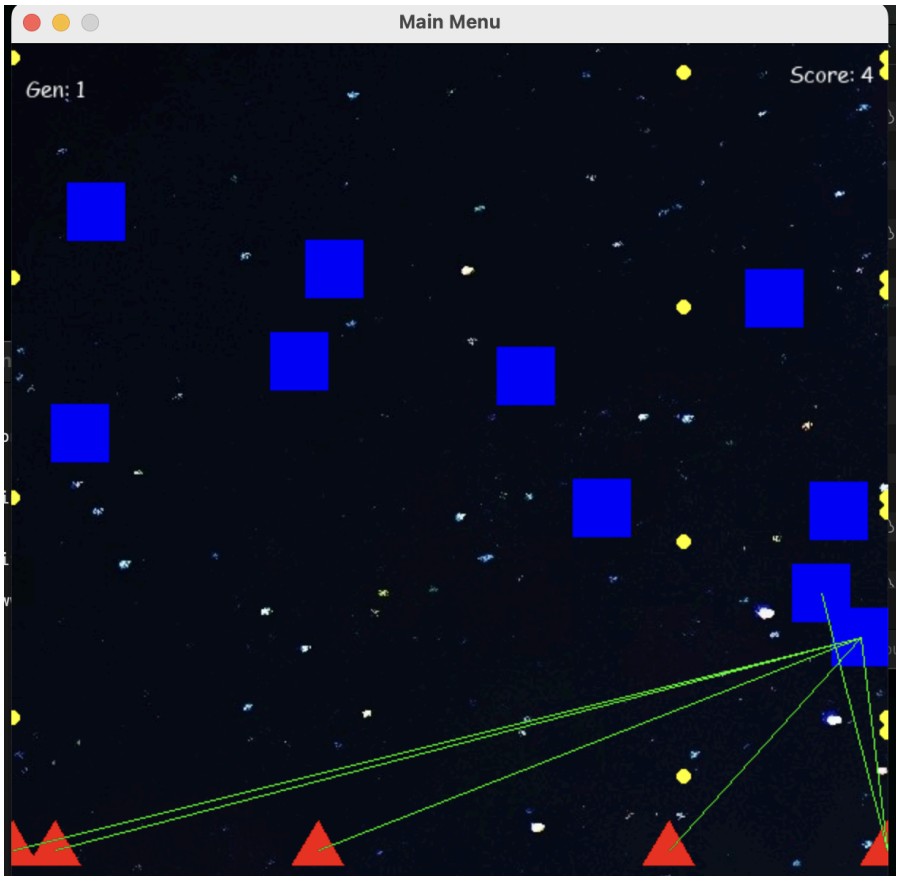
Project Title: Space Invaders AI with NEAT Algorithm

Screenshot of Project:



Overview

This project is a Python-based AI system designed to play a Space Invaders-style game using the NEAT (NeuroEvolution of Augmenting Topologies) algorithm. The objective is for AI-controlled bots to learn to play the game indefinitely when originally given no instruction about their objective. The NEAT algorithm enables the bots to evolve neural networks that optimize their gameplay performance over time, eventually reaching a level where they can navigate and shoot with minimal errors.

Game Structure

The game is similar in design to classic Space Invaders, featuring:

- A player bot which moves horizontally and shoots enemies that descend from the top of the screen.
- A point-based scoring system to reward the bot for successfully destroying enemies and avoiding damage.

NEAT Algorithm Application
The NEAT algorithm was used to train AI agents by:

1. **Initial Randomization**: Starting with a population of neural networks with random weights and topologies.
2. **Fitness Function**: Scoring bots based on their ability to survive, destroy enemies, and avoid taking damage. The longer a bot survives and the more efficient its gameplay, the higher its fitness score.
3. **Neuroevolution**: Evolving the neural networks over generations by selectively breeding top-performing networks while introducing mutations for diversity.
4. **Complexity Management**: Using NEAT's topology augmentation capabilities, the neural networks grow more complex only if it improves performance, preventing unnecessary complexity.

Implementation Details

- **Programming Language**: Python
- **Libraries**: NEAT-Python for implementing the NEAT algorithm, Pygame for game design and rendering.
- **Inputs to the Network**: The network receives input variables such as the bot's position, positions of enemies, and bullet trajectories.
- **Outputs from the Network**: Movement commands (left, right, shoot) that dictate the bot's actions in the game.

Training Process
The training process begins with a large, diverse population of bots. Each bot is evaluated in simulated gameplay for a certain period or until it is defeated. The highest-performing bots in each generation are selected for reproduction, allowing them to pass down their network configurations and weights with some mutations, optimizing them over generations.

Results
After several generations, the AI bot demonstrates a significant improvement in gameplay performance, moving and shooting with increasingly effective strategies. By generation 75 (approximately), the AI can survive for extended periods, effectively navigating the game environment with minimal errors, and occasionally achieving an endless gameplay loop.

Future Improvements

- **Custom Neural Network:** Hardcode my own neural network to use for the AI, shifting away from relying on the NEAT library

- **Dynamic Difficulty Scaling**: Introducing adaptive difficulty to challenge the AI further as it improves.
- **Obstacle Detection**: Adding more environmental obstacles for complex navigation challenges.