**Multiplayer Snake Game Using Deep Q-Learning: Detailed Report**

*Manashwy Padhi 2348528*

**1. Introduction**

This project implements a multiplayer version of the classic Snake game where two AI agents (players) compete against each other. The game environment is enhanced with unique rules, and the agents are trained using **Deep Q-Learning (DQL)**. The game offers a competitive edge as both snakes interact directly, and their movements affect each other's performance.

**1.1 Game Rules**

1. **Multiplayer Setup**: Two snakes compete simultaneously in the same environment.
2. **Food Mechanics**: Multiple food items are spawned randomly on the grid, and consuming them increases the snake's length and score.
3. **Collision Rules**:
   * A snake loses a segment of its body if its head collides with the opponent's body.
   * A head-on collision results in a loss for both players.
   * Collision with walls or one's own body ends the game for that snake.
4. **Reward Mechanism**:
   * Positive rewards for consuming food.
   * Negative rewards for collisions or movement away from food.

**2. Deep Q-Learning (DQL) Framework**

The AI players are trained using a reinforcement learning approach powered by **Deep Q-Learning**. The model learns from the environment and optimizes its actions to maximize cumulative rewards.

**2.1 Agent Design**

Each snake operates independently, and the game dynamics for the two agents are handled as follows:

* **State Space**:
  + The state comprises:
    - Current direction of movement.
    - Proximity of obstacles in the surrounding tiles.
    - Relative position of the closest food item.
  + The state is represented as a binary vector of length 11.
* **Action Space**:
  + Three possible actions:
    - Continue moving in the current direction.
    - Turn left.
    - Turn right.
* **Reward System**:
  + +50 for consuming food.
  + -10 for collision with walls or own body.
  + Negative proximity rewards for moving away from food.
  + Head-to-body collision with the opponent results in penalties for both players.

**2.2 Neural Network Architecture**

The neural network model is designed to predict Q-values for the agent's state-action pairs:

1. **Input Layer**: 11 neurons representing the state space.
2. **Hidden Layers**: Two fully connected layers with ReLU activation.
   * Layer 1: 512 neurons.
   * Layer 2: 256 neurons.
3. **Output Layer**: 3 neurons representing Q-values for the three possible actions.

**2.3 Training Process**

* The agent utilizes an **epsilon-greedy strategy**:
  + Starts with a high exploration rate (epsilon) and gradually shifts to exploitation as training progresses.
* **Experience Replay**: Stores past experiences in a replay buffer to train the model in batches, ensuring stability.
* **Loss Function**: Mean Squared Error (MSE) between predicted Q-values and target Q-values.

**3. Game Environment**

The multiplayer snake environment was implemented in Python using **Pygame** for visual rendering and game mechanics.

**3.1 Game Mechanics**

* **Food Placement**:
  + Food is randomly placed on the grid, ensuring it does not overlap with the snakes' bodies.
  + The environment spawns 10 food items at the beginning and replenishes them as they are consumed.
* **Snake Movements**:
  + Movement directions are controlled based on the output of the DQL model.
  + Collision detection ensures the integrity of the game's rules.

**3.2 User Interface (UI)**

* A minimalistic UI shows:
  + The position of both snakes and food.
  + Real-time scores of the players.
  + Color-coded representation:
    - Blue for Player 1.
    - Green for Player 2.
    - Red for food items.

**4. Performance Metrics**

**4.1 Training Results**

* Over 200 games, the agents learned to optimize their movements to avoid collisions and consume food efficiently.
* Scores improved significantly with training, as evident from the plotted results.

**4.2 Metrics**

1. **Player 1 (Blue)**:
   * **Best Score**: 18.
   * **Mean Score**: 8.4.
2. **Player 2 (Green)**:
   * **Best Score**: 15.
   * **Mean Score**: 7.2.

**5. Visualization**

**5.1 Training Progression**

A graph of the scores and mean scores for both players over the course of 200 games is shown below:

**Key Observations:**

* Both players showed steady improvement in their scores.
* Mean scores indicate that Player 1 (Blue) consistently outperformed Player 2 (Green).

**6. Challenges and Solutions**

1. **Challenge**: Balancing exploration and exploitation.
   * **Solution**: An adaptive epsilon-greedy approach was implemented to balance exploration and exploitation dynamically.
2. **Challenge**: Managing head-to-head collisions.
   * **Solution**: Added specific penalties for head-on collisions to discourage reckless behavior.
3. **Challenge**: Training instability due to large state space.
   * **Solution**: Limited the state representation to the immediate vicinity and relative position of food items.

**7. Future Enhancements**

1. **Multi-Agent Collaboration**: Introduce cooperative gameplay where both snakes work together to achieve a common goal.
2. **Dynamic Environment**: Add obstacles or moving food items to increase complexity.
3. **Improved Model**: Experiment with more complex neural networks or other reinforcement learning algorithms like PPO or A3C.

**8. Conclusion**

This project successfully demonstrates a competitive multiplayer Snake game powered by **Deep Q-Learning**. The AI agents effectively learn strategies to maximize their scores while avoiding collisions. The implementation provides a robust foundation for exploring advanced multi-agent systems and competitive AI training.

A graph with blue lines and orange lines

Description automatically generated

