1 Game Search (Minimax) (5 Points)

In this exercise, you will use the Minimax algorithm to play Tic-Tac-Toe[1] on a standard 3x3 grid. The game is played by two players Max and Min who alternatingly mark a vacant square of the grid with their respective token (X for Max, O for Min). The player who first builds a horizontal, vertical or diagonal line of three of their respective tokens wins immediately and the game ends. If the board is completely filled and no player achieved such a line, the game ends in a draw.

The terminal states in which Max wins are assigned a utility value of +100, the states in which Min wins a value of −100 and the states in which the game is drawn a value of 0.

O	X	X
X		O
		O

Figure 1: Tic-Tac-Toe game state for Exercise 1. Max to move.

1. Draw the Minimax search tree for the Tic-Tac-Toe game state depicted in Figure 1 (cf. *Alpha Go Part 3, slides 11–13*) where it is Max to move. Draw each search node as a depiction of the corresponding board state, as in slide 11. Annotate terminal states with their utility value as specified above and annotate the nodes of the search tree with their Minimax value as computed in the algorithm (i.e., the value returned by Min-Value(s) or Max-Value(s), whichever is called). What is the play for Max according to the result of Minimax? (2.5 Points)

2. Next, consider the following evaluation function h defined as

$$h(s) = \text{Count}_{\mathsf{X}}(s) - \text{Count}_{\mathsf{O}}(s)$$

where $\text{Count}_{\mathsf{X}}(s)$ (and analogously $\text{Count}_{\mathsf{O}}(s)$) is the number of rows, columns and diagonals that contain exactly two X tokens. For example, for the state depicted in Figure 1, we have $\text{Count}_{\mathsf{X}}(s) = 1$ and $\text{Count}_{\mathsf{O}}(s) = 2$ and therefore $h(s) = 1 - 2 = -1$.

Draw the search tree generated by the Minimax algorithm when applied with a depth limit of 2 for the Tic-Tac-Toe game state depicted in Figure 1. Use the evaluation function h to estimate the utility of the non-terminal (!) cut-off states, i.e. non-terminal states encountered at recursion depth 2[2].For the terminal states, use the utility function. As in the previous task, annotate the nodes of the search tree with their Minimax value. Here, annotate non-terminal (!) cut-off states with their h value. What is the play for Max according to the result of Minimax? (2.5 Points)

2 Minimax Implementation (5 Points)

In this exercise, you will implement the Minimax algorithm. See `assignment07.ipynb` for details.

3 N-gram language model (10 points)

Assume a speech recognition system receives a voice utterance "*I can see the word ...*".
Based on the sound signals, the system predicts that the last word could be *"here"* or *"hear"*, which are homophones (equal-sounding words).

The system incorporates a bigram language model that is trained from the following data.

[1] https://en.wikipedia.org/wiki/Tic-tac-toe
[2] The initial recursion depth is zero, not one

<start> you are here <end>
<start> I can hear your voice <end>
<start> I wrote the word here <end>
<start> she crossed the word here <end>
<start> Sam can see the mistake <end>
<start> he cannot even spell the word hear <end>
<start> I can hear the word <end>

a) Write down the probabilities of all the bigrams occurring in the utterances "*I can see the word here*" and "*I can see the word hear*" from the training data of the language model. (hint: all sentences start with <start> and end with <end>.)

(5 points)

b) Explain whether the system will recognize the utterance as "*I can see the word here*" or "*I can see the word hear*" when the system takes the language model into account.
Show the calculation for the probabilities of the two alternative sentences.

(3 points)

c) Would a trigram model trained on the same data recognize the utterance similarly as the bigram model? Briefly explain.

(2 points)

Submission Details:

Upload your submission to our CMS until *January 22, 2023 at 23:59*. Late submissions will not be graded! The submission should be uploaded by exactly **one** team member. Make sure that your submission contains the name and matriculation number of each team member. Submit your solution in **separate *pdf* files** for exercises 1 and 3, and a **separate *ipynb* file** for exercise 2.