# Laboratory of Artificial Intelligence and Data Science

## *Project 2 – Develop an Alpha Zero Game Player*

November 10th, 2023

## 1. Introduction

The objective of the Laboratory of Artificial Intelligence and Data Science (Lab AI & DC) course is to provide students with software development methodologies, AI and DC projects, teamwork, and communication through the implementation of projects designed for this purpose. Students should apply the knowledge obtained from the courses from previous years and research methodologies to solve the problem.

In this second project, the students will create a AlphaZero Algorithm based Game Player and test it in five different scenarios: Attaxx game (with empty 4x4 and 6x6 board sizes and generic 4x4 to 6x6 board with flexible initial configurations) and Go (with 7x7 and 9x9 board sizes) and compete in a final event for the title of best game playing agent.

## 2. Context

AlphaZero, a creation of DeepMind, an AI research firm, was designed to master chess, shogi, and go. This program adopted a methodology similar to AlphaGo Zero.

On December 5, 2017, DeepMind introduced AlphaZero through a preprint paper. Remarkably, after just 24 hours of self-training, it reached an extraordinary skill level in these games, surpassing champion programs Stockfish, Elmo, and AlphaGo Zero (three-day version). This achievement was bolstered by the use of specialized tensor processing units (TPUs), tailored for Google's programs. AlphaZero's learning was exclusively through self-play, employing 5,000 first-generation TPUs for game generation and 64 second-generation TPUs for neural network training, all operating simultaneously. It did not rely on traditional chess opening books or endgame tables. Within just four hours, AlphaZero surpassed the Elo rating of Stockfish 8, and after nine hours, it outperformed Stockfish 8 in a 100-game match under time constraints, recording 28 wins, no losses, and 72 draws. This advanced algorithm operated on a single machine equipped with four TPUs.

The findings on AlphaZero were published in the "Science" journal on December 7, 2018. However, the AlphaZero program itself was not released to the public. In 2019, DeepMind released a new study about MuZero, an algorithm that extends the capabilities of AlphaZero. MuZero can play both Atari and board games without pre-existing knowledge of the game's rules or structure.

## 3. The Games to Play

A board game is characterized by the type of board and tiles, the rules of movement of the pieces (operators/possible moves) and the finishing conditions of the game with the respective score. In this work, the aim is to implement the Attaxx and Go games for two players.

Attaxx is an abstract strategy board game and also a strategy video game that was published in arcades by The Leland Corporation in 1990 with the name Ataxx. It also appeared the same year as Spot: The Video Game and later as the Microscope Puzzle from the 1993 CD-ROM game The 7th Guest. The game was invented by Dave Crummack and Craig Galley in 1988 and was originally called Infection. It was first programmed on Amiga, Commodore 64, and Atari ST. Although Ataxx was originally sold as a video game and not with a physical game board it is also an abstract strategy board game that may be played with a physical board. Several online Attaxx online games are available such as for DosBox. Figure 1 shows the original game and an Attaxx online game.

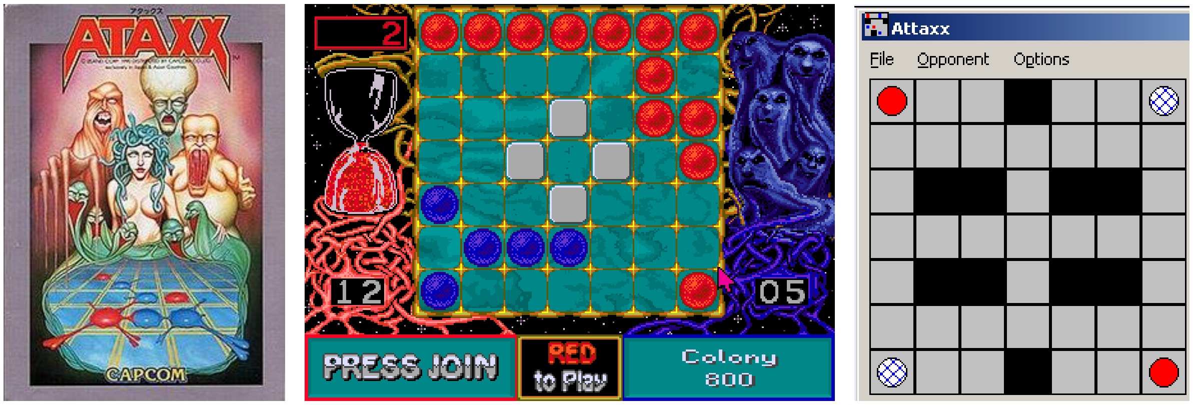

*Figure 1: Attaxx game: Original version and a MS Dos implementation*

Attaxx is a game that involves play by two players originally on a seven-by-seven square grid although it may be played in a board of any dimension. The object of the game is for the player to make its own pieces to constitute a majority of the pieces on the board at the end of the game, by converting as many of their opponent's pieces as possible.

Typically, each player begins with two pieces (although several variants exist), white and black, for the first player and second player respectively. The game starts with the four pieces on the four corners of the board (although again this may be flexible), with white in the top left and bottom right and black on the other two. White moves first as it is typical in board games.

During their turn, players move one of their pieces either one or two spaces in any direction. Diagonal distances are equivalent to orthogonal distances, i.e. it is legal to move to a square whose relative position is two squares away both vertically and horizontally. If the destination is adjacent to the source, a new piece is created on the empty departure square. Otherwise, the piece on the source moves to the destination. After the move, all the opponent player's pieces adjacent to the destination square are converted to the color of the moving player. Players must move unless no legal move is possible, in which case they must pass.

The board configuration may be altered by having certain squares filled (not playable by either player), using different board sizes or having distinct initial configurations instead of the typical 4 corner pieces start. The game ends when all squares have been filled or one of the players has no remaining pieces. The player with the most pieces wins the game. A draw may occur when the number of squares is even, either from non-playable squares or nonstandard sizes having an even number of squares. Some versions also implement the threefold repetition rule from chess.

The second game is Go, a classic two-player strategy game where the objective is to encircle more territory than the opponent. Originating in China over 4,500 years ago, it is reputed to be the world's oldest continuously played board game. A 2016 survey by the International Go Federation, which has 75 member nations, revealed that around 46 million people globally are familiar with Go, with more than 20 million active players, predominantly in East Asia.

The game uses black and white stones as playing pieces. Players alternate placing these stones on the vacant points (intersections) of the board. Once a stone is set, it cannot be moved. However, stones can be captured and removed from the board if they are completely surrounded on all orthogonally adjacent points by the opponent's stones. The game ends when both players decide not to make further moves. The winner is determined by tallying each player's captured territory, stones, and komi (extra points given to the player with white stones for playing second).

Typically, Go is played on a 19×19 line grid, forming 361 points. In ancient China, Go was one of the four essential arts of a cultured aristocratic scholar. The earliest known mention of the game is in the historical annal "Zuo Zhuan" (4th century BC).

Despite its straightforward rules, Go is a game of profound complexity. Compared to chess, Go features a larger board offering wider strategic possibilities, typically longer games, and a greater number of potential moves per turn. The number of possible legal positions on a Go board is about 2.1×10^170, a figure vastly exceeding the estimated 10^80 atoms in the observable universe.

## 4. Work to Develop

- You should prepare an Alpha Zero algorithm or similar solution (for example the improved MuZero version). The agent may be developed in any programming language or combination of languages.
- You should create two games: Attaxx and Go.
- You should create a simple graphical interface for each game agent enabling it to play the game with a human and with another agent.
- You should implement a socket-based communication protocol enabling the agent to communicate with another agent. The protocol will use plain text commands and answers, such as: MOVE x,y (to indicate a move).
- You should train five agents to play Attaxx (with 4x4, 6x6 and flexible board sizes) and Go (with 7x7 and 9x9 board sizes).
- You should evaluate each agent against human players and other agents.
- You should share your solution in a gitlab/github (share the link in moodle by the second practical class: week of 20th of November);
- The solution should be delivered in moodle by January 4th, 2024, at 23:59:59;
- Students must present their work at January 15, 2024 in the morning.
- Students should participate in the final agent competition at January 15, 2024 in the afternoon.

- The final code solution should be submitted as a notebook documented, explaining your decisions and discussion about the results obtained.
- One-page document documenting the main features of the work.
- Auto-evaluation file provided by Professors.

## 5. Workflow

The workflow will be the following:

- **Game Implementation.** Students should first implement the games (starting by the attaxx game or implanting the two games in parallel), ensuring they can represent the game state, determine legal moves, and check for terminal states (wins, draws).
- **Communication Protocol.** Students should implement the simple communication protocol proposed and test if they may put their agents playing against agents developed by other groups.
- **Neural Network Model.** Students will design a simple neural network that will serve as the policy and value network for the game. This may be different for different variations of the game depending on the complexity of the game and board size.
- **Monte Carlo Tree Search (MCTS).** Implement MCTS as the primary mechanism for exploring game states. The neural network guides the search and evaluates the positions.
- **Self-Play Training.** Train the neural networks using games played against itself. This is similar to how AlphaZero trains, iterating between MCTS-guided gameplay and network training.
- **Competition.** After a certain training period, pit the students' agents against each other in a tournament-style competition in the 5 variations of the games.

## 6. Evaluation Criteria

Your work will be evaluated on the following criteria:

- **Quality of Implementation (25%).** Clear code, appropriate and efficient implementations of the games, the neural network, the MCTS algorithm and the communication protocol.
- **Training Process Implementation (25%).** Given the limitation of training on laptops, students might focus on how efficiently they can train their agents. They may also experiment with different neural network architectures. Techniques to enhance training speed or efficient use of data can be a part of the challenge. Also they may introduce concepts like board symmetry to reduce the state space or techniques to save and reuse search trees from MCTS.
- **Documentation and Presentation (25%).** Quality of the Report, Presentation and documentation:
- **Performance in the Tournament (25%).** The performance of the agents against each other and the tournament results.

Work as a team: The time is very short, our suggestion is that you distribute tasks well amongst the team;

## Bibliography and Links

The following links contain information about the work and pointers to papers that describe AlphaZero and related concepts:

- https://en.wikipedia.org/wiki/AlphaZero

- https://github.com/suragnair/alpha-zero-general
- https://web.stanford.edu/~surag/posts/alphazero.html
- http://neuralnetworksanddeeplearning.com/
- https://web.archive.org/web/20180629082128/http://mcts.ai/index.html
- https://www.codingame.com/playgrounds/58137/alphazero-like-implementation-for-oware-abapa-game-codingame/alphazero-like-implementation-for-oware-abapa-game
- https://tmoer.github.io/AlphaZero/
- https://joshvarty.github.io/AlphaZero/
- https://www.youtube.com/watch?v=wuSQpLinRB4&ab_channel=freeCodeCamp.org
- https://www.youtube.com/watch?v=WXuK6gekU1Y&t=305s&ab_channel=GoogleDeepMind
- https://en.wikipedia.org/wiki/Go_(game)
- https://en.wikipedia.org/wiki/Ataxx