Stuart Harley

Lab 6 : Lab Report

Hypothesis: Is it possible to train a neural network to correctly identify the winner of a tic tac toe game based on the final board positions.

Methodology: For this experiment, we are only testing tic tac toe board combinations that have a winner (no ties) and that do not leave any spaces blank. We will be providing implementation for boards that do have spaces in a later lab. We will be implementing stubbed out methods of a neural network class that has been provided for us in order to build our neural network.

Results: (Part 1) After completing the neural network class except for implementing biases, the code passed tests 1, 2, 6, and 7, of the 11 tests provided for us.

(Part 2) The first tests I did were without implementing biases. Aka all biases within the NN are set to 0. I trained the NN with 1000 epochs on the tic-tac-toe.csv file, which contains 32 board formations, 16 wins for X, 16 for O, and then tested it against the tic-tac-toeFull.csv file. I printed out the board, the actual output, and the calculated output (rounded to either 0 or 1 for easier readability since all the outputs were either basically 0 or 1, which is expected.) (X win = 1, O win = 0).

[1 1 1 1 0 0 1 0 0] Actual Output: [1] Calculated Output: 1

[1 1 1 0 1 0 1 0 0] Actual Output: [1] Calculated Output: 1

[1 0 1 1 1 0 1 0 0] Actual Output: [1] Calculated Output: 0

[0 1 1 1 1 0 1 0 0] Actual Output: [1] Calculated Output: 0

[1 1 1 0 0 1 1 0 0] Actual Output: [1] Calculated Output: 1

[1 1 0 1 0 1 1 0 0] Actual Output: [1] Calculated Output: 1

[1 1 0 0 1 1 1 0 0] Actual Output: [0] Calculated Output: 0

[1 0 1 0 1 1 1 0 0] Actual Output: [1] Calculated Output: 0

[0 1 1 0 1 1 1 0 0] Actual Output: [1] Calculated Output: 0

[1 0 0 1 1 1 1 0 0] Actual Output: [1] Calculated Output: 0

[0 1 0 1 1 1 1 0 0] Actual Output: [1] Calculated Output: 0

[0 0 1 1 1 1 1 0 0] Actual Output: [1] Calculated Output: 0

[1 1 1 1 0 0 0 1 0] Actual Output: [1] Calculated Output: 1

[1 1 1 0 1 0 0 1 0] Actual Output: [1] Calculated Output: 1

[1 0 1 1 1 0 0 1 0] Actual Output: [0] Calculated Output: 0

[0 1 1 1 1 0 0 1 0] Actual Output: [1] Calculated Output: 0

[1 1 1 0 0 1 0 1 0] Actual Output: [1] Calculated Output: 1

[1 0 1 1 0 1 0 1 0] Actual Output: [0] Calculated Output: 0

[1 1 0 0 1 1 0 1 0] Actual Output: [1] Calculated Output: 1

[1 0 1 0 1 1 0 1 0] Actual Output: [0] Calculated Output: 0

[1 0 0 1 1 1 0 1 0] Actual Output: [1] Calculated Output: 0

[0 1 0 1 1 1 0 1 0] Actual Output: [1] Calculated Output: 0

[0 0 1 1 1 1 0 1 0] Actual Output: [1] Calculated Output: 0

[1 0 1 1 0 0 1 1 0] Actual Output: [1] Calculated Output: 0

[1 0 1 0 1 0 1 1 0] Actual Output: [1] Calculated Output: 0

[0 1 1 0 1 0 1 1 0] Actual Output: [1] Calculated Output: 0

[0 0 1 1 1 0 1 1 0] Actual Output: [1] Calculated Output: 0

[1 1 0 0 0 1 1 1 0] Actual Output: [0] Calculated Output: 0

[1 0 1 0 0 1 1 1 0] Actual Output: [0] Calculated Output: 0

[1 0 0 1 0 1 1 1 0] Actual Output: [1] Calculated Output: 0

[1 0 0 0 1 1 1 1 0] Actual Output: [0] Calculated Output: 0

[0 1 0 0 1 1 1 1 0] Actual Output: [1] Calculated Output: 0

[0 0 1 0 1 1 1 1 0] Actual Output: [1] Calculated Output: 0

[1 1 1 1 0 0 0 0 1] Actual Output: [1] Calculated Output: 1

[1 1 1 0 1 0 0 0 1] Actual Output: [1] Calculated Output: 1

[1 1 0 1 1 0 0 0 1] Actual Output: [1] Calculated Output: 1

[1 0 1 1 1 0 0 0 1] Actual Output: [1] Calculated Output: 0

[0 1 1 1 1 0 0 0 1] Actual Output: [0] Calculated Output: 0

[1 1 1 0 0 1 0 0 1] Actual Output: [1] Calculated Output: 1

[0 1 1 1 0 1 0 0 1] Actual Output: [1] Calculated Output: 0

[1 1 0 0 1 1 0 0 1] Actual Output: [1] Calculated Output: 1

[1 0 1 0 1 1 0 0 1] Actual Output: [1] Calculated Output: 0

[1 0 0 1 1 1 0 0 1] Actual Output: [1] Calculated Output: 0

[0 1 0 1 1 1 0 0 1] Actual Output: [1] Calculated Output: 0

[0 0 1 1 1 1 0 0 1] Actual Output: [1] Calculated Output: 0

[1 1 0 1 0 0 1 0 1] Actual Output: [1] Calculated Output: 1

[0 1 1 1 0 0 1 0 1] Actual Output: [0] Calculated Output: 0

[1 1 0 0 1 0 1 0 1] Actual Output: [1] Calculated Output: 1

[1 0 1 0 1 0 1 0 1] Actual Output: [1] Calculated Output: 0

[0 1 1 0 1 0 1 0 1] Actual Output: [1] Calculated Output: 0

[1 0 0 1 1 0 1 0 1] Actual Output: [1] Calculated Output: 0

[0 1 0 1 1 0 1 0 1] Actual Output: [0] Calculated Output: 0

[0 0 1 1 1 0 1 0 1] Actual Output: [1] Calculated Output: 0

[1 1 0 0 0 1 1 0 1] Actual Output: [0] Calculated Output: 0

[0 1 1 0 0 1 1 0 1] Actual Output: [1] Calculated Output: 0

[0 1 0 1 0 1 1 0 1] Actual Output: [0] Calculated Output: 0

[1 0 0 0 1 1 1 0 1] Actual Output: [1] Calculated Output: 0

[0 1 0 0 1 1 1 0 1] Actual Output: [0] Calculated Output: 0

[0 0 1 0 1 1 1 0 1] Actual Output: [1] Calculated Output: 0

[1 0 1 1 0 0 0 1 1] Actual Output: [0] Calculated Output: 0

[0 1 1 1 0 0 0 1 1] Actual Output: [0] Calculated Output: 0

[1 1 0 0 1 0 0 1 1] Actual Output: [1] Calculated Output: 1

[1 0 1 0 1 0 0 1 1] Actual Output: [1] Calculated Output: 0

[1 0 0 1 1 0 0 1 1] Actual Output: [1] Calculated Output: 0

[0 1 0 1 1 0 0 1 1] Actual Output: [1] Calculated Output: 0

[0 0 1 1 1 0 0 1 1] Actual Output: [0] Calculated Output: 0

[1 0 1 0 0 1 0 1 1] Actual Output: [1] Calculated Output: 0

[0 0 1 1 0 1 0 1 1] Actual Output: [1] Calculated Output: 0

[1 0 0 0 1 1 0 1 1] Actual Output: [1] Calculated Output: 0

[1 0 0 1 0 0 1 1 1] Actual Output: [1] Calculated Output: 0

[0 1 0 1 0 0 1 1 1] Actual Output: [1] Calculated Output: 0

[0 0 1 1 0 0 1 1 1] Actual Output: [1] Calculated Output: 0

[1 0 0 0 1 0 1 1 1] Actual Output: [1] Calculated Output: 0

[0 1 0 0 1 0 1 1 1] Actual Output: [1] Calculated Output: 0

[0 0 1 0 1 0 1 1 1] Actual Output: [1] Calculated Output: 0

[1 0 0 0 0 1 1 1 1] Actual Output: [1] Calculated Output: 0

[0 1 0 0 0 1 1 1 1] Actual Output: [1] Calculated Output: 0

[0 0 1 0 0 1 1 1 1] Actual Output: [1] Calculated Output: 0

As shown above, some of these boards are classified incorrectly. Some trends I noticed were

- if the board ended in 11100 and was an X win, it was classified as an O win.
- If the board ended in 1010 or 0110 and was an X win, it was classified as an O win.
- If the board ended in 11001 and was an X win, it was classified as an O win 4/5 times.
- Boards that ended in 101, 10011, 0111, or 001111 were generally classified as O wins, even when they were actually X wins.

Since the NN was trained on only some of the possible board formations, it learned that those patterns provided an O win in that training dataset. However, in the full dataset, there are boards with those patterns that result in X wins, so it classified them incorrectly.

I then did the same test except I trained the NN with the tic-tac-toeFull.csv file instead. These are the results.

[1 1 1 1 0 0 1 0 0] Actual Output: [1] Calculated Output: 1

[1 1 1 0 1 0 1 0 0] Actual Output: [1] Calculated Output: 1

[1 0 1 1 1 0 1 0 0] Actual Output: [1] Calculated Output: 1

[0 1 1 1 1 0 1 0 0] Actual Output: [1] Calculated Output: 1

[1 1 1 0 0 1 1 0 0] Actual Output: [1] Calculated Output: 1

[1 1 0 1 0 1 1 0 0] Actual Output: [1] Calculated Output: 1

[1 1 0 0 1 1 1 0 0] Actual Output: [0] Calculated Output: 1

[1 0 1 0 1 1 1 0 0] Actual Output: [1] Calculated Output: 1

[0 1 1 0 1 1 1 0 0] Actual Output: [1] Calculated Output: 1

[1 0 0 1 1 1 1 0 0] Actual Output: [1] Calculated Output: 1

[0 1 0 1 1 1 1 0 0] Actual Output: [1] Calculated Output: 1

[0 0 1 1 1 1 1 0 0] Actual Output: [1] Calculated Output: 1

[1 1 1 1 0 0 0 1 0] Actual Output: [1] Calculated Output: 1

[1 1 1 0 1 0 0 1 0] Actual Output: [1] Calculated Output: 1

[1 0 1 1 1 0 0 1 0] Actual Output: [0] Calculated Output: 1

[0 1 1 1 1 0 0 1 0] Actual Output: [1] Calculated Output: 1

[1 1 1 0 0 1 0 1 0] Actual Output: [1] Calculated Output: 1

[1 0 1 1 0 1 0 1 0] Actual Output: [0] Calculated Output: 1

[1 1 0 0 1 1 0 1 0] Actual Output: [1] Calculated Output: 1

[1 0 1 0 1 1 0 1 0] Actual Output: [0] Calculated Output: 1

[1 0 0 1 1 1 0 1 0] Actual Output: [1] Calculated Output: 1

[0 1 0 1 1 1 0 1 0] Actual Output: [1] Calculated Output: 1

[0 0 1 1 1 1 0 1 0] Actual Output: [1] Calculated Output: 1

[1 0 1 1 0 0 1 1 0] Actual Output: [1] Calculated Output: 1

[1 0 1 0 1 0 1 1 0] Actual Output: [1] Calculated Output: 1

[0 1 1 0 1 0 1 1 0] Actual Output: [1] Calculated Output: 1

[0 0 1 1 1 0 1 1 0] Actual Output: [1] Calculated Output: 1

[1 1 0 0 0 1 1 1 0] Actual Output: [0] Calculated Output: 1

[1 0 1 0 0 1 1 1 0] Actual Output: [0] Calculated Output: 1

[1 0 0 1 0 1 1 1 0] Actual Output: [1] Calculated Output: 1

[1 0 0 0 1 1 1 1 0] Actual Output: [0] Calculated Output: 1

[0 1 0 0 1 1 1 1 0] Actual Output: [1] Calculated Output: 1

[0 0 1 0 1 1 1 1 0] Actual Output: [1] Calculated Output: 1

[1 1 1 1 0 0 0 0 1] Actual Output: [1] Calculated Output: 1

[1 1 1 0 1 0 0 0 1] Actual Output: [1] Calculated Output: 1

[1 1 0 1 1 0 0 0 1] Actual Output: [1] Calculated Output: 1

[1 0 1 1 1 0 0 0 1] Actual Output: [1] Calculated Output: 1

[0 1 1 1 1 0 0 0 1] Actual Output: [0] Calculated Output: 1

[1 1 1 0 0 1 0 0 1] Actual Output: [1] Calculated Output: 1

[0 1 1 1 0 1 0 0 1] Actual Output: [1] Calculated Output: 1

[1 1 0 0 1 1 0 0 1] Actual Output: [1] Calculated Output: 1

[1 0 1 0 1 1 0 0 1] Actual Output: [1] Calculated Output: 1

[1 0 0 1 1 1 0 0 1] Actual Output: [1] Calculated Output: 1

[0 1 0 1 1 1 0 0 1] Actual Output: [1] Calculated Output: 1

[0 0 1 1 1 1 0 0 1] Actual Output: [1] Calculated Output: 1

[1 1 0 1 0 0 1 0 1] Actual Output: [1] Calculated Output: 1

[0 1 1 1 0 0 1 0 1] Actual Output: [0] Calculated Output: 1

[1 1 0 0 1 0 1 0 1] Actual Output: [1] Calculated Output: 1

[1 0 1 0 1 0 1 0 1] Actual Output: [1] Calculated Output: 1

[0 1 1 0 1 0 1 0 1] Actual Output: [1] Calculated Output: 1

[1 0 0 1 1 0 1 0 1] Actual Output: [1] Calculated Output: 1

[0 1 0 1 1 0 1 0 1] Actual Output: [0] Calculated Output: 1

[0 0 1 1 1 0 1 0 1] Actual Output: [1] Calculated Output: 1

[1 1 0 0 0 1 1 0 1] Actual Output: [0] Calculated Output: 1

[0 1 1 0 0 1 1 0 1] Actual Output: [1] Calculated Output: 1

[0 1 0 1 0 1 1 0 1] Actual Output: [0] Calculated Output: 1

[1 0 0 0 1 1 1 0 1] Actual Output: [1] Calculated Output: 1

[0 1 0 0 1 1 1 0 1] Actual Output: [0] Calculated Output: 1

[0 0 1 0 1 1 1 0 1] Actual Output: [1] Calculated Output: 1

[1 0 1 1 0 0 0 1 1] Actual Output: [0] Calculated Output: 1

[0 1 1 1 0 0 0 1 1] Actual Output: [0] Calculated Output: 1

[1 1 0 0 1 0 0 1 1] Actual Output: [1] Calculated Output: 1

[1 0 1 0 1 0 0 1 1] Actual Output: [1] Calculated Output: 1

[1 0 0 1 1 0 0 1 1] Actual Output: [1] Calculated Output: 1

[0 1 0 1 1 0 0 1 1] Actual Output: [1] Calculated Output: 1

[0 0 1 1 1 0 0 1 1] Actual Output: [0] Calculated Output: 1

[1 0 1 0 0 1 0 1 1] Actual Output: [1] Calculated Output: 1

[0 0 1 1 0 1 0 1 1] Actual Output: [1] Calculated Output: 1

[1 0 0 0 1 1 0 1 1] Actual Output: [1] Calculated Output: 1

[1 0 0 1 0 0 1 1 1] Actual Output: [1] Calculated Output: 1

[0 1 0 1 0 0 1 1 1] Actual Output: [1] Calculated Output: 1

[0 0 1 1 0 0 1 1 1] Actual Output: [1] Calculated Output: 1

[1 0 0 0 1 0 1 1 1] Actual Output: [1] Calculated Output: 1

[0 1 0 0 1 0 1 1 1] Actual Output: [1] Calculated Output: 1

[0 0 1 0 1 0 1 1 1] Actual Output: [1] Calculated Output: 1

[1 0 0 0 0 1 1 1 1] Actual Output: [1] Calculated Output: 1

[0 1 0 0 0 1 1 1 1] Actual Output: [1] Calculated Output: 1

[0 0 1 0 0 1 1 1 1] Actual Output: [1] Calculated Output: 1

As shown above, when the data is trained on the full combinations, it appears that since there are so many more X wins than O wins, it just assumes that X wins every time.

Part 3: I was unable to get my code to pass all the tests. I worked on this for a very long time and I tried it multiple ways, but I could not figure it out. The best I could do was to get it to pass all the tests except 2, 3, and 10.