**COMP813 Artificial Intelligence, Semester 2, 2023**

**AI Project Option A: Battleship Game**

*Finn Boyle, 18034590*

*Introduction:*

For this project, I have created a game of Battleship that is playable against a computer-controlled opponent. My initial plan for this project was to develop the game and train the AI player using reinforcement learning and Monte Carlo tree searching methods, however due to my initial game state implementation, the final product uses supervised learning, heuristic searching with elements of knowledge representation. This project was built nearly entirely by me alone, with some assistance in creating the AI from Coding Cassowary’s Battleship game tutorial series on YouTube **[1]**. I chose to build the project in Python, the game is text based and is played entirely using keyboard inputs.

*Problem Definition:*

To quickly define Battleship, it is a strategy game played without perfect knowledge of the game state. Players have two boards each representing friendly and enemy seas, hidden from the opponent, the friendly sea board is for placement of their own ships, and the enemy sea board is for recording shots taken against the opponent. Players place their five ships of differing lengths – five, four, three, three, and two – on their placement 10x10 grid. The aim of the game is to take turns “shooting” at co-ordinates on the opponent’s board. The opposing player will then confirm if the shot is a hit or a miss. If the shot is a hit, the shooter will mark on their enemy sea board the location of the hit with a red pin, and the opponent will mark their friendly board with a red pin. A white pin is used instead if the shot is a miss. The game continues with players taking turns “shooting” at their opponent’s board until one player has sunk every one of their enemy’s ships, this player is the victor. This is a slightly modified version of the official Hasbro Battleship game rules **[2]**. Rule modifications primarily include players not being required to state which ships have been hit and sunk, to increase difficulty for both the player and the AI, forcing the AI to search the board more rigorously.

The main problem with teaching an AI to play Battleship is making it learn where to place shots if it has not made any hits or has exhausted spaces surrounding a hit. The AI can of course search the board a random pattern, but this makes the game far easier for the player, as the AI will take many more shots than needed to find an enemy ship, often requiring the 100 shot maximum to find all enemy ships if firing completely randomly, as evidenced in a study by DataGenetics (2011) **[3]**. To make improvements, the AI can search the board in a checkerboard pattern, searching every other tile for a ship due to the smallest ship being of length two. DataGenetics **[3]** found that this method reduced the average number of shots down to approximately 60.

While nearby hit searching, random shooting and checkerboard searching were readily implementable, I wanted to expand on these methods with an AI that could determine optimal shot placement based off data collected from previously played games – i.e., the AI would be able to play against the human psyche, learning where humans are most likely to place their ships on a Battleship game board to increase their chances of winning the game. I needed a way to let the AI learn where humans will most likely place a ship on their game board, such that it could beat a human based on human unconscious thought processes.

*Motivation:*

I initially chose Battleship for my AI project because it was something I had not tried before yet felt that it was within my capabilities as a programmer. However, as I began researching the project, I began to feel that previous Battleship AI works did not touch on the specific problem I wanted to solve – making the AI learn from common human ship placements – many projects focused on single game learning, where the AI would make the most optimal shot based on where the previous shot landed rather than learning over time.

My final motivation, therefore, was to create an AI that learn from real historical data of human ship placements every game, every individual game improving its chances of hitting a ship on the player board sooner.

*Methodology:*

*Evaluation:*

*Discussion:*

*Limitations:*

*Future Work:*

*References:*

1. Coding Cassowary. (2022, April). *Battleship AI*. YouTube. <https://www.youtube.com/playlist?list=PLBLV84VG7Md8SgHlXuQXPMJLDvCaWVVQv>
2. Hasbro. (1990). *Battleship*. <https://www.hasbro.com/common/instruct/battleship.pdf>
3. DataGenetics. (2011, December 3). *Battleship*. Retrieved October 6, 2023, from <http://www.datagenetics.com/blog/december32011/>

Garbage to be removed:

Report

REFERENCE PROPOSAL, HOW DID THE PLAN CHANGE? WHAT’S NEW AND WHAT WAS REPLACED?

Code is sort of supervised learning!

Did approximately 40 ship placement sessions to train the system on where to take shots.

Coding Cassowary reference:

<https://www.youtube.com/playlist?list=PLBLV84VG7Md8SgHlXuQXPMJLDvCaWVVQv>

Future work:

* Ai tracks optimal places for it to place its own ships, learns over time.
* AI stops taking shots directly next to ships if it has already shot large amounts in the area, and moves on to new areas.

CHATGPT:

*Based on the provided code and context, your AI appears to incorporate elements of the following:*

*1. \*\*Search and Heuristic\*\*: Your AI performs a form of search in selecting the next shot location on the game board. It evaluates potential targets based on a weighted matrix and considers various strategies. While it doesn't explicitly use Minimax Search, Alpha-Beta Pruning, or Monte Carlo Tree Search, it does employ a heuristic approach to selecting shots.*

*2. \*\*Supervised Learning\*\*: Your AI uses a form of supervised learning to inform its decision-making process. It maintains a `weighted\_board` that accumulates data about common ship placements by the player. This information is used to make informed choices about where to shoot.*

*3. \*\*Knowledge Representation\*\*: Although your AI doesn't explicitly represent knowledge in a formal knowledge representation language (e.g., propositional logic or first-order logic), it does represent knowledge in the form of the `weighted\_board`. This board represents probabilistic knowledge about likely ship placements.*

*However, your AI does not appear to explicitly incorporate elements of:*

*1. \*\*Game Theory and Nash Equilibrium\*\*: Game theory and Nash equilibrium involve modeling strategic interactions between rational agents. Your AI follows predefined rules and heuristics but doesn't explicitly model the strategic decisions of an opponent or reach Nash equilibria.*

*2. \*\*Reinforcement Learning and Deep Neural Networks\*\*: Your AI doesn't use reinforcement learning or deep neural networks to learn optimal strategies through interactions with the game environment or opponent.*

*3. \*\*AI Safety and Explainable AI\*\*: There is no evidence in the provided code that the AI considers safety or includes mechanisms for making its decisions more explainable or interpretable.*

*So, in summary, your AI primarily incorporates elements of heuristic search and supervised learning but does not explicitly involve game theory, reinforcement learning, deep neural networks, or AI safety considerations.*