Games and Computation Homework #10: Abstraction and Neural Networks

Answer these questions within the HW #10 Moodle quiz:

Dice Squares

Consider this simple, original dice game created for this exercise: You have 9 dice and a 3x3 grid for placing these dice. Each turn, you roll a single die and commit it to a grid cell. Once you have filled the grid with all 9 dice, you score the grid as follows: For each row and column, classify each set of three dice as either a straight (three consecutive values appearing in any order), a triple (three of a kind), a pair (two of a kind), or a null (three different values not forming a straight). Let the straight score S points, the triple score T points, the pair score P points, and the null score 0 points. Suggest an abstraction of the Dice Squares game state in the style of the presented approach to Parameterized Poker Squares. In particular, how would you abstract a board with the following values (left-to-right, top-to bottom): row 1: 3, (blank), 5; row 2: (blank), 1, (blank); row 3: (blank), 1, 6. (1) First, describe how you would build the abstraction string for each (possibly partial) dice set. (2) Then perform the translation of the example board to such dice set abstractions. (3) Finally, name the reinforcement learning algorithm used by our Parameterized Poker Squares player to learn estimates of expected future grid scores using such abstractions.

Neural Networks

For each of the following Boolean functions, either identify the correct perceptron weights and threshold to compute the function, or indicate that no such weights exist.

NAND

Function NAND (NOT (A AND B)): The output is 1 (true) if and only if at least one input is 0 (false).

- ☐ W_A = W_B = -1, t = -0.5
- ☐ W_A = W_B = 1, t = 1.5
- ☐ Linearly inseparable
- ☐ W_A = W_B = -1, t = -1.5

XOR

Function XOR (“exclusive or”: A OR B but not A AND B): The output is 1 (true) if and only if one input is 1 (true) and the other is 0 (false).

- ☐ Linearly inseparable
- ☐ W_A = W_B = 1, t = 0.5
- ☐ W_A = W_B = -1, t = -0.5
- ☐ W_A = W_B = -1, t = -1.5

NOR

Function NOR (NOT (A OR B)): The output is 1 (true) if and only if both A and B are 0 (false).

- ☐ W_A = W_B = -1, t = -1.5
- ☐ Linearly inseparable
- ☐ W_A = W_B = -1, t = -0.5
- ☐ W_A = W_B = 1, t = -1.5

Learning Reverse Implication 1

Function A OR (NOT B) (a.k.a. “reverse implication”, A ← B) is defined as follows: The output is 1 (true) if and only if A is 1 (true) or B is 0 (false). Let us suppose that we have a perceptron with a step function from 0 (false) to 1 (true) at a threshold of 0. We can make an *effective* threshold of t by having a fixed input Bias of 1 and a weight W_Bias from that input equal to –t. So let us for this example have fixed input Bias = 1, and 0(false)-or-1(true) Boolean inputs A and B. There are then three weights W_Bias, W_A, W_B. Let learning rate alpha = 0.2, W_Bias = -0.260, W_A = 0.081, and W_B = 0.370. If we train the weights with one step of the perceptron learning rule with inputs A = 1 and B = 0, the resulting perceptron weights will be:

- ☐ `W_Bias = -0.060, W_A = 0.281, W_B = 0.370`
- ☐ `W_Bias = -0.460, W_A = -0.119, W_B = 0.370`
- ☐ `W_Bias = -0.260, W_A = 0.081, W_B = 0.570`
- ☐ `W_Bias = -0.260, W_A = -0.119, W_B = 0.570`

Learning Reverse Implication 2

Check all that apply for the next iteration of the Perceptron Learning Rule:

- ☐ A = 0, B = 0 will result in a change to one or more weights.
- ☐ A = 0, B = 1 will result in a change to one or more weights.
- ☐ A = 1, B = 0 will result in a change to one or more weights.
- ☐ A = 1, B = 1 will result in a change to one or more weights.
- ☐ No inputs will result in a change to one or more weights. (The function is learned.)

(NOT A) AND B Weights

Function (NOT A) AND B: The output is 1 (true) if and only if A is 0 (false) and B is 1 (true). Supply W_Bias, W_A, and W_B such that a perceptron would correctly compute (NOT A) AND B.