International Journal of Advances in Engineering and Management (IJAEM)
Volume 3, Issue 7 July 2021, pp: 2232-2237 www.ijaem.net ISSN: 2395-5252

Modelling Chess AI using Deep Learning – Survey

Magesh Surya Ambikapathi
Computer Science Student, D.R.B.C.C.C Hindu college, Chennai, Tamilnadu
Corresponding Author: MageshSurya Ambikapathi

Submitted: 01-07-2021 Revised: 13-07-2021 Accepted: 16-07-2021

ABSTRACT—Chess has come a long way since it first appeared on a computer, and new chess algorithms are developed every year that outperform the previous ones. This paper discusses the different techniques used to create a chess program that outperforms when compared to studies based on AI and deep learning algorithms available in the literature. Study also highlights how few of the model architecture described exceed present top chess engines or professional human experts; some have reached extremely high performance, setting new domain benchmarks, while others employ a novel architecture or approach and produce average results.

Keywords— Chess programs, Artificial Intelligence, Deep Learning

I. INTRODUCTION

Tradition chess game algorithms available in the literature uses manual feature extraction and tuning. These methods are actually based on heat and trials and may not meet researcher's expectation. Due to automated learning, fast processing and excellent performance of deep learning in many applications, it gains popularity in the development of computer chess game. In fact, improvement in the performance of chess using artificial intelligence became one of the most researched filed in this domain [1]. The concept for developing such applications comes from the fact that people may enhance their skills in any game after playing it a few times. Many academics have been encouraged to work on deep learning's use in games, as well as striving for artificial general intelligence, by its capacity to recognize repeated patterns from dataset collected. The goal of this study is to look at existing research on leveraging deep learning to create chess engines. The articles presented here discuss the evolution of extremely simple network designs to highly sophisticated networks containing a wide range of characteristics as well as numerous hand-tuned features and domain-specific information. We try to understand how the architectures and various search algorithms perform in order to increase the model's performance.

The articles explain the characteristics of the various architectures employed, their benefits and drawbacks, and their performance in comparison to a certain benchmark. The architecture used is determined by the board game chosen, the target engine or minimum score to be achieved, or the determined to improve a degree of performance previously inaccessible in that environment. Multi-dimensional Recurrent Neural Networks (MDRNN), MultiLayer Perceptron (MLP), and Convolutional Neural Networks (CNN) are among the designs studied. This paper also emphasizes the necessity of selecting an effective search strategy. Choosing an effective searching algorithm is critical for a system to have as minimum time complexity as possible. Because certain architectures only operate with specific input formats, the input representation must also be addressed. A Convolutional Neural Network, for example, performs better when input is represented as an array. However, processing each image into a multidimensional array takes time and will slow down the operation. The efficiency of a flat array encoded with bits can be improved since it is much easier to compute on.

1.1 Chess Engines

Chess game was played initially on 10th century by human but after the development of artificial intelligence and machine learning, researchers always intended to improve its performance. Each engine is designed to operate on a typical computer, with no additional processing units necessary unless a highly intricate game tree or very high-definition visuals are required. Furthermore, several games are designed solely to be played via a network between two linked client PCs, allowing people to compete against one another or the computer.

- Min-Max Algorithm: The Min-Max Algorithm is a decision-based method that is employed in a game tree to minimize the worst-case loss. It was originally designed for two players, but it has since grown into other variations that may be used in more complicated versions of chess and other games.

- Pruning: Pruning is the process of cut-off the branch of the tree when the lowest or maximum node in a game tree has reached the value necessary and further traveling is not required.
- Surakarta Chess: It contains six horizontal lines and six vertical lines in this Chinese variation of combat chess. Chess pieces are placed on the thirty-six locations where horizontal and vertical lines cross. Eight arc lines connect all of these lines. The game is played with twelve chess pieces on each side.
- Hexagonal Chess: The hexagonal chess board has 91 cells and is shaped like a hexagon. Gliski's version, having a hexagonal board which is symmetric, is one of the most well-known. The chessmen's rules and moves differ according on the manufacturer.

II. LITARATURESURVEY

Over the last few decades, development of chess program using AI and ML attracted attention of many researchers. While the initial chess computers were incapable of challenging even a rookie player, now-a-days, a chess program developed using AI can perform better the even human professionls as seen by recent man vs. machine contests. Various researches have attempted to develop a computer that learns to play cognitive games with little or no prior understanding of the game's rules, according to a number of published papers. A typical chess playing machine investigates all of the possible moves from a chessboard setup before deciding on the next best move. The Deep Blue chess engine's brute-force searching approach has had a significant influence on artificial intelligence, although it has been proven to be resource intensive. The notion of Artificial Neural Networks is used in this research to provide a relatively easy and efficient way to developing an intelligent chess engine that can help and hint at possible moves.

Numerous methods have been used in a number of games throughout the course of more than 50 years of study in the field of computational engineering. Backgammon and checkers have both used reinforcement learning successfully [2]. Although reinforcement learning has been applied to chess, the resulting systems have only human master-level playing strength, which is far lower than the grandmaster-level playing power of state-of-the-art chess systems [3][4]. In [4], Thrun et al. proposed NeroChess based on automated training using end results. NeuroChess employs Artificial Neural Network-based assessment mechanisms (ANN). It employs an explanation-based neural network variation known as explanation-based neural network learning (EBNN). It creates neural network evaluation functions with TD. The sizes of the different board games are either fixed or adjustable. The branching factor grows exponentially as the board size grows. Because of these scaling difficulties, game engines have become more sophisticated in terms of time and space. As a result, in article [6], authors employs Multi-dimensional Recurrent Neural Networks (MDRNN) to address such a problem.

The results revealed that the size of the board had no meaningful impact on the architecture's performance.Neural networks are used to discover the best path from one board state to the next. When neural networks were used, it was discovered that pruning of expensive branches was done at a larger scale. This helped to pave the way for hybrid AI game-tree search systems. Therefore, in [7], Dendek et al. proposed a hybrid model consist of two CNN architecture. The suggested method is mathematically oriented, and it works by computing the Manhattan distance between a possible target square and a preset square that is arbitrarily chosen.

In recent years, developing alternative evaluation functions that may establish weights for the chess engine's neural network has been a prominent research area. The authors in [8], present a method for determining the positional value of each chess piece in this work. This chess engine employs the Alpha-Beta searching algorithm with iterative deepening, the Quiescence technique for position stability, hash tables, and the 0x88 hexadecimal approach for move generation.

It was demonstrated in [9] that trained deep value neural networks can play chess as well as high-level human chess players, without having to look ahead more than one move. The performance of Multilayer Perceptron and Convolutional Neural Network is investigated in this study. Different board representations are produced from data sets including a collection of games represented in Portable Game Notation (PGN). The Multilayer Perceptron and Convolutional Neural Network architectures employ these board representations as inputs. Stockfish, one of the most sophisticated chess engines, was utilized to assign a fractional centipawn (cp) value to each chess board position as a label for classification and regression experiments. The Mini-Max search and Alpha-Beta pruning algorithms are used in this neural network-based chess software to choose the next move. The results of the tests revealed that MLPs outperformed CNNs. The suggested system was still lagging behind the strongest existing chess engines and the best human players, based on these results

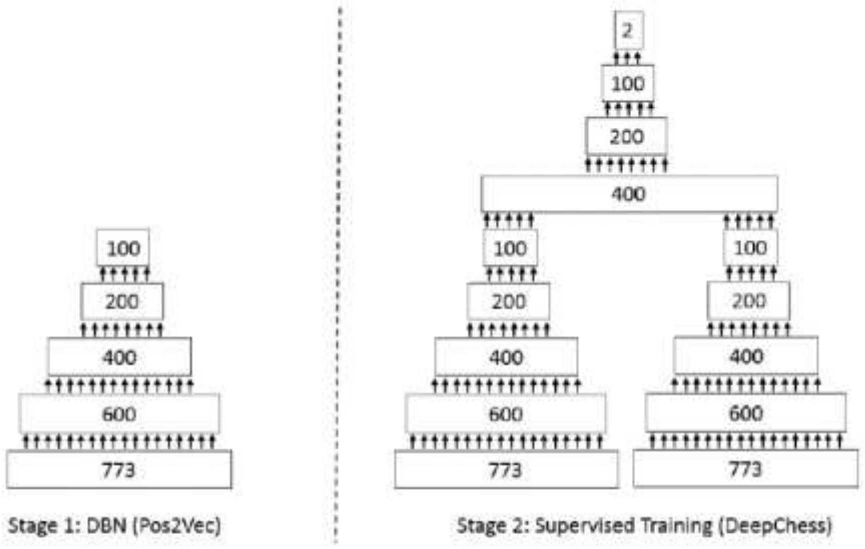

Figure 1: DeepChess Architecture [10]

In [10], the DeepChess program is developed, and its performance is comparable to that of grandmaster-level players. DeepChess' architecture is based on a deep neural network and shown in Figure 1. Manually, no domain-specific knowledge is supplied to the model. The training of deep neural networks is divided into two sections: unsupervised pre-training and supervised training. The model's unsupervised training allows for the identification of high-level characteristics. The model can analyze two chess board positions and choose the more advantageous one thanks to supervised training. The training is entirely based on a few million chess game databases. DeepChess provides scores to probable positions, with the score representing the position's quality. This model takes two locations as input and learns to predict which is the best position after comparing the two. In order to train, pairs of two moves are supplied as input. One move from a game in which White wins and another move from a game in which Black wins are included in these pairings.

An intelligent chess board is created in [11], which is extremely beneficial for novices to learn the rules of chess. The sophisticated method described here assists users in understanding all of the right locations of each chess piece on the board. The training technique for this project was a back propagation neural network. Without the use of computer devices, the architecture presented here works with the chess pieces according to their properties. Given the present condition of the board, the goal is to determine the next piece to be moved as well as the possible locations for this piece to go to.

Machine learning is utilized in [12] to build a new chess engine dubbed Giraffe. The TDLeaf method is used in the Giraffe software presented in this paper. The system accepted locations as inputs and outputted a sequence of integers that served as a signature for each position. The great search efficiency of humans is due to location similarity. If individuals can recognize the equally efficient actions, needless searching might be avoided. This resulted in a significant reduction in the average branching factor of the search trees. The Giraffe engine calculates the probability of all possible chess moves for a given chess piece.

This engine's primary flaw is its slow search pace, which is caused by its poor hit rate. The major aim of this article was to properly assess the location; otherwise, further searches would be necessary to compensate. This chess engine performed exceptionally well at the start and finish of the game because it concentrated on the tough moves rather than the long moves, and it also recognized intricate plays that are difficult for human players to comprehend. Extending further his study, Lai et al. proposed evaluator network that is made up of three layers (two hidden layers and an output layer), with all hidden nodes using Rectified Linear activation (ReLU). Features from various modalities are kept separate in the first two levels.

Only the last two layers are completely linked in order to capture interactions between high-level ideas obtained from distinct modalities' characteristics. The network design and restricted connection strategy are depicted in Figure 2, where an arrow indicates that all nodes in the first group are connected to all nodes in the second group.

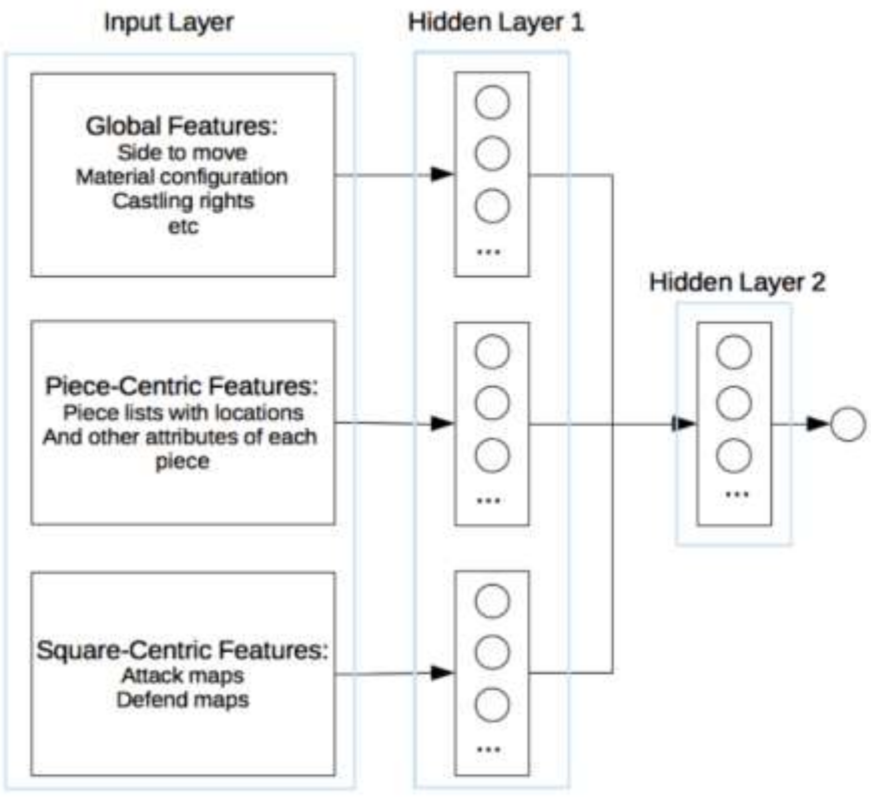

Figure 2: Architecture of proposed network.

In [13], author examines several techniques for training ANNs to assess chess positions using various combinations of architectures and input formats. The evaluation function of Stockfish, one of the best known chess engines, was used to label a dataset of roughly 3 million distinct chess situations played by extremely experienced chess players. They compared the results of multilevel perceptron and convolutional neural network for four different datasets that were normalized differently for different chess board notations, namely Bitmap and Algebraic notation. The findings demonstrate that MLP outperforms CNN for all datasets. Bitmap notation produces superior results whereas algebraic notation, which provides more information to the network, has a negative impact on the outcomes.

CrazyAra, a supervised learning based chess variant named crazyhouse, is presented in [14]. To forecast game movements, they employed networks and Monte-Carlo Tree Search. Although the dataset is smaller and of lower quality than those of Go and chess, the findings achieved are encouraging. For greater performance, they employed a more compact input board presentation by making the state completely Markovian, removing the history component in favor of the AlphaZero, and rescaling/normalization.

Surakarta chessboard (Figure 3) in computer gaming is a network application, according to Liqun Zhang et al [15]. Both sides on the client-side play on a server, and the game starts when both player log-in to the server. The server must be unique since the game alone determines who wins, and human interference merely slows down the engine. Unlike normal chess, this variant of the game follows the rule of movement of pieces either one square vertically or diagonally to a location where no other piece is present. In

addition, the board is shown in a unique style
.

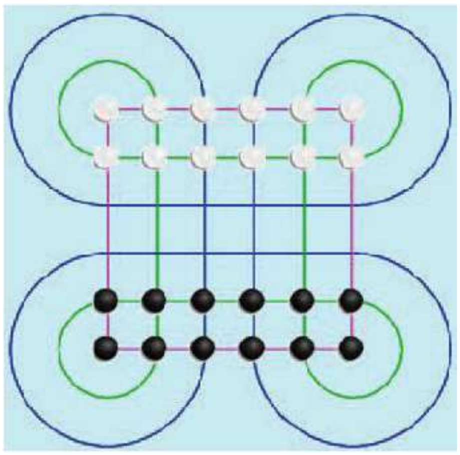
Figure 3: Surakarta Chess Board [16]

An improved chess game based on artificial neural network is proposed by Sharma et al. [18]. During the training phase, the proposed approach encodes the flow sequence, with input moves taken from the grandmaster game.

The encoded movements are then input into a specially constructed neural network, which is subsequently trained to provide the desired output. The algorithm picks up on the patterns (moves and strategies) that the human grandmasters create throughout their championship games. The model is trained till it minimizes the error to an optimum value. Figure 4 illustrates the train process of ANN.

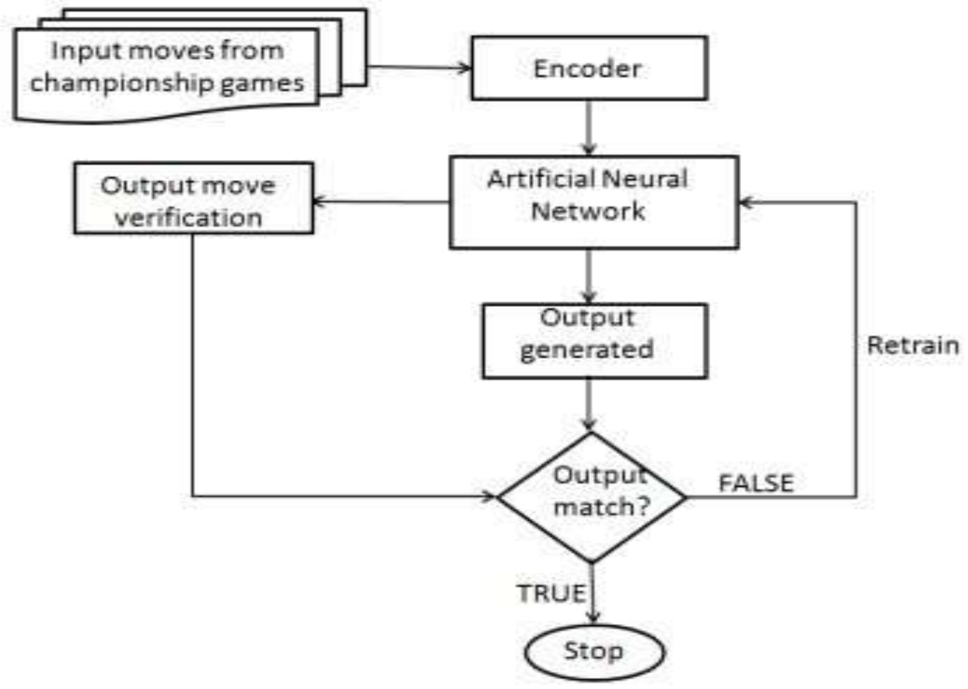

Figure 4: Training of ANN model

III. CONCLUSION

After analyzing past research on application of AI and DL for chess, we discovered that there exist a variety of architectures and algorithms that are suitable for building a Chess game that can outperform existing algorithms or even defeat human experts. It has been discovered that certain search approaches, structures, and algorithms provide promising results.

ANN has shown a significant reduction in the examination of low-performing board seats, with just the mostbeneficial options being investigated. Hardcoded engines, on the other hand, can explore millions of moves in a second and identify checkmate moves fast as the game progresses. ANNs can play tactical games and have won games, but they take a lot of computing resources as compared to hardcoded engines.

REFERENCES

[1] Schaeffer, Jonathan, MarkianHlynka, and ViliJussila. "Temporal difference learning applied to a high-performance game-playing program." In Proceedings of the 17th international joint conference on Artificial intelligence-Volume 1, pp. 529-534. 2001.

[2] Tesauro, Gerald. "Practical issues in temporal difference learning." Machine learning 8, no. 3 (1992): 257-277.

[3] Baxter, Jonathan, Andrew Tridgell, and Lex Weaver. "Learning to play chess using temporal differences." Machine Learning 40, no. 3 (2000): 243-263.

[4] Lai, Matthew. "Giraffe: Using deep reinforcement learning to play chess." arXiv preprint arXiv:1509.01549 (2015).

[5] Thrun, Sebastian. "Learning to play the game of chess." Advances in neural information processing systems 7 (1995).

[6] T. Schaul, J. Schmidhuber, “A Scalable Neural Network Architecture for Board Games,” Artificial Neural Networks – ICANN 2009 Lecture Notes in Computer Science, vol 5768, pp. 1005-1014, 2009.

[7] C. Dendek and J. Mandziuk, "A Neural Network Classifier of Chess Moves," 2008 Eighth International Conference on Hybrid Intelligent Systems, Barcelona, 2008, pp. 338-343.

[8] E. Vázquez-Fernández, C. A. CoelloCoello and F. D. SagolsTroncoso, "Assessing the positional values of chess pieces by tuning neural networks' weights with an evolutionary algorithm," World Automation Congress 2012, Puerto Vallarta, Mexico, 2012, pp. 1-6.

[9] Sabatelli, Matthia. "Learning to Play Chess with Minimal Lookahead and Deep Value Neural Networks." PhD diss., Faculty of Science and Engineering, 2017.

[10] O. E. David, N. S. Netanyahu, and L. Wolf, “DeepChess: End-to-End Deep Neural Network for Automatic Learning in Chess,” Artificial Neural Networks and Machine Learning – ICANN 2016 Lecture Notes in Computer Science, pp. 88–96, 2016.

[11] Omran, Alaa Hamza, Yaser M. Abid, and Huda Kadhim. "Design of artificial neural networks system for intelligent chessboard." In 2017 4th IEEE International Conference on Engineering Technologies and Applied Sciences (ICETAS), pp. 1-7. IEEE, 2017.

[12] M. Lai, “Giraffe: Using Deep Reinforcement Learning to Play Chess,” M.Sc. thesis, Dept. Computing., Imperial College London, London, 2015.

[13] M. Sabatelli, F. Bidoia, V. Codreanu, and M. Wiering, “Learning to Evaluate Chess Positions with Deep Neural Networks and Limited Lookahead,” Proceedings of the 7th International Conference on Pattern Recognition Applications and Methods, 2018.

[14] J. Czech, M. Willig, A. Beyer, K. Kersting, and J. Fürnkranz. (Aug. 2019). Learning to Play the Chess Variant CrazyHouseAbove World Champion Level with Deep Neural Networks and Human Data. [Online]. Available: https://arxiv.org/abs/1908.06660

[15] Zhang, Liqun, Lili Ding, and Zhenlai Li. "The design of Surakarta chess battle platform in computer game." In 2013 25th Chinese Control and Decision Conference (CCDC), pp. 2332-2335. IEEE, 2013.

[16] Nair, Anushka, KankshaMarathe and SuvarnaPansambal. “Literature Survey of Chess Engines.” International journal of engineering research and technology 5 (2018)

[17] Sharma, Diwas, and Udit Kr Chakraborty. "An Improved Chess Machine based on Artificial Neural Networks." International Journal of Computer Applications®(IJCA) (2014):097