**Research Question**

Can a Deep Q-Network (DQN) effectively learn to play the classic Snake game autonomously and consistently achieve high performance using reinforcement learning techniques?

**Introduction**.

The Snake game is a popular choice for testing reinforcement learning because it has a small number of possible moves, a changing environment, and rewards that come after a delay. It helps us build AI agents that can learn and adapt to complex situations, like how robots and machines need to make decisions in the real world. In this project, the goal is to create a Snake-playing agent that learns by itself using Deep Q-Networks (DQNs), and to see how it becomes smarter over time without giving it fixed rules.

**Related Literature**

1. DeepMind's DQN on Atari Games: Introduced DQNs with experience replay and target networks for stable training.
2. Snake Game AI with Reinforcement Learning: Although focused on Flappy Bird, this influenced our state representation and training loop.

**Proposed Approach**

Our implementation utilizes a Deep Q-Network architecture with several key components:

Environment: A custom Pygame implementation of Snake with configurable parameters

State Representation: 11 input features including:

1. Danger detection in all directions

2. Food location relative to snake head

3. Current direction of travel

4. Tail position awareness

**Progress and Achievements**

* Developed a working Snake environment using Pygame.
* Implemented a DQN agent using PyTorch with experience replay and epsilon-greedy exploration.
* Basic training loop with logging and early reward-based improvements.
* Visualizations show initial learning, agent avoiding walls more frequently.

**Challenges and Workarounds**

1. **Sparse Reward Problem:** At first, the agent had trouble learning because it only got rewards after eating food, which was delayed. To fix this, we gave small rewards when it moved toward the food and small penalties when it moved away.
2. **Catastrophic Forgetting:** Sometimes, the agent forgot the good strategies it had learned. To help with this, we made the memory (replay buffer) bigger and used a method that helps it remember more important experiences.
3. **Self-Collision with Longer Snake:** As the snake got longer, it started running into itself more often. We fixed this by giving the agent more information about where its tail was and made the game harder gradually by increasing the speed as the snake grew**.**

**GitHub Repository**  
<https://github.com/SunainaMakkena/AI_Project>