2022 IEEE 7th International conference for Convergence in Technology (I2CT)
Pune, India. Apr 07-09, 2022

Digitization of Chess Board and Prediction of Next Move

Preet Karia
Computer Engineering Department
Shah and Anchor Kutchhi Engineering College
Mumbai, India
preet.karia@sakec.ac.in

Vaibhav Jain
Computer Engineering Department
Shah and Anchor Kutchhi Engineering College
Mumbai, India
vaibhav.jain@sakec.ac.in

Monik Shah
Computer Engineering Department
Shah and Anchor Kutchhi Engineering College
Mumbai, India
monikketan.shah@sakec.ac.in

Sarika Rane
Computer Engineering Department
Shah and Anchor Kutchhi Engineering College
Mumbai, India
sarika.rane@sakec.ac.in.

2022 IEEE 7th International conference for Convergence in Technology (I2CT) | 978-1-6654-2168-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/I2CT54291.2022.9825379

***Abstract* -In recent years digital technologies and gadgets have grown so much. Along with other things, the field of Artificial Intelligence has also grown dramatically. Computer vision is one of the most compelling types of AI, which everyone must have experienced in one or the other way like in OCR, Image Recognition, object detection, google lens, etc. One might even come to the assumption that given any image or video to a computer, the computer can understand what's going on in it and can comment a few things on it, just like WE HUMANS. But that assumption is not yet very true. It may be possible in the future but at the present these need some work to become reality. Leveraging this same motivation, we intend to give some contribution for our dream future to become reality. Recognizing and understanding any arbitrary chess board from the image, thereby digitizing it, is the thing that we will be achieving from the proposed system. Along with all that, giving a comment on the current state of chess board, thereby predicting the next optimal move in the game is another thing which is included in proposed system. Think of a human-like robot trying to play chess. Robots must have something like this in order to play the game. In the proposed research work digitization of Chess Board is done by server application from chess board image acquired by user. Current state of the board is generated and sent back as a response to client application. With the help of GUI, the chessboard is generated from the received Forsyth-Edwards Notation (FEN) on the client side. Depending upon blacks turn or whites turn next optimal move to be played is shown to the user. With our proposed system digital instance is generated with accuracy of over 90%. All blank cells of the board are generated with almost 100% accuracy.**

Keywords: Chess, Chess board, Digitization, Chess Piece, Alpha beta pruning, Digital board, Xception

I. INTRODUCTION

Digitization is the process of converting any information in its digital format which computers can understand. By digitization of Chess Board, we mean to generate the digital instance of a particular chessboard state into the computer memory. Digitization is also important for various chess tournament organizers to broadcast there game online or for players who wish to share their game with friends.

In the year 1997 IBM Supercomputer named Deep Blue had defeated the world chess champion Garry Kasparov. This was the moment in the development of what we call "Machine Thinking", as computers had proven itself better than humans. Two decades later, computers regularly beat humans at chess. Nowadays, we all know computation power has increased drastically. We are able to carry a small, yet powerful computer in our pocket daily. Now after we have the digital instance of the chess board in the memory, computers can also Predict the next optimal move to be played.

The impact of computer chess has been seen in top level coaching and players as well. Coaches use software to help analyze the games of their students and trainees. And every playerneeds to be prepared for their opponents to play the unique, even inhuman moves learned from machine games. With our proposed system, easy computer aid could be taken on the ongoing chess game. Where the computer could provide its view on the current status of the game. Making a system that scans the photo of a chessboard and recognizes chess board from the image. Also, it can help to solve various problems in board gaming, like analyzing the game, etc. [1] Image processing is used to correctly detect and identify the configurations of chess pieces. [2]

II. EXISTING WORK AND DISCUSSION

Maciej A. Czyzewski, Artur Laskowski and Szymon Wasik Reference [1] proposed a system that scans the photo of a chessboard and recognizes chess board from the image. They used various iteration to detect lattice points and crop the chessboard. They then passed that cropped image to their algorithm that creates the FEN.

Cherly Danner, Mia Kafafy Reference [2] proposed a system that uses image processing technique to correctly detect and identify the configurations of chess pieces. The system uses the image processing to determine the chess pieces.

Cynthia Matuszek, Brian Mayton, Roberto Aimi, Marc Peter Deisenroth, Liefeng Bo, Robert Chu, Mike Kung, Louis LeGrand, Joshua R. Smith, Dieter Fox Reference [3] made Gambit: a custom, mid-cost 6-Dof robot manipulator system. This robot can play physical chess against human opponents in a non-idealized environment. This system includes a low-cost Kinetic-style visual sensor, a custom manipulator, and state-of-the-art learning algorithm for recognition of board and chess pieces on it. However, this approach is expensive and difficult to implement. Computer vision approach is extremely cost effective.

DING, J. Chessvision Reference [10] proposed a method to take an image of the chess board and output the computer representation of that board with piece recognition. It improves the previous works on piece recognition and makes it able to identify the more colored pieces. This

system is more robust to the similarities in color seen in real-life chess boards.

Chollet, F Reference [11] proposed the architecture of the Xception model. Various fine-tuning concepts are listed in this paper. It replaced the Inception modules with depth wise separable convolutions. This, model is used in this system to predict the probability of the piece on given position.

Rawat, W., and Wang, Z Reference [8] shows how convolution neural network (CNN) has been applied to visual task since 1980's. But they were dormant until mid-2000s when development in computational power and large amount of data supplemented in their development and brought progression since 2012. It also covers the development of a system from their predecessor to current stage. It also analyzed the early successes, role in deep learning renaissance, etc. of the algorithm.

T.A. Marsland Reference [4] reviews different search algorithms for alpha-beta pruning like Mini-Max search, Alpha-Beta algorithm, Quiescence Search, Minimal Game Tree, Aspiration Search, and Horizon Effect. It also compares different Enhancements for Alpha-Beta pruning like Minimal Window Search, Forward Pruning, Move Ordering Mechanisms, Progressive and Iterative Deepening, Transposition and Refutation Tables, Interpretation. This journal helped us to understand Alpha-Beta pruning algorithm and how it can be enhanced.

Hal-Tao Liu, Bao-En Guo Reference [5] designed a new pruning algorithm that selects the best move for Chinese Chess Computer Game (CCCG). In this new algorithm the endgame patterns have been fused into the search engine to prune the game tree.

ZHANG-Congpin, CUI-Jinling Reference [6] proposed improved alpha-beta pruning, here the estimated values of each node is calculated while expanding the nodes. Based on those estimated values the nodes are inserted into the tree. It means the deeper the tree is, the more nodes are cut down.

III. SYTEM METHODOLOGY

Our proposed system consists of two applications, server application and client application. User takes the snapshot of the board. After image acquisition, the image of the chess board is sent to server application for digitization. FEN of the current state of the board is generated and sent back as a response to client application. Now with the help of GUI the chessboard is generated from the received FEN on the client side. Depending upon blacks turn or whites turn next optimal move to be played is shown to the user. Figure 1 shows the block diagram.

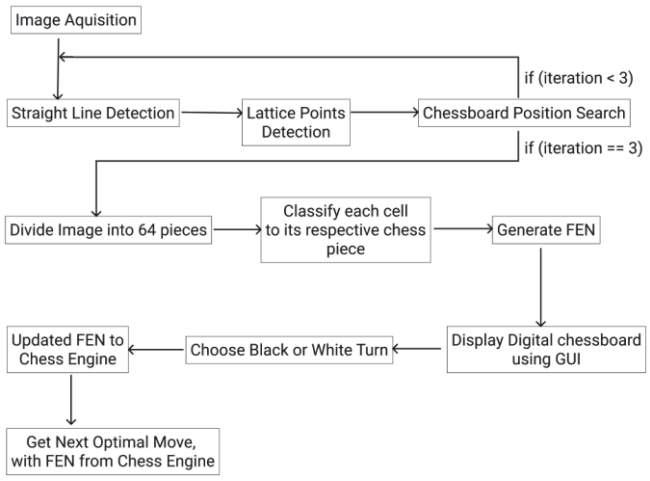

Fig. 1. Block diagram

The individual components in our proposed system:

- Digitization of Chess Board
  - Recognition of chessboard from a given image
    - Straight Line Detection
    - Lattice point search
    - Chessboard position search
  - Chess Piece and Color Recognition
  - Creating Digital board in Forsyth-Edwards Notation
- Prediction of Next Move

A. *Digitization Of Chess Board*

1) *Recognition of chessboard from a given image*

For the digitization of the chessboard, we first need to recognize the chessboard from the given image. Recognition of the board from the image is the obvious candidate for computer vision. Solution is based on generating a heat map to calculate the probability of a chessboard being located in a subsection of the image and cropping a tetragonal sub-area with highest probability values. Getting the final chessboard from the image in a single take increases the complexity drastically. In order to inculcate real world usage, the algorithm uses multiple iterations to get the final chessboard. Algorithm uses three iterations to get the final board. Now the chessboard is cropped in each iteration. Final iteration shows the cropped chessboard with corners A3, B3, C3 and D3. [1]

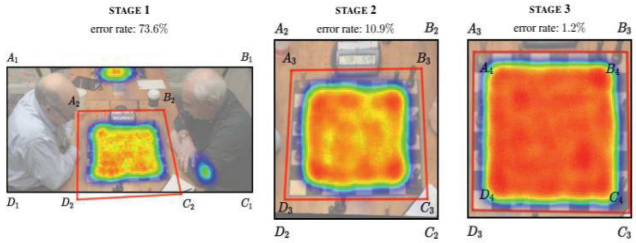

Fig. 2. Visualizing the process

a) *Straight Line Detection:*

Finds straight lines in an image and filters out lines that are expected to have no usability for detecting a chessboard position. The process of generating lines we used a Canny Edge detector. To extract all the structured information from the image, four different cases are analyzed in which parameters were set manually to correct low light, overexposure, blur and underexposure. Multiple collinear line segments are merged into a single line with Hough Transform to ensure the robustness of the algorithm. To improve the complexity for that purpose, disjoint set data structure is used.

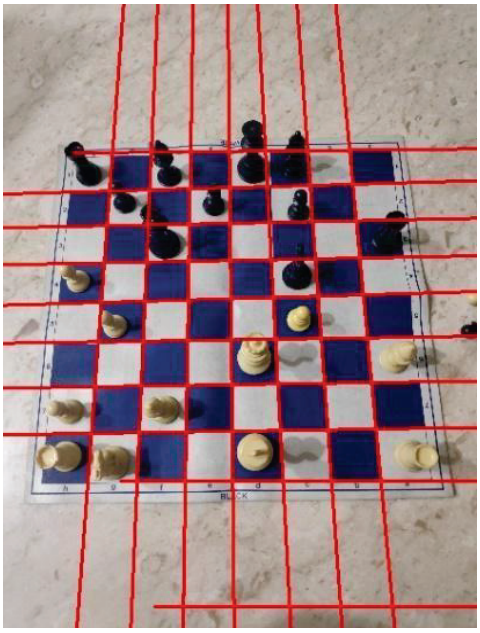

Fig. 3. Straight Line Detection in image

b) Lattice Points Detection:

Points that have a high probability of being a lattice point (i.e., a point where the corners of chessboard squares intersect) are selected. Initially, the algorithm assumes that each intersection of any pair of lines detected by the Straight Line Detection can be a chessboard lattice point. All these points are further processed by a geometric and neural detector which confirms whether the points are lattice points or not.

Fig. 4. Lattice Points Detection

c) Chessboard position search:

Our algorithm finds only the inner lattice points of the chessboard (i.e., a grid separating 6 × 6 squares, omitting lattice points located on the edges of the chessboard). To get the entire chessboard, an offset is included in all four sides of the quadrilateral formed by the outermost edges of the lattice points. Offset value is calculated based on the iteration and area A of the inner quadrilateral formed. Minimum offset considered is $\sqrt{A}/6$ which is the approximate side length of the unit square of the board. Value of offset is larger in initial iterations to compensate for the error.

$$E = 100\% - A / A_0 \quad (1)$$

$$O = \{\sqrt{EA}, \frac{\sqrt{A}}{6}\} \quad (2)$$

here O is the offset, E is estimated error, A_0 is the area of the image.

Fig. 5. Chess Board position search

2) Chess Piece and Color Recognition

Once the board's location in the image is known, it is separated into its individual squares to classify them and obtain the final result. This is done by simply dividing the image from the last iteration of the chessboard detection, a square, into an 8 × 8 grid. After dividing the original image into each of the board's squares, the next step is to classify them.

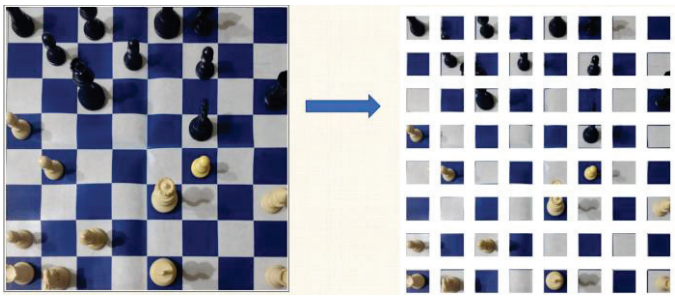

Fig. 6. Split the Cropped Image into 64 pieces for inferring the Chess Piece from the Convolutional Neural Network.

Each square could be empty or occupied with either black or white pieces. Thus, each individual square is classified into 13 distinct individual classes.

Each square would be classified into:

- Empty
- Black Pawn
- Black Rook
- Black Bishop
- Black Knight
- Black King
- Black Queen
- White Pawn
- White Rook
- White Bishop
- White Knight
- White King
- White Queen

The most widespread and the best algorithms to classify an image into their respective categories are CNNs. Here for getting better results we have fine-tuned the pre-trained model. In our case we have used the state of the art, Xception model in keras as our pre-trained model. The Xception Model is proposed by Francois Chollet. Xception is an extension of the inception Architecture which replaces the standard Inception modules with depth wise Separable Convolutions. For fine tuning the Xception model we have used around 57000 labelled images which we build from previous works in digitization of chessboard as well as we added some of our own image labelled data.

3) Creating Digital Board in Forsyth-Edwards Notation.

Forsyth-Edwards Notation (FEN) is a standard notation for describing a particular board position of a chess game. It makes it easy to translate any chess position into a single line of text. It facilitates the process of recreating positions using computers and allows players to share them and restart games from any point they desire. Digital state of chess is best described by this notation. Now combining everything from above together. Since a neural network will give the probability of a particular piece in a particular cell. We will use various chess rules to confirm the probabilities

Chess rules used are:

- There cannot be more than 32 pieces on a chessboard
- There can be a maximum of 16 pieces for a color
- At all times we need one king of each color on the board
- Maximum 8 pawns of each color

From all these we will generate FEN (Forsyth-Edwards Notation)

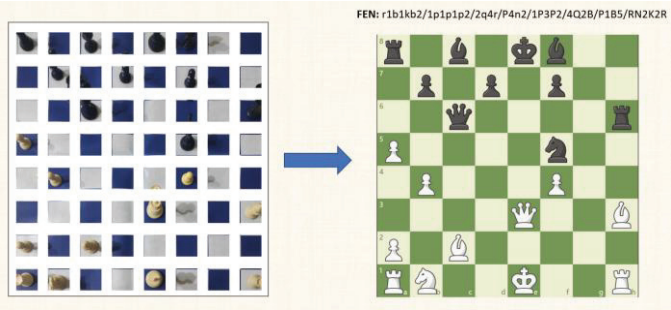

Fig. 7. Chessboard digitized to FEN and thus loading the GUI of chess state on the right

B. *Prediction Of Next Move*

After we get FEN, we can now pass this as an input to chess engine to predict the best move possible.

Creating a chess engine is our next major problem.

Algorithm used will be Alpha-Beta Pruning.

We can break this process in 2 parts.

1. Forming the list of all the legal moves.

2. Selecting the best Move from that list using the algorithm.

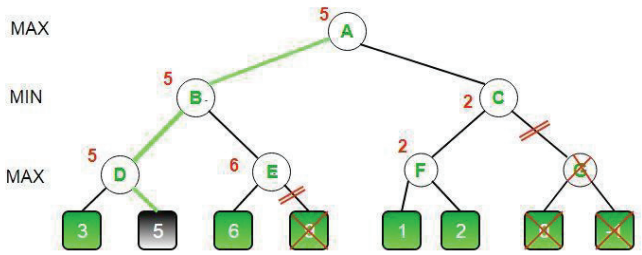

Fig. 8. Alpha Beta Pruning

Generating the legal move is an important step for this algorithm, because it decides the score of each move. Alpha-Beta Pruning is an optimized version of min-max algorithm, where each node chooses minimum or maximum value from its child. Each and Every level of the graph represents a White or black move. The main objective of the playing side is to maximize its score, and the main objective of the opposing side is to minimize the scores. Scores are calculated on the bases of position and pieces. Each piece is defined with some scores. In Alpha Beta Pruning, after all the recursion root node will hold the best move to be played.

IV. RESULT & ANALYSIS

In our proposed system, Digitization of Chessboard and Prediction of next move we have successfully implemented what we had actually desired in the problem statement. Now with this computing device in their pocket, one can just take the snapshot and get the digital version of the board. Recognizing the chessboard position from the image is quite robust and accurate.

In this presented work dataset is gathered from Reference [12]. Since the dataset was limited some more datasets have been created to increase the accuracy of the model.

First the dataset is partitioned into two sets: training dataset and testing dataset on 80:20 rule.

This dataset is used to retrain the existing Xception model.

Nest is the chessboard image is passed through various stages as discussed above which crops the chessboard and divide it into 64 parts. Those parts are then sent to Xception model which gives the probability of piece in that part. After having probabilities FEN is generated to digitize the chess board.

That FEN is passed to prediction algorithm which uses alpha-beta pruning to predict the next move.

While fine tuning the Xception model we got the following results in terms of Model Accuracy and Model Loss.

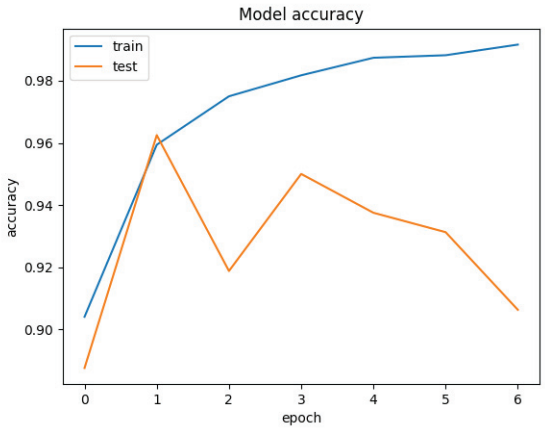

Fig. 9. Xception Model Accuracy

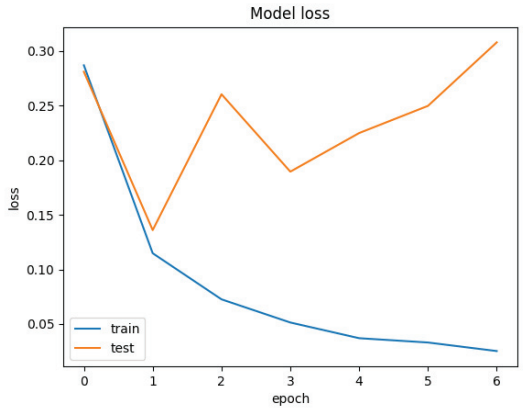

Fig. 10. Xception Model Loss

Results shown by the graph are quite good, yet when it comes to chess pieces recognition there are few invalid predictions of chess pieces. Although accuracy is decent enough of more than 90% and empty squares are almost detected with 100% accuracy, few improvements in move prediction could be made if size of the data set is increased. In our case we had around 57000 labelled images which we got access to from previous works as well as some of which we had created on our own.

To overcome the issue of a few invalid predictions, we have added an option in our GUI to change board pieces manually. This feature is quiteuseful and overcomes all the shortcomings.

V. CONCLUSIONS

With our proposed system one can scan the physical ongoing chess game, get the digital instance of that state of the chessboard, just by taking the single snapshot. After getting the digital instance, with the help of GUI, digital board can be accessed by the user. Depending upon black's turn or white's turn, next best optimal move is shown to the user using chess engine. Though the accuracy of the digitization is not 100% accurate, it can be improved by increasing the size of the data set. Never the less, our system is able to detect empty chess cells with 100% accuracy and other chess pieces are also detected with descent accuracy.

Previously the Digitization of Chess Board in tournaments was done with human aid. This approach was error prone and time consuming as well. So, some of the big tournaments used electronic chess boards for digitization. These boards were really expensive. Typically to get the move predicted by the computer in an ongoing chess game, you need to load the chess board manually in the computer's memory. Thus, it is a bit time consuming. With our approach you just need to get the snapshot of the board and get the digital instance of the board and get the next best optimal move to be played. It is also another step forward to a future where we dream that computers can understand any image or video. Thereby make a comment on it, just like we humans.

ACKNOWLEDGEMENT

The authors would like to thank Shah and Anchor Engineering College for giving us the background to enable this class project, to our project guide Sarika Rane for her valuable guidance, who worked on the project called Digitization of Chess Board and Prediction of Next Move in 2021

REFERENCES

[1] Maciej A. Czyzewski, Artur Laskowski and Szymon Wasik, Chessboard and Chess piece recognition with the support of neural networks, 2017.

[2] Cherly Danner, Mia Kafafy, Visual chess recognition, 2015.

[3] Cynthia Matuszek, Brian Mayton, Roberto Aimi, Marc Peter Deisenroth, Liefeng Bo, Robert Chu, Mike Kung, Louis LeGrand, Joshua R. Smith, Dieter Fox, Gambit: an autonomous chess-playing robotic system, 2011.

[4] T.A. Marsland, A review of game-tree pruning, 1986.

[5] Hal-Tao Liu, Bao-En Guo, A new pruning algorithm for game tree in Chinese chess computer game, 2012.

[6] ZHANG-Congpin, CUI-Jinling, Improved alpha-beta pruning of heuristic search in game-playing tree, 2009.

[7] Edwards, S. J. Portable game notation specification and implementation guide: Forsyth-edwards notation, 1994.

[8] Rawat, W., and Wang, Z. Deep convolutional neural networks for image classification: A comprehensive review. Neural Computation 29, 9 (2017), 2352– 2449

[9] DING, J. Chessvision: Chess board and piece recognition. Tech. rep., Stanford University, 2016

[10] Chollet, F. Xception: Deep learning with depth wise separable convolutions,2016.

[11] Image data for training CNN.

[12] https://www.dropbox.com/s/618l4ddoykotmru/Chess%20ID%20Public%20Data.zip?dl=0

[13] Schubiner, C. Chessboard image to fen. https://github.com/cschubiner/chessboard-image-to-fen, 2019