JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

AI BASED CHESS ENGINE

[1]Anil Kumar Dubey,[2] Karan Singh Rawat,[3] Md Mashhood Raza Siddiqui

[1]Assistant Professor,[2]Scholar,[3]Scholar
[1]Department of Electronics and Communication Engineering,
[1]Greater Noida Institute of Technology, Greater Noida, India

Abstract: Game playing in chess is one of the important areas of machine learning research. Though creating a powerful chess engine that can play at a superhuman level is not the hardest problem anymore with the advent of powerful chess engines like Stockfish, Alpha Zero, Leela chess zero etc. Most of these engines still depend upon powerful and highly optimized look-ahead algorithms. CNN which is used primarily for images and matrix-like data is been proved successful with games like chess and go. Treating chess like a regression problem. In this paper, a supervised learning approach is proposed using the convolutional neural network with a limited look ahead. Data was collected around 44029 chess games from the FICS chess database with players having an Elo rating of 2000 and above. Our goal is to create a zero-knowledge chess engine. The trained model is then paired with a minimax algorithm to create the AI. Our proposed supervised system can learn the chess rules by itself from the data. It was able to win 10% of the games and draw 30% of games when manually tested against Stockfish computer engine with Elo of 1300. CNN can detect various tactical pattern to excel in games like chess even when using a limited lookahead search.

***IndexTerms*–CNN, Backtracking, look ahead algorithm, Evaluation Function**

I. INTRODUCTION

The improvement of PC chess programming methods and information can be partitioned into particular periods. Every time has its own arrangement of improvements, some of which can be followed back to expanded processor power, the accessibility of new equipment gadgets, or algorithmic headways [2]. Alan Turing began investigating a chess computer in 1946, but his notion was limited to handwritten records, which were further refined by Claude Shannon. Bernstein (1957) made the primary significant chess playing program, which worked on an IBM 704 PC fit for doing around 42,000 operations each second. This was not a 'brute force' software because it only considered the best seven moves based on chess lore heuristics. This is a rather narrow range of moves when compared to today's advanced brute force programs that create the entire range of moves at the root. In 1968 Greenblatt's program was the first to attain any type of distinguishable level of play. For many years, this was the most capable chess program, with an Elo rating of around 1500. It was the principal program to utilize rendering tables to diminish the inquiry space, and it included distinct peacefulness rules to work on strategic strength, the program used an underlying determination strategy to decrease the size of the game-tree. Yet again on account of its selectivity at the root hub, this program slips into the principal period. The first program to reach its full potential. Belfast-based software Blitz was created by computer pioneer Richard Hyatt and entered in the 1976 ACM North American Computer Chess Championship. By 1981, it was searching around 3000 nodes per second and routinely performing six ply searches. The introduction of assembly language and the Cray XMP computer with multiprocessing capabilities boosted this rate of analysis to 20,000 - 30,000 nodes a second in 1983. In 1996 Deep Blue won the opening game, making history as the first computer to defeat a world chess champion in a tournament setting. Every three minutes, Deep Blue computed 50 billion positions. Every three minutes, Kasparov calculated ten new positions. 200 processors were used in DEEP BLUE.

A Neural Networks technique, which is based on the way neurons work in people and can execute a variety of jobs just like humans. CNN, a sort of feed-forward neural network used in image analysis, has shown to be particularly successful in a variety of AI game tasks [4]. It is currently commonly utilized in Go and Chess engines of the modern era. With Alpha Zero's recent success, it's clear that CNN can be used to successfully forecast professional level chess moves. Our goal is to see how accurate CNN is in analyzing chess positions.

II. METHODOLOGY

For Dataset source, dataset form, data set analysis similar to the Dataset provided in Oshri, Barak, chess database having an Elo rating of 2000 and above of the year 2020 [1]. All the chess game in the database is in PGN (Portable Game Notation) format. The chess database contains 44029 total games of chess.

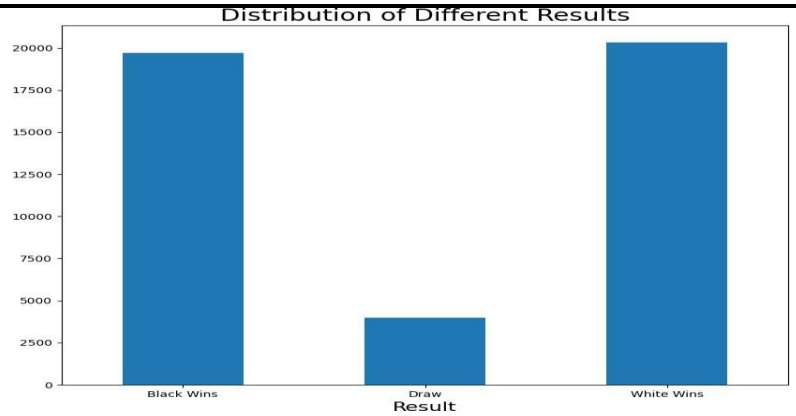

Fig.1 histogram representing total wins of black/white and draws

Bitboard representation outperforms traditional ACN (Algebraic chess notations) format during data training. In the bitboard format, the chess board is represented as an 8×8×12 bit vector of 0s and 1s, with each 12 piece represented by an array of 1s and blank space by 0s. Chess is shown in a categorized format. The turn bit (a single bit that specifies the turn), the castling bit (four bits that denote the castling right of both pieces), and the check bit are among the other attributes (two bits which represent whose king is in check)

The representation of a chessboard is a vector with 775 binary features. The bitboard representation is used to show where each piece in the vector is located. The first 64 features are white pawn locations, the following 64 features are white knight placements, and so on. The following is a representation of white pawns on a bitboard: Figure 2 number two A bitboard depiction of white is shown.

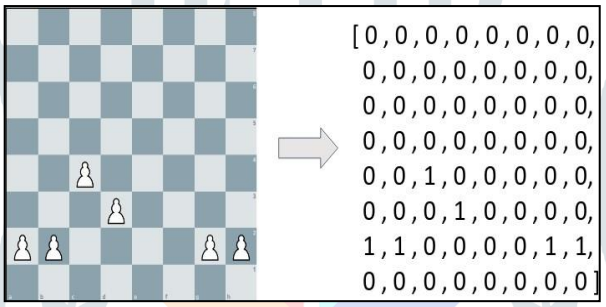

Fig 2: Pawns on the board (The diagram on the right is a one- dimensional matrix)

All piece vectors are then concatenated with each other along with the turn bit, castling bit and check bit.

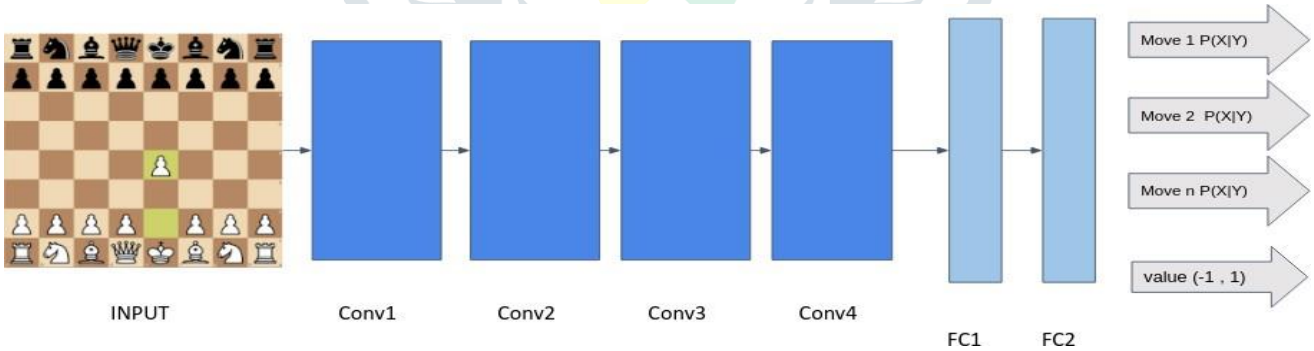

Fig 3. Visualization of move prediction network and neural network architecture

The neural network is implemented using PyTorch [2][6]. The FEN (Forsyth Edward notation) representation of the chessboard is input into the convolutional neural network, which is then transformed into a bitboard representation and followed by four 2D convolutional layers and two fully connected layers. The kernel size of the first layer is 3×3, the second layer is 2 × 2, and the last two levels are 1×1. The numbers 8, 16, 32, and 64 were used as filters [3]. ReLU-activation is used in all of these layers. For value activation, the tanh activation function was used. MSE (Mean squared error) was used as the loss function, and Adam's optimizer with a batch size of 256 was used. The training took place throughout 100 epochs [5].

III. Alpha-Beta Pruning

The alpha-beta method is an enhancement on the minimax search algorithm that decreases the number of nodes assessed on a big scale. This method is being used by Stockfish 12 with significant improvements. To use a real-world example, imagine someone is playing chess and it is their turn. The player's position will improve if he makes move "A" [7]. The player keeps on searching for moves to guarantee that a superior one has not been disregarded. Move "B" is likewise a fine move, yet the player rapidly comprehends that it permits the rival to compel checkmate in two maneuvers. Therefore, additional results from playing move B are presently excessive on the grounds that the adversary can drive a success [8]. The most elevated conceivable score that the adversary could compel later.

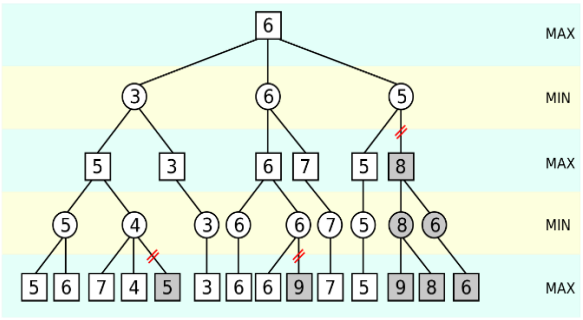

Fig 4: Alpha-Beta Pruning

IV. Comparison-Based with Alpha-Beta Search

The alpha-beta search technique is commonly used in chess engines. Alpha-beta is a profundity first inquiry technique that prunes the hunt tree's foreboding branches prior, expanding search efficiency. The search tree's root is a specific location, and the allowable motions for either side result in the layer nodes that follow. The strength of the search tree increases proportional to time, and hence the greater you're playing strength. The leaf nodes make use of an evaluation function [1].

The alpha-beta search algorithm is commonly used by chess engines. Alpha-beta is a depth-first search algorithm that prunes unpromising search tree branches earlier, enhancing search efficiency. The root of the search tree is a particular point, and the permissible moves for each side produce the following layer nodes. The additional time that is accessible, the further this search tree can be handled, bringing about a higher generally speaking playing strength. An evaluation function is used at leaf nodes.

A clever variant of an alpha-beta calculation that requires no position scores to finish the search tree is integrate to Chess AI. Store positions pos and pos instead of and values. For each new position, Chess AI are used to compare it to the existing pos and pos positions; if the comparison demonstrates that the new position is better than pos, it becomes the new pos; otherwise, the current node is pruned. Because Chess AI always compares positions from White's perspective, the predictions should be inverted when used from Black's perspective.

V. Conclusion

The convolutional neural network is trained to play chess using the publicly available data at FICS. The trained model was able to detect patterns on the chessboard and was able to deliver various tactics to improve and win the game. We conclude that a better model is possible with more refined and larger dataset having more high-level games. The model can be improved by giving more training time and can be benefited by using better hardware. The network can be improved by using Residual Convolutional Neural Network in the neural net.

References

[1] Omid E. David, Nathan S. Netanyahu, Lior Wolf. "DeepChess: End-to-End Deep Neural Network for Automatic Learning in Chess", 'Springer Science and Business Media LLC', 2016

[2] Botvinnik, M.; Cherevik, D.; Vladimirov, V., and Vygodsky, V. "Solving Shannon's Problem: Ways and Means, Advances in Computer Chess 7", Maastricht, University of Limburg, 1994

[3] Oshri, Barak. "Predicting Moves in Chess using Convolutional Neural Networks.'' 2015.

[4] Maddison, Chris J., Aja Huang, Ilya Sutskever, and David Silver. "Move evaluation in Gousing deep convolutional neural networks." arxiv preprint arXiv:1412.6564. 2014.

[5] Thrun, Sebastian. "Learning to play the game of chess" Advances in neural information processing systems, pp. 1069-1076. 1995.

[6] Lai, Matthew. "Giraffe: Using deep reinforcement learning to play chess." arxivpreprint arxiv:1509.01549, 2015.

[7] Knuth, D.E, and Moore, R.W. "An analysis of alpha-beta pruning. Artificial Intelligence" 6(4):293-326, 1975.

[8] Maesumi, Arman. "Playing Chess with Limited Look Ahead." arxiv preprintarXiv:2007.02130. 2020.