NAME: SHRINATH HIMANE

ROLL NO: 60

REG NO.: 11802347

**Final Report: AImojoBot VS Player (Connect4)**

The following is a brief 2 nd year Final report summarizing my

Complete work as of 31st March 2020. It is based on the Min-Max Algorithm and is

built in Python language.The packages and libraries used are discussed further in the report.

**Table of content:**

|  |  |  |
| --- | --- | --- |
| 1 | Abstract/Overview | Summary of the Project. |
| 2 | Introduction | Formal presentation of the concept , usefulness of AI in project. |
| 3 | Motivation | Reasons to create a artificially intelligent bot for a game. |
| 4 | Aim | Intention of achieving fully predictive and calculating bot . |
| 5 | Objectives | Goal to develope a bot using AI , to predict best move according to situation. |
| 6 | Scope | the extent of the area that this project is relevant with. |
| 7 | Related Work | Description of the existing work in regards with this project. |
| 8 | Implementation | Foreseeing to what extent this project can be taken and discussing newly introduced work in regards with this project. |
| 9 | Libraries Used | Mentioning all python libraries and packages used for this project. |
| 10 | Reference | Links and mentions of websites and books that helped to achieve the fully working state of the project. |

**Abstract/Overview:**

A gameplay example (right), shows the first player starting Connect Four by dropping one of their yellow discs into the center column of an empty game board. The two players then alternate turns dropping one of their discs at a time into an unfilled column, until the second player, with red discs, achieves a diagonal four in a row, and wins the game. For games where the board fills up before either player achieves four in a row, then the games are a draw.

Connect Four has been solved with brute-force methods, beginning with John Tromp's work in compiling an 8-ply database (February 4, 1995). The artificial intelligence algorithms able to strongly solve Connect Four are minimax or negamax, with optimizations that include alpha-beta pruning, move ordering, and transposition tables. The code for solving Connect Four with these methods is also the basis for the Fhourstones integer performance benchmark.

In this project I have also developed a bot called as AI mojobot which uses the minmax algorithm along with alpha beta pruning which has been taught in our course , also scoring method has been used inorder to give value to each move depending on likeliness of the move resulting in victory of the bot.

Various python libraries have also been used inorder to create an interface like pygame . The game would only take input through trigger keys , which in this case are all designated through the mouse itself.

**Introduction:**

The project mainly applies Mini-Max Algorithm

with Alpha-Beta pruning to play Connect4 game. Mini-Max would be

used to compute minimax values of each following node, and

backtracking would be used to find out the best move . The Mini-Max

Algorithm would predict the state of the board ahead of time in order to

make the best move.

Mini-Max search algorithm is good at predicting its opponent’s move

and then beating it, but the runtime of minimax would always be an

issue. In order to shorten its runtime, this project applies alpha-beta

pruning to minimax. Since time is too limited for minimax to look at

every node in the game tree, the main goal of Alpha-Beta pruning would

be to increase minimax’s efficiency by pruning any unnecessary move

that has no influence on making the final decision .

However, even though runtime would be shortened, it would still take

too much time for Mini-Max to reach peak optimization levels in some

cases. For example, when playing Connect4 game, system would lack the

ability to search the bottom value of game tree in an optimal speed. In

order to speed up the progress of making the best move, heuristic

functions would be applied in this project. Heuristic functions determine

which branch to follow by sorting the alternatives in each branch-step

based on available information . Since minimax picks the highest value

that heuristic generates, heuristic becomes so essential that every subtle

change in it can alter the final outcome of the move. Therefore, this

project would use scoring technique for each node to choose the best

path.

**Motivation:**

The reason I took up this project was because MinMax algorithm as well as alpha beta pruning is the part of the syllabus and by this I will not only have the knowledge of theory part of both but also a strong grasp of the practical implementation of the same. Also, I could put to use the python programming skills used in the game such as pygame .

The main algorith used in the project ie. Minimax , is a decision rule which simulate the decision of a player to find the optimal move for the player. It assumes that the opponent will play with optimal choice too. There are two actors in the Minimax. It’s maximizer and minimizer. The maximizer will search the highest score as possible and the minimizer will search the lowest score as possible. Each players will see the opponent’s move best move and get the best one for him/her. Usually, the searching is represented in tree structure data.

I am also genuinly interested in the functioning of AI and its algorithms , which is also why I tries to make the project all by myself .

**Aim:**

The aim is mainly to create a fully functional bot for playing against a user in the connect 4 game abiding the rules of the game . Also using pygame to create the interface of the game . The bigger Aim is also to learn how an AI can calculate the not just the next best step according to the scoring method but also will create and traverse an entire 21 level tree of possible sequence of moves just to make one single move. The Aim is also to grasp this functionality of the the and try to apply the same not just in other games but also other fields in the world , such as predictive analysis , which will heavily help in saving time , money , workforce and energy.

**Objectives:**

**1}** Properly implementing minmax algorithm to the working of this project such that the AIbot will absolutely create and traverse the tree completely inorder to calculate the next best step not just to MAXIMIZE the chances of the AIbot winning but also MINIMIZE the chances of the user to come close to a possible winnning move.

**2}** Using alpha beta pruning whiuch is a is a search algorithm that seeks to decrease the number of nodes that are evaluated by the minimax algorithm in its search tree.

**3}** Developing an interface which concludes of a connect4 playing board which is actually a matrix, also the checkers for both the players I.e the AI bot and the user.

**4}** Using the sys library inorder to take the necessary inputs from the user through the mouse , the user only needs to drop his own checker int he appropiate column on the board.

**Scope:**

1}Look-ahead search (a.k.a. tree search)-

Connect4 is a board game that involves two players, who take turns to place checkers on a board . Points are given for forming sequence of checkers in one line. One notable difference with a game like Chess is the number of possible moves one can make, which in Connect4 is of the order of 10 times that in Chess.

At the heart of Chess and Connect4 programs is an algorithm that examines possible sequences of moves (player A plays this move, then player B may play these moves, then player A may play those moves, etc.) and evaluates who has the advantage at the end of these sequences. This is referred to as tree search, where nodes of the tree contain representations of the board and each branch corresponds to a possible move (which makes a lot of branches per node). For this approach to work, one needs a good way of evaluating positions at leaf nodes, and for the search to be efficient, one needs a good way of prioritizing branches to explore.

2} Long-term decision making-

Connect4 is a game where players practice long-term decision making. It involves choosing from lots of options based on an understanding of complex situations, and it involves long-term planning (long sequences of actions that impact the situation) in an adversarial setting. According to Troy Anderson, author of The Way of Connect4, many leaders in Asian business, politics and the military apply the game metaphorically to “maximize their time and resources, seize the initiative, adapt to change and compete in an established market.” Connect4 is also part of the curriculum in some Western business schools.

3} Challenges in applying Connet4’s technology in the real world-

Advancing the technology is crucial, but another part of the difficulties in tackling real-world problems is in modelling the world into a format that the machine can work with in an unsupervised way. This involves listing out what makes up the state of the world, which actions can be carried out, and then representing states and actions (the equivalents to Connect4’s boards and moves) in the machine’s memory.

These representations can be at a very low level and poorly structured, as they are in Connect4 (i.e. our matrix with ternary values). Deep learning can be used to extract useful representations.

You have to choose a type of neural network to work with (e.g. for Connect4 they chose convolutional networks), an interconnection pattern between the different layers of neurons and a size (number of layers, number of neurons).

You need training data, i.e. examples of previous states of the world and actions/decisions taken by humans. Deep learning needs a lot of data to produce good quality networks. Part of the data may come from self-play of the algorithm against itself, but you still need some data to get started.

In reality, you’re often not sure of the state you’ll arrive in after taking an action (or, to put it differently, in the consequences of your decisions). You would need to account for (and to model) uncertainty in the look-ahead search.

**Related Work:**

Currently , Connect4 is not considered to be a competitive game and is also playe indoors just as a part of refreshment.Hence , no current of past work has been done regarding this but that does not mean there is no future of this project and its developement . Also , in the near future if it becomes a competitve game then the players can use this project as a measure to play as rationally as possible . Since the game uses scoring technique , we can see if we make the best move and in return check with the AIbot.

Also, a lot of work can be done regarding this project in the field of other 2 player games and also new 2 players games can use the algorithm and the scorng methods used .

**Libraries/Packages Used:**

**1}NUMPY**

The numpy library is used to create a 6\*7 matrix of zeroes which is also

flipped inorder to make the checker drop directly down to the last

square of a column.

**2}RANDOM**

This module implements pseudo-random number generators for various distributions.

**3}PYGAME**

To develope the interface for the entire game , which consists of squares

, checkers (different for both players), and the board.It is specifically intended to help you make games and other multimedia applications. Built on top of the highly portable SDL (Simple DirectMedia Layer) development library, pygame can run across many platforms and operating systems

**4}SYS**

To recieve the input , and to designate keys to play the game with each

of their functions. Also , the sys module provides information about constants, functions and methods of the Python interpreter. dir(system) gives a summary of the available constants, functions and methods. Another possibility is the help() function. Using help(sys) provides valuable detail information.

**5}MATH**

These functions cannot be used with complex numbers; use the functions of the same name from the cmath module if you require support for complex numbers. The distinction between functions which support complex numbers and those which don’t is made since most users do not want to learn quite as much mathematics as required to understand complex numbers. Receiving an exception instead of a complex result allows earlier detection of the unexpected complex number used as a parameter, so that the programmer can determine how and why it was generated in the first place.

**Reference:**

1} https://www.youtube.com/watch?v=l-hh51ncgDI - for minmax

2} https://www.javatpoint.com/ai-alpha-beta-pruning - for alpha beta pruning

3} https://www.youtube.com/watch?v=FfWpgLFMI7w - for pygame