Battleship AI

Aman Tiwari

May-June,2017

Contents

1 About The Game

1.1 Introduction

Battleship is a board game played between two players. The game is played on a 10x10 board. First, each of the two players places their ships on their board. The ships are of varying length and can be placed vertically or horizontally on the board. In our game there are five ships of lengths 5,4,3,3 and 2. They are respectively called the Carrier, Battleship, Cruiser, Submarine and Destroyer. Our game also allows players to place the ships adjacent to each other. After this, the players take turns to guess the positions of ships of the opponent. They will be told if they hit or miss or if they have sunk a particular ship. The first player to guess the positions of all of their opponent's ships wins.

2 Probability

2.1 Introduction

The basic idea of the probabilistic approach was to find the most probable position on the grid to attack given the state of the board. The state of the board means that for each cell on the board we know if it is hit(part of a ship is present there), a miss or it has not been attacked. Assuming that all possible configurations of ships for a given state of board are equally likely, we find the most probable location - the cell which has the highest probability of a ship passing through it.We implemented it by making two modes for the AI - Hunt and Target.

2.1.1 Hunt Mode

The hunt mode of the AI comes to work when zero or more ships have been completely sunk and tries to find the next most probable location where a part of unsunk ship may be present. Again, it assumes that all possible configurations of the ships are equally likely.

2.1.2 Target Mode

The target mode comes to work when we get a hit from the hunt mode. The Target mode then tries to completely sink the ship which has been hit. It selects the most probable cell from the four surrounding cells, and then proceeds in that direction.

The AI uses the hunt mode and target mode repeatedly until all ships have been sunk.

2.2 Testing

For testing the code we wrote a code that generates random configurations of ships placed on the board. Using that code we tested the AI for 10000 random configurations and calculated the average of the number of moves it took to finish the game. We obtained an average of **44.9** moves.

2.3 Interactive game

We also wrote a text based interactive game for a human player to play against the AI. The code generates a random board for the opponent. The player has to place his ships by writing to a file.

3 Machine Learning

3.1 Introduction

The Battleship game can be considered as a supervised Machine Learning problem.Our goal is to achieve the following:Given the states of all the cells on the grid(hit,miss,unknown) we should be able to predict the next best possible move.**Neural Networks** are suitable to solve this problem since the number of input variables is large.This is a regression problem and not a classification problem since the neural network aims to calculate the probability of the presence of ship on a cell of the grid rather than directly classifying the best possible cell.The training data for the neural network is generated using a model similar to our probability model.

3.2 Neural Network

3.2.1 Introduction

Neural Networks are computing systems inspired by the biological neural networks that constitute animal brains. Such systems learn (progressively improve performance) to do tasks by considering examples, generally without task-specific programming.

3.2.2 Implementation

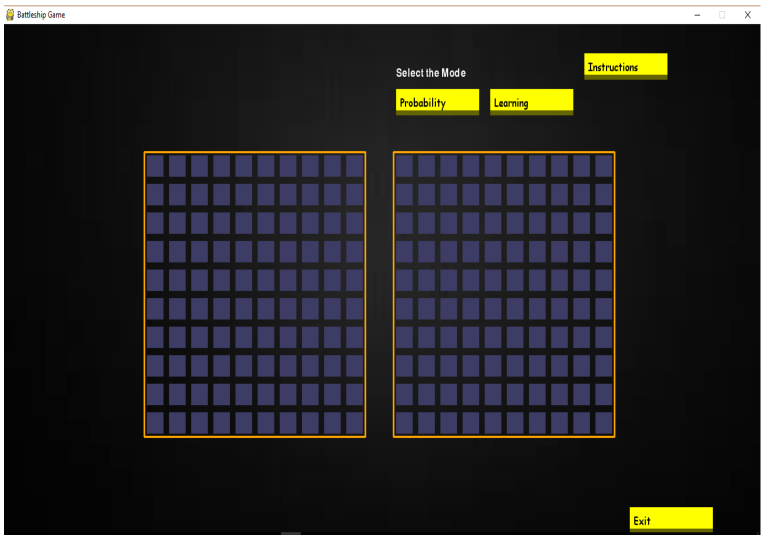
The task of the neural network is to calculate the probability of presence of ship on a particular cell of the grid. The neural network takes 100 input variables x_i's,where x_i=1,0,-1 corresponds to a hit,unknown,miss respectively.There are two hidden layers each consisting of 100 nodes.The neural network output y is a real number variable from the range [0,1] which represents the probability of the presence of a ship on that cell.There are 100 networks,one for each cell on the grid. After obtaining the probability values for every cell on the grid,the cell with highest probability value is selected as our next target.We used the *scikit-learn* library of python to implement the neural network.

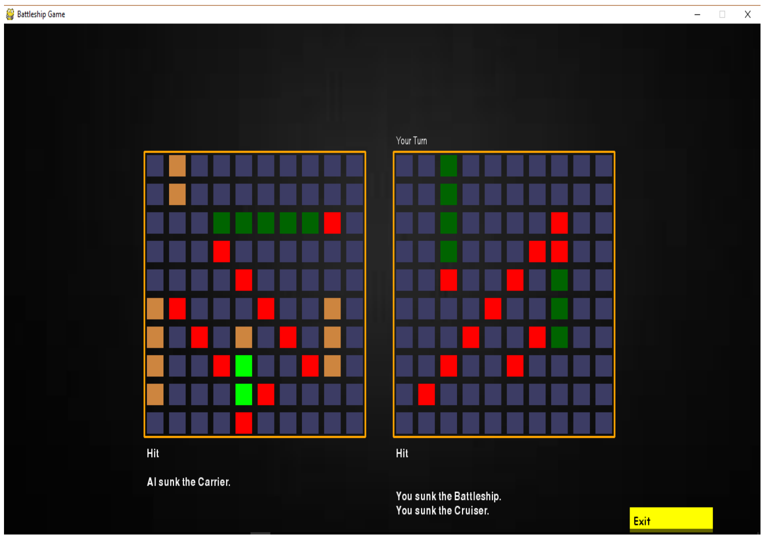
3.2.3 Training Database

We used a probability model to generate the database for our neural network.If we are provided with the states of all the cells on the grid then it is possible to calculate the probability values for all the cells by generating all possible ways in which ships can be placed on the grid satisfying the constraints.The problem with this method is that in the initial stages the number of configurations is very large and not feasible to compute. So we limited ourselves to 10000 random ship configurations and on that basis calculated the probability for every cell on the grid.We used the probability values generated by this method to prepare the training database.

4 Game GUI

We made a graphical interface for a human player to play the game against the AI. The GUI was made using the *pygame* module. The GUI includes both the AI modes - Probability and Learning, and the player can choose which mode to play against. The player can place his ships using the keyboard and the choose the cell to attack on the opponent's board using mouse. The game generates a random board for the player.





Hit

AI sunk the Carrier.

Hit

You sunk the Battleship.
You sunk the Cruiser.