**Deep Learning for Conway’s Game of Life**

**Introduction**

The problem addressed in this project was to train a neural network to learn the rules of Conway’s Game of Life. Conway’s Game of Life is a cellular automaton, which means it is a grid with cells that can be in different states, and given an initial state, the cellular automaton can generate the next generation based on a set of rules. Conway’s Game of Life uses an infinite grid, with each of its cells being one of two possible states, alive or dead. It has four transition rules:

1. Any live cell with fewer than two live neighbors dies
2. Any live cell with two or three live neighbors lives on to the next generation.
3. Any live cell with more than three live neighbors dies
4. Any dead cell with exactly three live neighbors becomes a live cell[[1]](#footnote-1)

This cellular automaton is Turing Complete, meaning that it can simulate any Turing Machine, so that it can compute any computable algorithm, although it may take a much longer time than a conventional computer. Despite its simplicity, Conway’s Game of Life is capable of producing complex structures, and has many moving pieces interacting with each other, which can make the results seem unpredictable to an untrained outside observer. The aim of this project is to try to get a neural network to learn the rules of the Game of Life, and simulate a game given a random starting board.

**Methods**

The model used is a neural network with two convolutional layers, each looking at a 3 by 3 grid in the board, followed by a fully connected layer that maps to a 32 by 32 board. The rules of the game of life are spatially based, so Convolutional layers are needed. Very little information is available from taking the entire data and not considering their spatial relationship if we used a model without the Convolutional layers. These layers look at a 3 by 3 grid because all that determines whether a cell lives or dies at the next time step are the cells that surround it. The activation function used was tanh, as it was demonstrated that it performed better than ReLU activation functions in other models trained on the Game of Life.[[2]](#footnote-2) The model was trained using data generated from a board specifically designed to demonstrate all the rules of the game efficiently, and which improved performance compared to random data sets.[[3]](#footnote-3)

The generation of the test data takes in the csv file containing the starting board position, and uses a PyTorch convolution with set weights to calculate the next step, with the center having a weight of 10 and the rest having a weight of 1, so that if the weights add up to 3, 12 or 13, the cell lives, and else dies.[[4]](#footnote-4) The optimizer used was Adam, once again based on the performance benchmarks in the Bibin and Dereventsov paper.[[5]](#footnote-5) To evaluate the model, the loss function used was Mean Squared Error. The accuracy measured the total percentage of the cells which were correctly predicted, using the model used for data generation as its point of reference.

First attempts at training the model used a deeper network, however the improvements in accuracy ended up being marginal, and the smaller network proved to be practically as effective, while being significantly faster, so in the end I chose to go ahead with the simpler model. Due to the small number of factors involved in determining the next step of an individual cell, there might not be the need to have many layers, and I believe that the model would still have performed well with a single convolutional layer.

**Results**

**Training Loss per Epoch for GameOfLifeModel graph
** The model performed very well, reaching an accuracy of 99.8% in predicting the next step. After the model is trained, it is used to run a sample game using a randomly generated starting position, which produces an accurate game. The loss per epoch fell very quickly in the first few epochs, before levelling out around 0.02 as can be seen in the figure. Below is a sample of the first 5 generations of a randomly generated starting board.

****

Overall, the model was very successful in generating games, with the results perfectly matching how the game should progress.

**Conclusions**

The model used was very successful in simulating a game of Conway’s Game of Life. With the high accuracy of the shallow network used, there was no need to deepen it. Since the model can predict the next state quite easily, improvements to this model might aim to get it to try and produce specific starting board configurations to achieve some goal board state. These could range from reaching a stable state with a certain number of live cells, or producing more complex structures such as oscillators[[6]](#footnote-6) or spaceships[[7]](#footnote-7). Since these would require the consideration of the starting position as a whole, or larger sections of it, there would need to be other layers which take in larger sections of the board. Additionally, there might need to be new training data using starting positions that can generate these structures. Additionally, further improvements could be making the model predict n states ahead, given a board configuration, which would need to account for a larger portion of the board.

**Bibliography**

Bibin, Anton, and Anton Dereventsov. “Data-Centric Approach to Constrained Machine Learning: A Case Study on Conway’s Game of Life.” arXiv, August 23, 2024. https://doi.org/10.48550/arXiv.2408.12778.

Grek, Tom. “Evolving Game of Life: Neural Networks, Chaos, and Complexity.” *Medium* (blog), January 27, 2020. https://medium.com/@tomgrek/evolving-game-of-life-neural-networks-chaos-and-complexity-94b509bc7aa8.

Johnston, Nathaniel, and Dave Greene. *Conway’s Game of Life: Mathematics and Construction*. Canada: Lulu, 2021.

1. Nathaniel Johnston and Dave Greene, *Conway’s Game of Life: Mathematics and Construction* (Canada: Lulu, 2021), 3. [↑](#footnote-ref-1)
2. Anton Bibin and Anton Dereventsov, “Data-Centric Approach to Constrained Machine Learning: A Case Study on Conway’s Game of Life” (arXiv, August 23, 2024), 7, https://doi.org/10.48550/arXiv.2408.12778. [↑](#footnote-ref-2)
3. Bibin and Dereventsov, 5. [↑](#footnote-ref-3)
4. Tom Grek, “Evolving Game of Life: Neural Networks, Chaos, and Complexity,” *Medium* (blog), January 27, 2020, https://medium.com/@tomgrek/evolving-game-of-life-neural-networks-chaos-and-complexity-94b509bc7aa8. [↑](#footnote-ref-4)
5. Bibin and Dereventsov, “Data-Centric Approach to Constrained Machine Learning,” 6. [↑](#footnote-ref-5)
6. Johnston and Greene, *Conway’s Game of Life*, 53. [↑](#footnote-ref-6)
7. Johnston and Greene, 83. [↑](#footnote-ref-7)