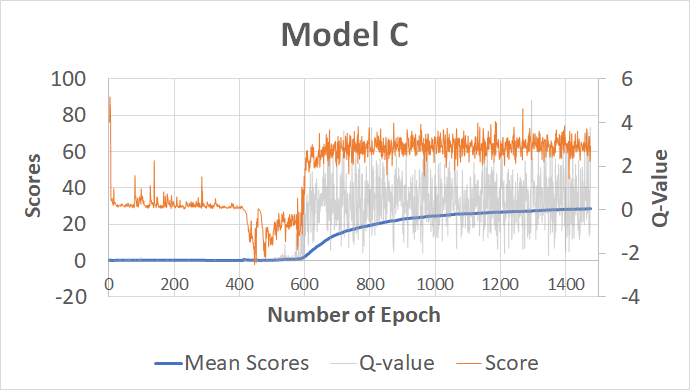


Figure 3: Plot of score, mean score and Q-value of model B

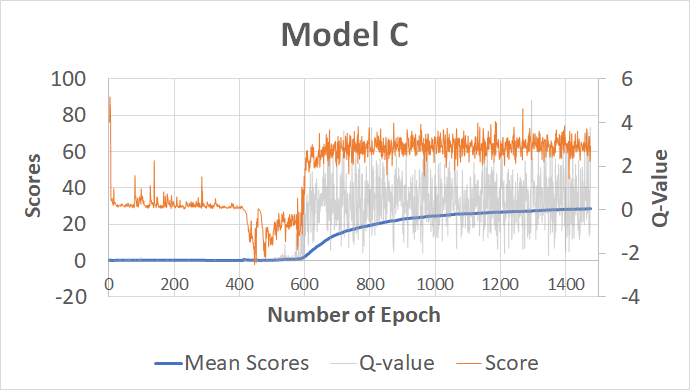


Figure 4: Plot of score, mean score and Q-value of model C

Table 3: Case Study Results

|  |  |  |
| --- | --- | --- |
| Model | Mean Score | Mean Q value |
| A |  |  |
| B |  |  |
| C |  |  |

Referring to table 3, Model B performs the best. It has a higher mean score, speed and accuracy.

1. **Conclusion**

In conclusion, the adopted DQN model is able to train the model to play Snake. The model with a \_\_ linear layer provides the best performance. Some limitations of this project is due to the software requirements, time constraints and lack of knowledge. Future work would include using CNN network, double DQN for more precise convergence.

1. **References**
2. W. Teri, "https://www.techopedia.com/reinforcement-learning-vs-deep-reinforcement-learning-whats-the-difference/2/34039," techopedia, 24 September 2020. [Online]. Available: https://www.techopedia.com/reinforcement-learning-vs-deep-reinforcement-learning-whats-the-difference/2/34039. [Accessed 11 May 2021].
3. V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra and M. Riedmiller, "Playing Atari with Deep Reinforcement Learning," *DeepMind Technologies,* 2013.
4. C. Mauro, "towardsdatascience," Medium, 15 November 2018. [Online]. Available: https://towardsdatascience.com/how-to-teach-an-ai-to-play-games-deep-reinforcement-learning-28f9b920440a. [Accessed 13 May 2021].
5. P. Loeber, "github.com," python-engineer, January 2021. [Online]. Available: https://github.com/python-engineer/snake-ai-pytorch. [Accessed 4 May 2021].