“An evaluation of the effectiveness of using a genetic neural network to improve previous algorithm-based game theory solutions”

**Alexander Moses**

6277932

Faculty of Engineering, Environment and Computing, Coventry University

MSc Data Science and Computational Intelligence (ECT104) Stage 1

Coventry, United Kingdom

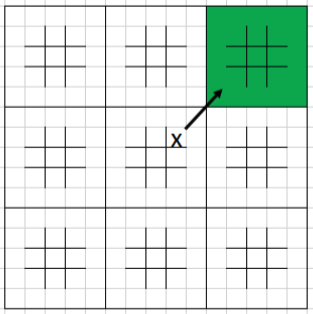
MosesA4@uni.coventry.ac.uk

***Abstract -* In game theory, traditional approaches to creating an AI for games is to use algorithms that rely on brute force, generating all the possible game states whilst also assessing them. The most common of these techniques being Min-Max using Alpha-Beta pruning and Monte Carlo tree search. While these are effective, they are also very taxing, and can be beaten by a human. A more recent, and better alternative is to use a neural network to create game agents capable of learning to outperform the traditional AIs. More and more attempts to create either a neural network to play the games, or add a neural network to the previous techniques, are made but these attempts only highlight the performance of the new solution with none of them performing an in-depth side by side comparison between traditional AI and deep learning-based AI. With this in mind, this project attempts to do so by taking Alpha-Beta pruning, apply a neural network on it and compare this AI against a previously created alpha-beta pruning AI. Hopefully this will allow for the identification of areas for future improvements and future studies.**

I. Client, Audience and Motivation

As traditional game theory techniques are limited by the computational power available as they usually rely on some kind of brute force, there are many attempts to implement neural networks into these solutions in an attempt to make the AIs more challenging to play against and make the AIs seem more human like. However, a lot of these attempts go into depth about their overall capabilities but not about how much of an impact using neural networks have in improving these techniques directly. So, this project attempts to take a previous solution and improve it with a neural network to see the direct impact of using neural networks and in turn can allow for an in-depth evaluation of using such solutions and potentially how they can be improved in the future. Game developers could benefit the most from this project as the in-depth evaluation of the application of the neural network can allow for the identification of ways to further improve previous neural network-based solutions.

II. Primary Research Plan

There is an advanced version of noughts and crosses, which is designed with the same 3x3 grid of normal noughts and crosses with each square in the grid containing its own 3x3 grid, totaling to 81 squares (3x3 grid within a 3x3 grid, refer to the diagram below). Each square in an inner 3x3 grid sends the next player to the respective square on the outer 3x3 grid. The current solution to making an AI for his game was attempted in the paper (Moses, 2018) stated in the literature review, where an alpha-beta solution was constructed. However, as this version of noughts and crosses has a total of 81 squares, the total amount of potential game states is too large (factorial 81) for a brute force solution to work. The alpha-beta solution creates a decision tree of all the possible game states and evaluate them based on their final outcome, and as it cannot create the entire tree in a timely fashion, the max depth of the decision tree is capped which stops the AI from being able to evaluate those paths. This flaw can be is somewhat fixed by creating an evaluation function to attempt to evaluate a partial game state, but this AI while capable at beating a human opponent, it took a large number of calculations and the difficulty of the AI was lacking.

The aim is to build a neural network to take this the partial alpha-beta made decision tree and evaluate the potential score for that partial tree, as the tree is incomplete the basic alpha-beta pruning algorithm is unable to provide a score, so the neural network will be used as the new evaluation function instead of the basic ones used in traditional solutions. The intended neural network is a recurrent neural net, as these networks have the ability to learn ‘memories’ over time with one of the most effective RNN being the LSTM. The long short-term memory network (LSTM) takes the concept of recurrent neural network and allows it to ‘memories’ long patterns over the entire data set, instead of a sequence which is what a normal RNN does, whilst forgetting short term ‘memories’. The neural net can be amplified further by implementing genetic programming as an optimization technique. Genetic programming mimics the process of natural selection, by creating chromosomes of the topology of the network (neuron layout) and evaluating them, with the ‘strongest’ chromosomes passing on their genes until the topology is fully optimized. The same can be done for the weights for the neurons so they are local minimums instead of global minimums.

The constructed neural net is trained against the Alpha-beta AI (AB-AI), the initial max depth of the AB-AI will be set to low (i.e. 12) to allow it to make its decisions in a fraction of a second. This is for the start of the training, as at the start the neural net will make bad decision and will require a lot of training for it to match the original AI so a lower depth allows this stage to be done much quicker. Once the neural net beats the AB-AI at 100 games in a row, the max depth of the AI is increased which increases its difficulty but also lengthens its response time. It does it continuously, slowly increase the max depth of the AB-AI, until the AB-AI reaches a max depth that causes it to take too long to respond of which would then be a good time for it to be trained against a human. The neural net is tested against a human, determining whether its trained enough and if so trained further against the human and potentially against other human volunteers to both test its capabilities and to improve it further.

Once fully trained and tested, the neural net is split into two different networks, one without the genetic programming and one with (with both having the same neuron weights). These two networks are then compared to several different configurations of the alpha-beta AI with various different improvements, this is so each different improvement can be evaluated and seen whether or not it can be an effective addition to the neural net, with the same being for the genetic programming. These different models are then compared on both their response time and win rate to fully evaluate the effectiveness of all the different configurations and the use of neural networks in game theory solutions.

* Create a clean (unmodified) alpha-beta pruning algorithm for the neural net
* Create the recurrent neural network
* Create the genetic algorithm optimization algorithm
* Train the genetic-RNN against the original alpha-beta AI
* Test the RNN against a human
* Train it further against a human
* Evaluate the effectiveness of the introduction of the neural network