Artificial Intelligence
Neural Network - Autopilot Assignment

G00338424 – Ryan Harte

Introduction

This project uses a Neural Network (NN), ShipTrainer, to control a spaceship (ship) navigating through a scrolling tunnel autonomously. The main goal is for the NN to keep the ship flying without hitting the tunnel walls for at least 30 seconds, showcasing effective AI control of the ship.

AI Control

The game is set on a 30x20 grid where the ship can move up or down but is fixed on the X-axis. The NN's task is to make real time decisions to avoid the tunnel walls (black boxes). This setup tests the neural network's ability to learn and adapt to changing game states, emphasizing the importance of accurate and efficient AI control mechanisms.

Feature Engineering and Training Data

For the NN to navigate through the game efficiently, it's equipped to analyse the environment three columns ahead, as highlighted by the **sample()** method in **GameView.java**.

The process for gathering training data involves saving game states into **data.csv** and the ship movement actions into **expected.csv**.

The **data.csv** captures information from the three columns ahead of the ship. This data is vital for training the neural network on which manoeuvres are best based on upcoming challenges.

The **expected.csv** complements this by storing the correct movements (up, down, or straight) as per the situation detailed in **data.csv**.

After the training session, NN's configuration is saved in **trainedShipModel.data**. This file is a snapshot of the trained neural network, preserving its architecture and weights.

Neural Network Design and Configuration

The NN is designed with three main components for efficient operation.

1. **The input layer:** 7 nodes that process the game state data, specifically capturing information from the three columns directly ahead of the ship (as is set in the GameView.java - public double[ ] sample() method). This setup ensures that the network focuses on immediate obstacles and paths for navigation.

2. **The hidden layer:** 5 nodes, using the **TANH** activation function (Hyperbolic Tangent Function). This choice of activation function allows the network to handle nonlinear relationships in the data, essential for predicting complex manoeuvres the ship needs to make to avoid obstacles.

3. **The output layer:** 1 single node. This node outputs the decision for the ships next move, determining whether it should move up, down, or continue straight.

The network uses a sum of squared errors loss function (SSE), and training parameters are carefully chosen to balance learning speed and accuracy.

NN Training and Configuration Results

SECTION:	**RESULTS:**
Input Layer	Activation: Identity Transfer Function **Nodes:** 7 = [0.2, 0.5, 0.15, 0.55, 0.1, 0.6, 0.6]
Hidden Layer	Activation: Hyperbolic Tangent Transfer Function **Nodes:** 5=[0.01611446705156785,0.3534731213976976,0.5755226342097112,0.06943507492057151,-0.28322096299061117] **Weights:** [-0.39,0.26,-3.18,-0.40,-0.57] [0.05,0.02,0.11,-2.45,-0.27] [-0.20,-0.21,0.29,-0.66,-1.58] [0.11,-0.20,-2.64,-2.78,-0.29] [0.35,1.30,-1.12,-0.15,0.93] [0.19,0.80,-0.64,-2.60,0.44] [-0.10,-0.77,-1.83,4.56,-0.36] [-0.05,0.30,4.24,1.84,0.22]
Output Layer	Activation: Hyperbolic Tangent Transfer Function **Nodes:** 1=[0.9598315910711207] **Weights:** [-0.32] [-1.17] [3.30] [-3.77] [-1.34] [0.35]
Training Results	Loss Function: Sum of Squared Errors Loss Function, Softmax: false Alpha:0.001, Momentum:0.93, Max Epochs:50000, Max Error:0.00001 Time:2 secs, Epochs:39990, Training Error:0.000009999

The training process is efficient, allowing the NN to quickly learn the required control patterns to navigate the games environment. Achieving a low training error of 0.000009999 in a short amount of time demonstrates the model's capability to adapt and make accurate manoeuvres with the ship, highlighting the effectiveness of the NN’s architecture and training approach.

Conclusion

This NN/ShipTrainer Autopilot project successfully demonstrates the application of a NN in a gaming environment, specifically in relation to real time control tasks (moving the ship autonomously).