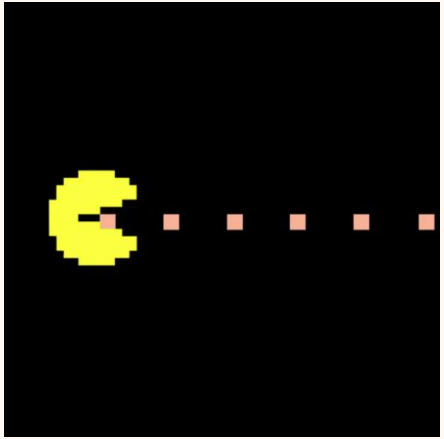
DSC190 - Final Project 12/04/2025

Learning to Play Snake with RL: A Q-Learning Approach



Shuchang Liu

Background

- Prior research often uses DQN to train Snake agents using raw image inputs and CNNs — effective, but computationally heavy and memory-intensive.
- Many approaches rely on large replay buffers (1M+ frames) and complex reward designs, which increase resource requirements.
- Recent work proposes memory-efficient DRL with:

  • image preprocessing (RGB → grayscale → binary),

  • smaller replay buffers,

  • lightweight CNNs,

  achieving similar Snake performance with dramatically reduced memory.

- This motivates exploring simpler RL agents—including tabular Q-learning—when the state space can be abstracted into compact features instead of relying on pixel images.

Motivation

Why Snake with RL?

- Classic sequential decision-making game
- Simple rules, but surprisingly deep strategy
- The Snake agent must learn from interaction, not supervision
- Requires: Long-term planning; Balancing risk vs reward; Learning safe movement patterns; Adapting to dynamic food locations

Challenges of the Snake Environment:

- Sparse rewards (food is rare, death is catastrophic)
- Large state space (grid positions, snake body configuration)
- Delayed credit assignment (early mistakes lead to death much later)
- Exploration difficulty (most random moves lead to dying quickly)

Algorithm Used: Tabular Q-Learning

Quick review: Q(s, a) = expected future reward of taking action a in state s

Bellman Update Rule: $Q(s,a) \leftarrow Q(s,a) + \alpha\left(r + \gamma \max_{a'} Q(s',a') - Q(s,a)\right)$

Why Tabular?

- The state is encoded as a compact feature vector and easy to store in a Python dictionary

Why Q-Learning Works Well for Snake?

- Suitable for discrete state + discrete action settings
- Off-policy nature lets agent learn optimal behavior while still exploring
- Able to learn long-term planning (avoid walls, chase food, stay alive)

Snake as an MDP

State (s): Compact feature vector describing:

- Danger: straight / left / right
- Food direction: x / y
- Current movement direction

Actions (a): Turn Left / Turn Right / Go Straight

Rewards (r): +10 eat food; –10 death; ±1 move closer/farther from food (shaping)

Transitions: Deterministic Snake game rules:

- Move → new head position
- Grow on food, die on collision

Episode Ends when snake hits wall or itself.

Exploration Strategy (ε-Greedy)

At each step:

- With probability $\varepsilon \rightarrow$ take a random action (explore)
- With probability $1 - \varepsilon \rightarrow$ take the best Q-value action (exploit)

Decay Schedule

- ε decreases gradually during training:

$$\boldsymbol{\varepsilon : 1 \rightarrow 0.05}$$

- High ε early $\rightarrow$ lots of exploration
- Low ε later $\rightarrow$ stable greedy behavior

Reward Shaping

Why Reward Shaping?

- Snake has sparse rewards(+10 for food; –10 for death; 0 otherwise)

This makes learning slow because feedback is rare.

Shaped Rewards to provide continuous guidance:

- +1 → if the snake moves closer to the food
- –1 → if the snake moves farther from the food

Result

- Smooth navigation
- Safe movement patterns
- Efficient food-seeking strategies

Without shaping, scores would stay low for a very long time.

Training Procedure

1. Agent Plays Episode

- Observe state
- Choose action using ε-greedy
- Receive reward and next state

2. Learn From Experience

- Short-term update from latest step
- Experience Replay: sample minibatch and train the Q-network

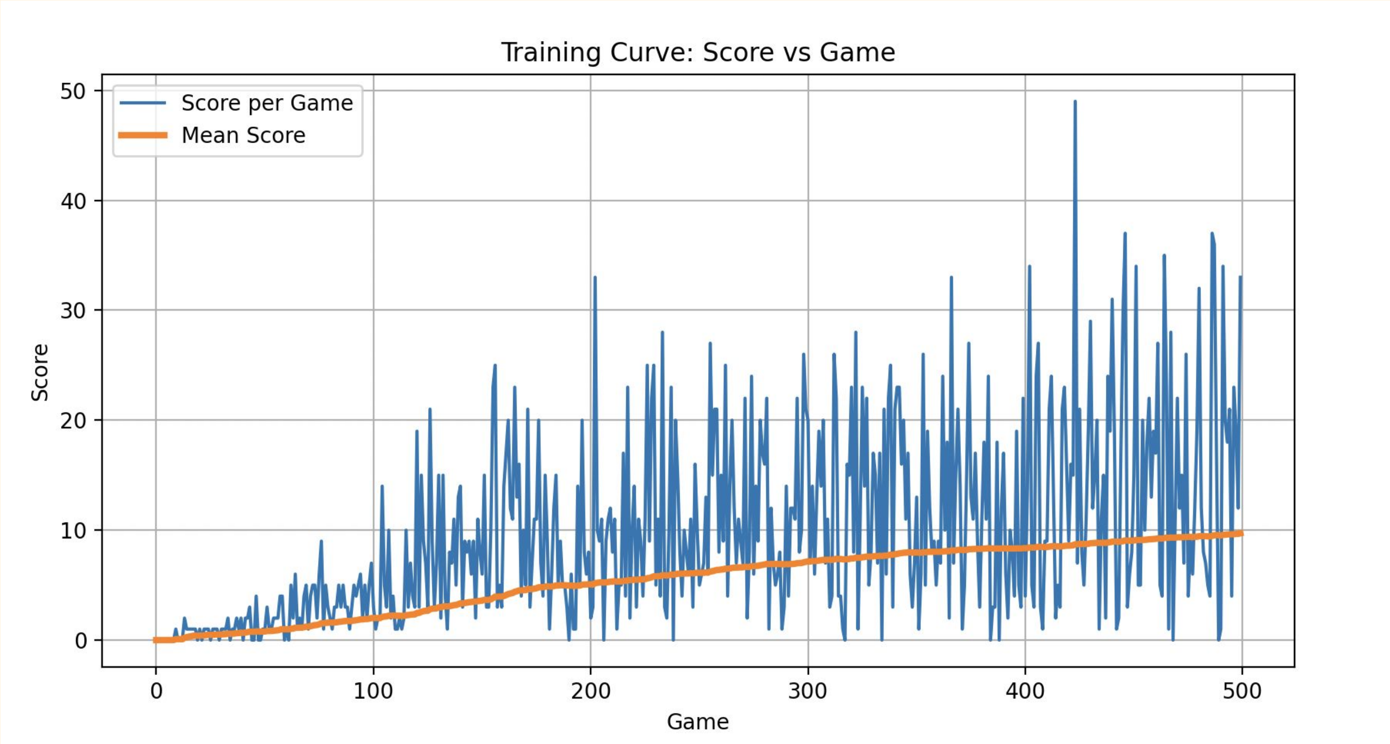
3. Exploration Decay

- Reduce ε after each game
- Stops at ε = 0.05 for small, safe exploration

4. Track Progress

- Record score each episode

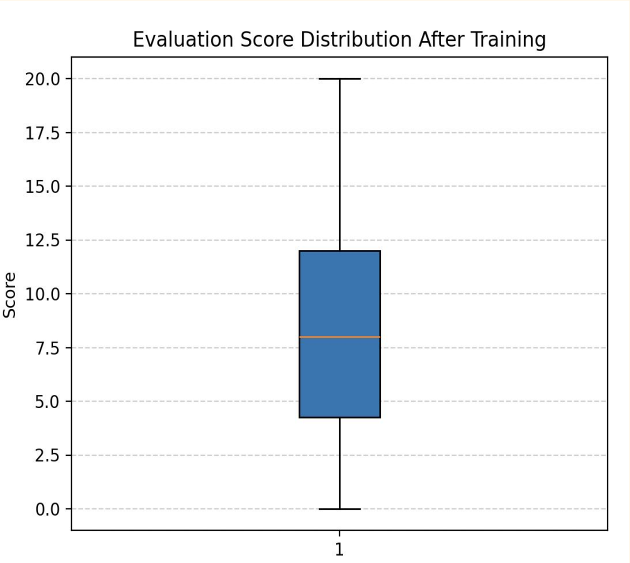
Results



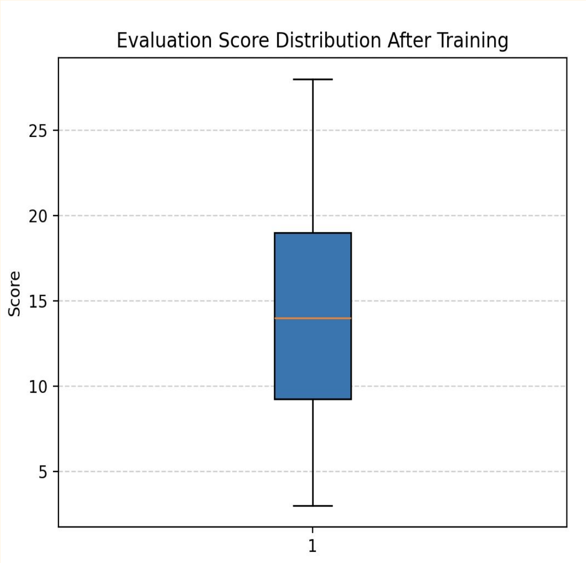
Training Curve: Score vs Game
Score per Game
Mean Score
Score
Game
0
10
20
30
40
50
0
100
200
300
400
500

Results

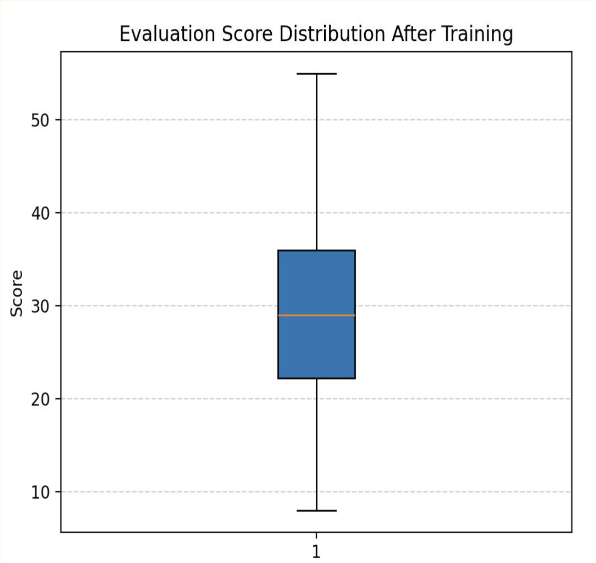
Early Training (~50 games)



Mid Training (~150 games)



Late Training (~500 games)



Discussion

Strengths

- Q-learning + reward shaping leads to fast and stable learning
- Agent develops clear food-seeking behavior and survival strategies

Limitations

- Q-table grows quickly → state abstraction is coarse
- Reward shaping can inject bias (learns to chase food but not full path planning)

Key Takeaways

What We Learned

- RL can teach an agent to play Snake without demonstrations
- Reward shaping greatly boosts learning efficiency
- Exploration (ε-greedy) is crucial in early training

What the Agent Achieved

- efficient food chasing
- collision avoidance
- basic space management

Future Extensions

- Adversarial perturbations: Add small disturbances to the agent’s state or environment to test robustness and design stronger policies

Thank you!