Julian Burgos, Matt Gashaw, Joseph Jubilee, Samuel Kim, and Trevor Lorin

Professor Sujeong Kim

CMSC421

6 May 2024

AI Connect-4 Opponent

Literature Review

Nasa et al. [1] discusses how the mini-max algorithm can be applied to Connect 4. The two players take turns, where one player tries to maximize the score and the other player tries to minimize the score. Based on the values assigned to different board states, the players will choose the move that results in either the highest or lowest score, depending on which player's turn it is. In our project, we worked to analyze the board states and allow the AI to choose the action with the highest reward. Intuitively, this involves considering heuristics such as the number of meaningful 2 or 3 in a row and the number of possible moves.

Schneider et al. [2] discuss the capability of neural networks in learning the game of Connect 4, allowing for an AI that can smartly make moves based on the board state. The article describes the structure of a neural network used to learn Connect 4 and how the number of neurons used affects the learning of the network. The researchers found that by choosing high-quality games for the neural network to learn from, it was able to play Connect 4 more effectively than other methods. After learning about the capability of neural networks to solve complex problems, we were interested in applying it to Connect 4.

Project Introduction

Connect-4 is a classic two-player strategy game played on a grid board. The two players hold game pieces, typically red and blue discs, and alternate turns, dropping one of their discs

into any of the columns. The disc will fall to the lowest unoccupied space in that column. The goal is to be the first player to connect four of your colored discs in a row horizontally, vertically, or diagonally. As the game progresses, players aim to not only connect their discs but also block their opponent from forming their line of four. This requires strategic placement of discs to both build your line and disrupt your opponent's plans. Depending on how you play it, Connect-4 can be a zero-sum game where one player has to win while the other has to lose, or it could result in a tie when the board completely fills up without a winner.

Connect-4 presents a complex problem for an AI to solve due to its branching complexity. With numerous possible moves available at each turn, the game's grid creates a vast search space for potential game states. As players make moves, the number of possible future positions escalates rapidly, which would require significant computational resources to explore all outcomes. Furthermore, Connect-4 demands both offensive and defensive strategies, necessitating the evaluation of one's potential winning moves alongside those of the opponent. This combination of an expansive search space and strategic plays makes Connect-4 a formidable challenge for computers to solve optimally, requiring advanced algorithms and computational capacity to navigate effectively.

Creating an AI to be a rational adversary in Connect-4 is a valuable problem to solve for several reasons. Firstly, developing an AI opponent can enhance the player experience by providing challenging gameplay and opportunities for skill development. Additionally, solving Connect 4 with AI has practical applications beyond the game itself. The same algorithms and approaches used to create a strong Connect 4 AI can be adapted and applied to other real-world problems that involve decision-making under uncertainty, such as logistics, resource allocation, and scheduling.

Solution

For our solution to the problem, our AI adversary is meant to drop its disc into the column that has the highest reward. In the offensive sense, the AI can choose the optimal column that would allow it to win, for example, if it is one disc away from a Connect-4. If the AI is less than one move away from winning, it will attempt to build off of an existing line of discs or create a new line. In the defensive sense, the AI can determine if the human player is one move away from winning and place its disc to prevent a 4-in-a-row line.

Much of the way we approached the solution was influenced by Homework 3. Using Pygame for the user interface, our solution works through a neural network we defined using the PyTorch library. We decided not to use a pre-trained model because we wanted to maximize the customization of our network to our needs. Moreover, we thought we'd get a firmer grasp of the concepts and material if we were to design our own network instead. We defined a class "NN" that inherits from "nn.Module", which is the base class for all neural network modules in PyTorch. We defined our neural network to have four fully connected linear layers that, when used consecutively in a forward pass with the Rectified Linear Unit (ReLU) activation functions in between, reduce an input of size 42 into 3.

This network classifies game board states into one of three categories: negative 1 meaning the human wins, positive 1 meaning the AI wins, and 0 if it is a tie. Using a cross-entropy loss function and Adam optimizer with a customizable learning rate (0.5 for a dumb model, 0.052 for an average model, 0.01 for an expert model), we trained our model through 5 epochs with a batch size of 64.

We also implemented a minimax function in which the AI can choose the most optimal action after using a depth-limited search on the adversarial game tree at the depth respective to the chosen difficulty (1 for a dumb model, 5 for an average model, 15 for an expert model). Terminal utilities are replaced with an evaluation function for non-terminal positions. This allows our AI to make long-term strategic decisions.

The strength of our AI is that it proves to be a competitive opponent as it can make immediate moves that will increase its chances of winning as well as block off any winning moves made by the human player. Its weakness, however, is more on the long-term, strategic end. It is unable to predict and block long-term strategies made by humans as it is unrealistic to search down to the leaves of an adversarial game tree. Due to the depth-limited search, the guarantee of optimal play is gone.

Results and Evaluation

For each of the 3 difficulty modes, we played 50 games against the AI to get a measure of how the win rate of the AI changes between difficulties. For each game, we'd also change our playing strategies against the AI so we couldn't get repeated wins off of the same set of moves. When it came to each difficulty, the "Low" AI had a 14% win percentage, the "Mid" difficulty AI had a 22% win percentage, and the "Hard" AI had a 46% win percentage.

During evaluation, we faced an unexpected outcome where if our strategy consisted of placing consecutive discs along the bottom row, the AI would not be able to counter it unless, by chance, the AI placed its piece in the center column, making a connect 4 impossible for a 7 column game board. As a result, the player would be able to quickly place 4 in a row along the bottom row without getting blocked. We hypothesize that this is due to the NN prioritizing connecting 4 of their own pieces rather than blocking a connected 4 from the human.

For the qualitative measures of the AI, we found that for the mid and expert difficulties, the AI can immediately identify situations where it is able to make a winning move, if it exists. Additionally, it identifies situations where it is able to set up 3 in a row such that the opposing player is unable to immediately block it, increasing its chances of winning. Through our testing, we also noticed that the expert difficulty AI was making more long-term strategic plays whereas the lowest difficulty AI was only making the moves that would have the highest immediate reward. However, we did find cases where the AI prioritized moves that set up 3 discs in a row over moves that block the opponent's fourth in a row, resulting in a loss for the AI. Moving forward with this, we would like to train our neural network to additionally consider strategies such as blocking the opponent's third or fourth in a row and knowing when to make an offensive versus a defensive play.

Comparison of Initial Goals to Results

Our initial goals for this project were to create an AI to play an engaging game of Connect 4 against the player and to learn more about the capabilities of neural networks and mini-max algorithms in solving complex problems such as Connect 4. Based on the overall results of our final project, we reached our goals and now have a stronger understanding and interest in these fields.

After building and testing our AI Connect-4 opponent, we noticed that the AI, while impressive, still has a few flaws. We noticed that the AI would not perform nearly as good during the beginning of the game in comparison to the end game. This is likely due to the fact that there are fewer possible good moves and thus more predictability toward the end of the game. In addition to this, the AI often seemed to prioritize offensive plays over defensive plays, causing it to favor moves that make 3 or 4 in a row as opposed to moves that block the opponent's 3rd or

4th in a row, which can lose the AI the game. While we do see these areas of improvement in how our AI analyzes the board state and picks its move correspondingly, we now have a stronger sense of direction on how we would move forward differently. For example, we can consider using the neural network to directly output a number of the columns to place the disc in, saving time from the mini-max algorithm and using it towards considering more heuristics that are important for the strategies of Connect 4. Additionally, using a second AI to play against the first AI would give us important insights as to how the AI responds to certain types of moves and the comparative strength of the different difficulties of the AI.

Member Contributions

Each team member contributed equal shares throughout the project, with lots of collaboration through each step. We met together to collaborate on brainstorming ideas and writing code together in order to help each other debug and ensure contributions from all members. Matt set up the GitHub page and was in control of approving and pushing the code to the repo. Trevor was in charge of organizing documents, writing the proposal, and midterm check-in after the brainstorming sessions. Julian was in charge of testing the code and gathering results to evaluate the AI. Sam headed the poster creation and Joseph was in charge of debugging when we ran into issues. We strived to collaborate as much as possible to make each aspect a group effort which was made easy through our close living proximity.

Post-Presentation Updates:

During our presentation, we received excellent feedback from all of our TA's reviewers. Over the following two days, we implemented their suggestions. The first recommendation was

to have an unbiased party play against our AI agent. Accordingly, we invited some friends to test the bot, and we collected new data from these sessions. Additionally, a significant issue with our agent was its inability to play both defensively and offensively. After making a few adjustments to the logic, most notably introducing a decay factor, we enhanced the agent's capabilities. Now, when set to the hard difficulty, our agent is undefeatable—at least by our friend. After implementing the recommended changes, our AI's performance improved across different difficulty settings: it had a 16% win rate on low difficulty, a 32% win rate on medium, and an unbeatable 96% win rate on hard difficulty.

Citations

1. Nasa, Rijul, et al. “Alpha-Beta Pruning in Mini-Max Algorithm -An Optimized Approach for a Connect-4 Game” IRJET, 2018.

2. Schneider, Marvin O., et al. “Neural Connect 4 – A Connectionist Approach to the Game” IEEE Computer Society, 2002.