A CASE STUDY ON FLAPPY BIRD AI

Lappeenranta-Lahti University of Technology LUT

Foundations of Artificial Intelligence and Machine Learning, Project

December 10, 2024

Arno Törö, 0570180

1 INTRODUCTION

The game "Flappy Bird" was released in May 2013 and gained viral popularity in early 2014. The game gained popularity with its simple and minimalist gameplay mechanics, which were both addictive and frustrating for many players. In Flappy Bird, the players must navigate a bird between vertically moving pipes. The player earns one point for each pipe successfully passed.

This case study focuses on developing an AI agent to play Flappy Bird and beat a score of 100 in the game, which is considered challenging by the players. [1]

2 METHODS

The Flappy Bird AI agent in this study is developed using a Deep Q-Learning (DQN) algorithm. The agent learns by interacting with the game environment, making decisions based on the current game state and receiving rewards for actions that avoid losing the game. The agent is given the following information about the state of the game:

1. Horizontal distance to next pipe.
2. Vertical distance to lower next pipe.
3. Bird's current velocity.

Unlike many implementations of DQN that use Convolutional Neural Networks (CNNs) for feature extraction from images [2], this agent uses a simple feedforward neural network. The network takes the state vector as input and outputs Q-values corresponding to the two possible actions: flap or do nothing.
The agent's learning process is illustrated in Figure 1, showing the gathered information from the environment to the agent and the feedback loop for learning.

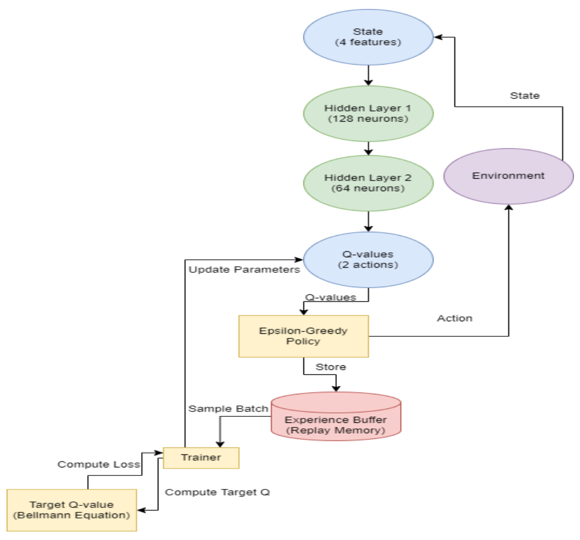

Figure 1. Method schematic showing the DQN learning process.

3 APPLICATIONS

The agent was trained for up to 250 iterations or until it achieved two consecutive games with scores exceeding 100 points. Due to the simple action space of Flappy Bird, the algorithm converged relatively quickly. During training, the agent balanced exploration and exploitation to optimize its decision-making and achieve high scores.
The hyperparameters used in the training process are summarized in Table 1.

Table 1. Training Hyperparameters for the Flappy Bird AI Agent.

Hyperparameter	**Value**
Learning rate α	0.001
Discount factor γ	0.95
Batch size	128
Replay buffer size	20,000
Exploration decay ϵ	From 1.0 to 0.1 over time

With the selected hyperparameters, the agent's performance varies during training. While the algorithm generally converges, it will occasionally diverge which results in suboptimal behaviour as shown in Table 2

Table 2. Training results: Iterations required for the agent to reach 100 points or fail (i.e., 250 iterations).

Training Session	**Iterations to Reach 100 Points or Fail**
1	99
2	250
3	250
4	90
5	250
6	250
7	105
8	95
9	94
10	97
Average	158

4 CONCLUSION

In this case study, the AI agent took an average of 158 iterations to learn how to achieve a score of 100 in Flappy Bird. The agent was able to find an optimal solution about 60% of the time. While the agent never failed miserably in the other cases, it was unable to reach the score of 100 points before 250 iterations of training.

Overall, these results show that reinforcement learning can work well in this kind of environment, but it's clear that the agent's performance depends heavily on the chosen hyperparameters. For future work, fine-tuning the algorithm such as by using convolutional neural networks and optimizing the hyperparameters, to enhance the agent's performance and reliability.

References

[1] Flappy Bird Wiki. Flappy bird. `https://flappybird.fandom.com/wiki/Flappy_Bird`, n.d. Accessed: 10 December 2024.

[2] S. Dutta. *Reinforcement Learning with TensorFlow: A Beginner's Guide to Designing Self-Learning Systems with TensorFlow and OpenAI Gym.* Packt Publishing, Birmingham, 1st edition, 2018.