Application And Challenges of Alphazero Algorithm in Chinese Chess

Xiaxi Li

Department of Engineering, Shenzhen MSU-BIT University, Shenzhen, 518172, China

1120200239@smbu.edu.cn

Abstract. In the field of artificial intelligence, AlphaZero has achieved significant success in games like Go and Shogi. However, its application in Chinese Chess remains relatively unexplored. This study aims to unveil the potential and limitations of AlphaZero in Chinese Chess and explore strategies for optimizing its performance. In this paper, we begin by reviewing the fundamental principles and algorithmic structure of AlphaZero, emphasizing its deep reinforcement learning and self-play characteristics. Subsequently, we delve into the specific application of AlphaZero in Chinese Chess, encompassing aspects such as chess rule representation, training procedures, position evaluation, and strategy prediction. By analyzing successful instances of AlphaZero in Chinese Chess, we showcase its potential in the domain of board games. Furthermore, this paper elucidates the challenges that AlphaZero encounters in the context of Chinese Chess, including issues related to computational complexity and position representation. We particularly underscore the potential cost escalation due to high computational resource requirements and the risk of data loss or excessive time consumption associated with extended computation periods. IN conclusion, this research provides valuable insights into the application of AlphaZero in the realm of Chinese Chess and offers recommendations for future research directions to further expand its utility in this domain.

Keywords: Artificial intelligence, alphaZero algorithm, reinforcement learning, Chinese Chess, computational complexity.

1. Introduction

In the present era, with the rapid advancement of computer science and technology, Artificial Intelligence (AI) has become the focal point of both the scientific and industrial domains [1-4]. Over the past decade, the AI field has witnessed remarkable achievements, and two algorithms have spearheaded the AI revolution: AlphaGo and AlphaZero [1].

AlphaGo astounded the world in 2016 as the first computer program to defeat a human world champion in the game of Go. This breakthrough not only showcased the potential of AI in complex intellectual games but also ignited global research interest in AI across various domains. Subsequently, AlphaZero emerged in 2017 and, in a breathtaking manner, engaged in self-learning, surpassing its predecessor AlphaGo [5-8]. AlphaZero not only excelled in the game of Go but also demonstrated remarkable capabilities in various board games, including chess and shogi [2].

This review will focus on the application of AlphaZero technology in the domain of Chinese Chess and the challenges it encounters. We will explore the underlying principles of AlphaZero and how it has propelled research and application in the field of Chinese Chess. Furthermore, we will delve into the specific challenges and difficulties faced by AlphaZero in the realm of Chinese Chess. Through this review, we aim to unveil the potential of AlphaZero in Chinese Chess and address critical questions for future research, offering novel perspectives on the application of AI in the field of chess.

2. Overview of AlphaZero Technology

The AlphaZero algorithm is a form of deep reinforcement learning designed to achieve superhuman performance in various board games. Its development, originating from Google DeepMind, incorporates several unique and crucial features, largely influenced by David Silver's

paper "Mastering the Game of Go without Human Knowledge" and Julian Schritt wieser ' s paper "Mastering Atari, Go, Chess, and Shogi by Planning with a Learned Model."

2.1. Basic Principles

One of the core principles of AlphaZero is Monte Carlo Tree Search (MCTS) [9]. Through MCTS, AlphaZero efficiently explores the game tree, evaluates the potential value of different moves, and selects the most promising paths. This technology allows AlphaZero to optimize its strategy while handling vast game state spaces.

2.2. Self-Play

Another key feature of AlphaZero is self-play. Unlike traditional Go and Chess engines, AlphaZero doesn't rely on prior knowledge from human experts or opening databases. It trains itself through self-play, accumulating experience from each game and improving its strategy and position evaluation using neural networks. This self-learning approach enables AlphaZero to excel in various board games without the need for specialized customization [1].

2.3. Deep Neural Networks

AlphaZero employs deep neural networks to estimate the value of a game position and predict optimal moves. This neural network processes the board state using convolutional layers and residual networks, transforming it into a format understandable by the computer. With iterations of self-play, the neural network gradually enhances its prediction accuracy, thereby refining the gameplay strategy.

What sets AlphaZero apart is not merely its technological breakthrough but its paradigmatic impact, igniting extensive research interest in the field of artificial intelligence. The fusion of self-play and deep neural networks opens new possibilities for computer performance in complex intellectual games, while also guiding future research directions in deep learning and reinforcement learning [1-2, 10-12].

3. Application of AlphaZero in Chinese Chess

In the realm of fully deterministic games, such as Shogi, Go, and Chess, the AlphaZero algorithm has exhibited exceptional performance. This section focuses on the application of the AlphaZero algorithm in the context of Chinese Chess.

3.1. Representation of Chinese Chess Rules

Regarding the representation of Chinese Chess rules, AlphaZero employs an encoding method based on the state of the game board. The Chinese Chess board is depicted as a matrix, with each grid containing information about the respective position. This information encompasses the type of chess piece (e.g., chariot, horse, elephant, guard, general, cannon, pawn, etc.), color (red or black), and the piece's specific location [13-15]. This approach enables AlphaZero to precisely capture the dynamics of the chessboard. To feed the representation of Chinese Chess rules into the neural network, AlphaZero utilizes a deep convolutional neural network (CNN) [16]. The design of the CNN is tailored to meet the demands of deep reinforcement learning, efficiently processing image-like data, which is analogous to the representation of the Chinese Chess board. Leveraging convolutional layers and residual networks, AlphaZero extracts pertinent features from the depiction of the Chinese Chess board, which are subsequently utilized for strategy prediction and position evaluation.

During the training phase of AlphaZero, the CNN learns various aspects of different chess positions, including the configurations of chess pieces, their relative positions, and potential moves. For instance, there are seven types of chess pieces for each player in Chinese Chess: chariot, horse, cannon, elephant, guard, general, and pawn, totaling 14 feature planes [11], with each player having seven feature planes. Additional feature inputs further contribute to the formation of the neural network. The dimensions of the neural network vary depending on the nature of the feature inputs.

These features are critical to Alpha Zero's decision-making process as they aid in predicting optimal moves and assessing the quality of the position.

To illustrate, in the case of Go and Chess: In Go, chess pieces are divided solely into black and white, and thus, an N (19) xNx2 representation is employed to depict a moment with black and white pieces. Eight moments, including the current one and the previous seven positions, are considered. In Chess, apart from distinguishing by color, chess pieces are categorized as king, queen, rook, knight, bishop, and pawn, amounting to six types. Consequently, an N (8) xNx(Tx12) representation is used, and an additional NxNx(Tx2) representation is incorporated to determine whether both sides have encountered a repetitive position. Similar to Go, there is a constant plane to indicate whose turn it is in the current position. Furthermore, Chess introduces an additional plane to signify the total number of moves made up to the current position. Chess also features a unique castling rule, which can be classified into short and long castling, necessitating four additional planes. Lastly, Chess rules dictate that if no capture occurs within 50 consecutive moves, the game results in a draw. Thus, an extra plane is allocated to indicate the number of moves without progress in the current position. In total, the number of planes amounts to 8x8x119 [1] (figure 1).

Go		Chess		Shogi	
Feature	Planes	Feature	Planes	Feature	Planes
P1 stone	1	P1 piece	6	P1 piece	14
P2 stone	1	P2 piece	6	P2 piece	14
		Repetitions	2	Repetitions	3
				P1 prisoner count	7
				P2 prisoner count	7
Colour	1	Colour	1	Colour	1
		Total move count	1	Total move count	1
		P1 castling	2		
		P2 castling	2		
		No-progress count	1		
Total	17	Total	119	Total	362

Figure. 1 Input features used by AlphaZero in Go, Chess and Shogi respectively [1]

3.2. Training Process and Position Evaluation

The training process of AlphaZero consists of two key components: the policy head and the value head. These two components work together to continually enhance AlphaZero's chess skills.

Policy Head: The policy head is a part of Alpha Zero's neural network responsible for generating a probability distribution over moves. During self-play, it learns to assign probabilities to each possible move, giving higher probabilities to more promising moves. This learning process benefits from Monte Carlo Tree Search (MCTS), which guides the network's choices based on previous self-play game experiences, improving move accuracy [1].

Value Head: The value head is another neural network component tasked with evaluating the current board position's value. It learns to predict the quality of positions. If a position favors AlphaZero, its corresponding value will be closer to 1; if unfavorable, it will be closer to -1. This value evaluation is learned from the outcomes of self-play games, where winning and losing positions provide valuable training signals.

AlphaZero's position evaluation is influenced by Monte Carlo Tree Search (MCTS) [1], and the loss function is derived from MCTS. During self-play, AlphaZero uses MCTS to simulate games and explore potential outcomes of different moves. Through multiple simulations, MCTS generates a multitude of positions and evaluates them based on the eventual game results. Part of the loss function comes from the outcomes of the simulated games in MCTS. Specifically, AlphaZero compares the neural network's position evaluation with the simulated evaluations from MCTS. If the neural network's evaluation aligns with the MCTS simulations, the loss will be close to zero. If differences

exist, the loss guides the neural network to adjust its evaluation to better reflect the outcomes of the simulated games.

The design of this loss function enables AlphaZero to continually improve its position evaluation based on the results of self-play, gradually enhancing its ability to accurately assess positions and, in turn, improving its strategy and move selection (Figure 2).

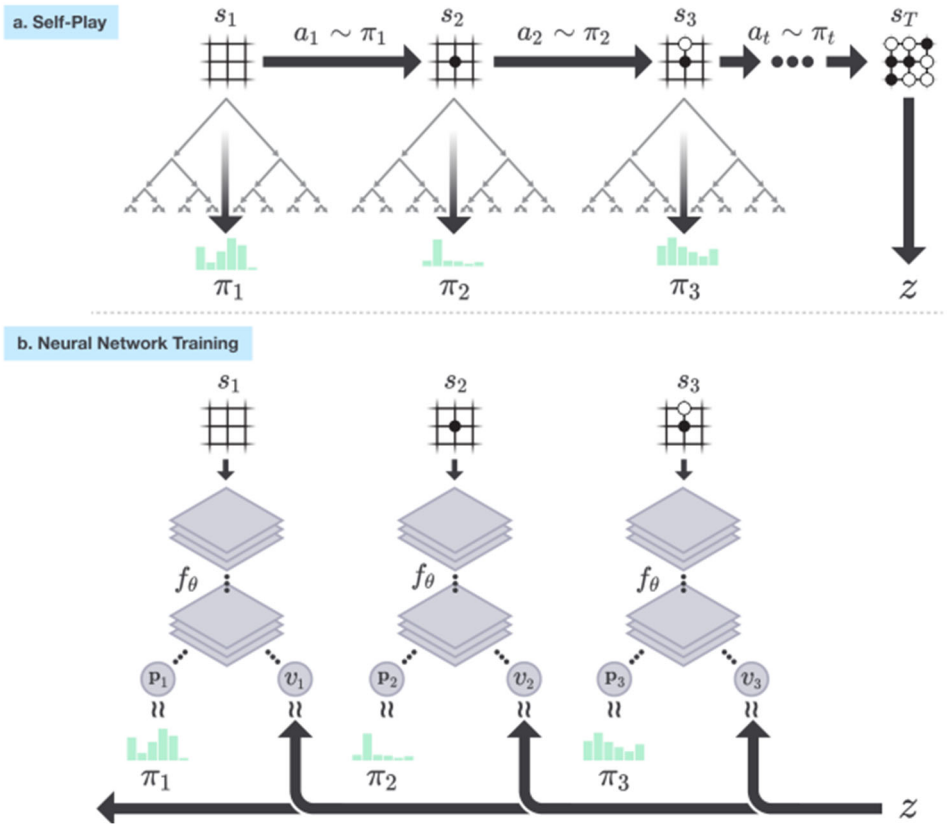

Figure. 2 Self-play reinforcement learning in AlphaGo Zero [1].

3.3. Policy Prediction

AlphaZero's policy network is a neural network component tasked with generating a probability distribution over possible move. The policy network takes the current board position as input and outputs the probability of each possible move. These probabilities guide AlphaZero's decisions in the game, making it more likely to select moves with higher potential value. The policy network's generation of move probability distributions is trained based on self-play data [1].

During self-play, AlphaZero accumulates a wealth of gaming experience and learns the probability distribution of moves through the neural network. This learning process benefits from Monte Carlo Tree Search (MCTS), which provides training data to the policy network by simulating the outcomes of different moves, allowing the network to more accurately estimate the quality of each move. The loss function plays a crucial role in policy prediction, guiding the training process of the policy network.

In AlphaZero, a portion of the loss function originates from the outcomes of simulated games in MCTS. Specifically, AlphaZero compares the move probability distribution generated by the policy network with the simulated move probability distribution from MCTS. If these two distributions align, the loss tends to zero. If significant discrepancies exist, the loss prompts the policy network to adjust its probability distribution to better reflect the results from MCTS. AlphaZero has achieved significant success in policy prediction.

Taking Go as an example [2], AlphaZero can accurately predict the best moves in complex chess positions without relying on traditional opening libraries or human expert knowledge. The training of its policy network is continually refined through self-play and MCTS-generated data, enabling

outstanding performance in various board games. In summary, policy prediction is one of the key factors contributing to AlphaZero's success. Through neural network learning and the guidance of the loss function, AlphaZero continually improves its move prediction during self-play, achieving performance beyond human capabilities.

4. Relationship Between AlphaGo and AlphaZero

AlphaGo and AlphaZero algorithms represent two significant milestones in the field of artificial intelligence. Their applications in board games like Go not only showcase the power of reinforcement learning but also reveal the potential of autonomous learning. Let's delve into the relationship between these two systems [16].

4.1. Similarities and Differences

AlphaGo and AlphaZero share important similarities and differences in the field of artificial intelligence. They both feature deep reinforcement learning and self-play as key characteristics. At their core, both employ Monte Carlo Tree Search (MCTS) for efficient game exploration. However, there are significant distinctions in how these principles are applied.

Difference Type	AlphaZero	AlphaGo	Difference
Use of Expert Data	Fully self-learned, no human	Utilized extensive human	AlphaZero's success in chess and shogi without human expert data demonstrates its ability to achieve outstanding results independently, whereas AlphaGo relied on extensive expert data.
Game Applicability	Applicable to multiple board	Mainly used for Go	AlphaZero excels in various board games, while AlphaGo is primarily limited to Go.
Self-Play Strategy	Self-play guides move	Relies on Monte Carlo Tree Search	AlphaZero learns its strategy directly through self-play, while AlphaGo combines Monte Carlo Tree Search with deep neural networks.
Computational	High	High	Both AlphaZero and AlphaGo require substantial computational resources, but AlphaZero typically demands fewer resources for the same task.
Complexity of Board	Relatively simple board	Relatively complex board	AlphaZero employs a relatively simple board representation, whereas AlphaGo's board representation is more complex, including scenarios with numerous stones.
Training Approach	Reinforcement learning	Combines supervised and	AlphaZero uses a purely reinforcement learning approach, while AlphaGo combines supervised learning with reinforcement learning.

Figure. 3 Similarities and Differences between AlphaZero and AlphaGo [1]

One of the most crucial differences is that AlphaGo leverages prior knowledge from human experts, including extensive libraries of Go openings and records of professional players. This knowledge provides AlphaGo with robust guidance, enabling it to perform exceptionally well in the early stages of a match. In contrast, AlphaZero learns entirely autonomously and does not rely on human knowledge, mastering board games from scratch. This autonomous learning approach grants AlphaZero versatility across different games [1] (Figure 3).

4.2. Outstanding Performance of the AlphaZero Algorithm

After extensive training, the AlphaZero algorithm demonstrates a commendable winning rate when playing against top-tier AI chess algorithms such as Stockfish [14]. (Figure 4-5).

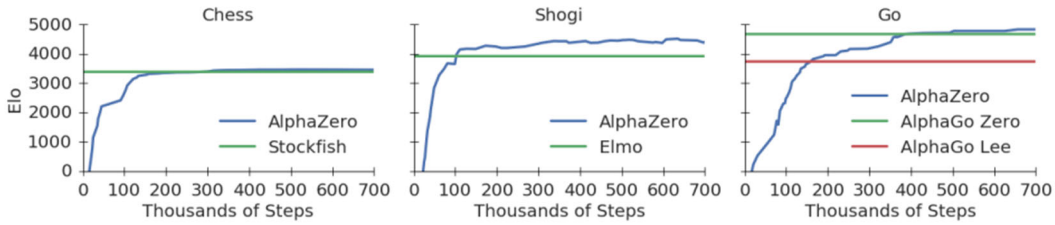

Figure. 4 Training AlphaZero for 700,000 steps [14]

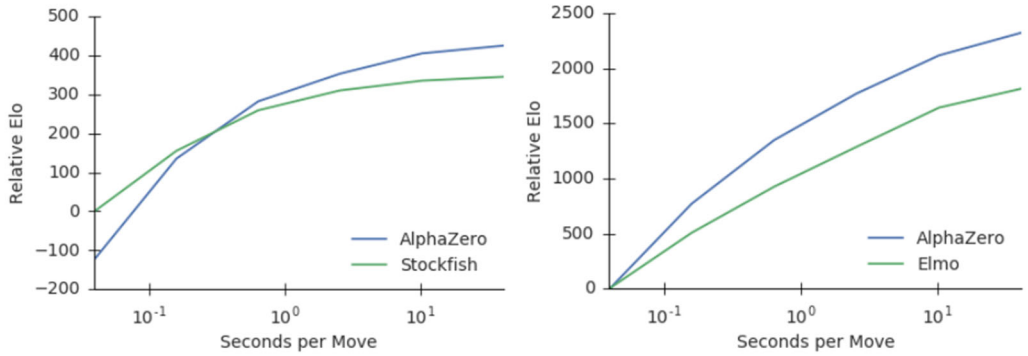

Figure. 5 Scalability of AlphaZero with thinking time, measured on an Elo scale [14]

5. Challenges and Limitations of AlphaZero in Application

The challenges and limitations of applying AlphaZero have been a topic of considerable concern, holding significant implications for practical usage. These issues often revolve around the problem of excessive training iterations, resulting in a substantial increase in computational costs. This encompasses not only hardware expenses but also entails time and energy costs.

5.1. High Costs

One major challenge that the AlphaZero algorithm faces in complex games like Chinese Chess is computational complexity. While AlphaZero demonstrates exceptional performance in self-play, tackling intricate games demands a significant number of computational resources, leading to a potential surge in computational costs. For instance, variants of AlphaZero, such as "JiajiaZero" [11] and "IcyChessZero," [12] may encounter issues related to excessive training iterations in practical applications, leading to cost escalation or situations where data is lost due to excessively long computation times.

In games like Chinese Chess, the state space is vast, and the features are intricate, surpassing the complexity of Go. AlphaZero employs Monte Carlo Tree Search (MCTS) to explore different moves and positions. However, in complex games, this search process requires more iterations and simulations, resulting in substantial computational demands. This not only necessitates more hardware resources but also significantly extends the time required for each step, involving extensive simulations and self-play.

Additionally, the computational complexity of AlphaZero is influenced by the training process. To achieve exceptional performance, AlphaZero must undergo millions of self-play games, posing a formidable challenge in terms of computational resources and time constraints. Training in environments with limited time and hardware budgets often yields suboptimal results.

5.2. Difficulties in Capturing Positional Features

Effective representation of game positions is crucial for AlphaZero's performance. AlphaZero employs neural networks to learn and represent game positions, but capturing critical information about positions is a significant challenge in complex games like Chinese Chess.

In the paper "Mastering the Game of Go without Human Knowledge," the authors highlighted the challenges of positional representation in Go. They noted that in Go, every stone on the board can influence the position dynamically, making it difficult to model using traditional methods. AlphaZero uses Convolutional Neural Networks (CNNs) to process Go positions, but the design of CNN structures and how to represent and process board states effectively remain areas of in-depth research.

In complex games like Chinese Chess, the challenges in positional representation become even more intricate. There are various types of chess pieces on the board, each with different movement rules and values. Changes in positions may be subject to specific rules, such as the check rule in Chinese Chess. Therefore, designing a representation that accurately reflects these factors is a formidable challenge. Some designers choose to quantify positional features themselves as input features for neural networks, but the original concept in the paper suggested that AlphaZero should independently extract positional features and model them during training [12].

Future research could explore more precise and efficient methods of positional representation to enhance AlphaZero's performance in complex games.

5.3. Generalization and Transfer

Generalization and transfer learning pose challenges and limitations for the AlphaZero algorithm. While AlphaZero excels in self-play, successfully extending its capabilities to different games or application domains remains a challenging endeavor.

For example, in "Mastering the Game of Go without Human Knowledge," AlphaZero achieved remarkable success in Go. However, applying the same algorithm to other board games, such as chess and shogi, requires retraining and adjustments. When applied to Chinese Chess, specific feature code and neural network dimensions were also rewritten. This suggests that there are limitations in the transfer learning aspects of AlphaZero across games. Generalization to different domains may also involve issues related to positional representation, as positions in different games may have distinct characteristics.

6. Future Research Directions

6.1. Balancing Achievements and Challenges

The successful application of AlphaZero in domains like Chinese Chess has brought about significant achievements and opened up new possibilities for the use of artificial intelligence in complex intellectual games. However, this success is accompanied by certain challenges, with computational complexity being the most prominent. AlphaZero requires extensive computational resources and time in self-play, which may be constrained in practical applications. The key to balancing achievements and challenges lies in finding methods to enhance the efficiency of AlphaZero, making it more applicable in real-game scenarios.

6.2. Further Research on Positional Representation

Positional representation plays a pivotal role in AlphaZero's performance. Future research directions can focus on improving positional representation methods to enhance AlphaZero's performance. For example, research can delve into better capturing features within chess positions and consider factors like historical information and opening libraries. These research directions hold the potential to further elevate AlphaZero's performance in the domain of chess.

6.3. Exploring the Potential of Human-Machine Collaboration

AlphaZero, as a tool for chess training, holds significant potential. Future research can explore modes of human-machine collaboration, enabling AlphaZero to collaborate with human grandmasters to collectively advance the field of chess. This collaborative approach will provide new perspectives and opportunities for chess training and research, contributing to the further advancement of the chess domain.

7. Conclusion

In summary, the AlphaZero algorithm has demonstrated immense potential in the domain of Chinese Chess. By combining core principles such as deep reinforcement learning, self-play, neural networks, and Monte Carlo tree search, it has successfully been applied to Chinese Chess, progressively improving its own chess-playing abilities through self-play. This algorithm does not rely on human expert data, making it applicable to various chess-like games and showing promising applications across a wide range of domains.

However, AlphaZero also faces several challenges in practical applications. Firstly, it demands significant computational resources, especially for games with high computational complexity like Chinese Chess, requiring substantial computing power and time. Secondly, there is still room for improvement in positional representation, necessitating more complex and precise methods to capture the intricacies of Chinese Chess. Lastly, while AlphaZero excels in self-play and achieves high win rates against top-level chess AI after training, its performance at lower training levels can be challenging. Future research directions include enhancing the computational efficiency of AlphaZero to reduce its dependence on computational resources, advancing positional representation methods to better capture the complexity of Chinese Chess, and exploring the potential of human-machine collaboration, leveraging AlphaZero as a chess training tool to assist human players in skill improvement.

In conclusion, despite significant progress in the application of AlphaZero in Chinese Chess, it still faces challenges. Future research should focus on addressing these challenges while further expanding the application of AlphaZero technology in Chinese Chess. This not only contributes to elevating the level of computer play in chess but also brings forth new insights and opportunities for the development of the human chess domain.

References

[1] Silver, D., Schrittwieser, J., Hassabis, D. (2017). Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. *Nature* **3**.

[2] Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., (2019). Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model. *Nature* **342.**

[3] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., (2015). Human-level control through deep reinforcement learning. *Nature*, **518(7540)**, 529-533.

[4] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y. (2016). Continuous control with deep reinforcement learning. *International Conference on Learning Representations.*1-11.

[5] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal Policy Optimization Algorithms. *International Conference on Machine Learning*, 1-10.

[6] Anthony, T., Tian, Z., & Barber, D. (2017). Thinking Fast and Slow with Deep Learning and Tree Search. *International Conference on Machine Learning* **73(2).**

[7] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ..., & Hassabis, D. (2013). Playing Atari with Deep Reinforcement Learning. *Conference On Neural Information Processing Systems* **454(1)**.

[8] Tian, Y., Ma, J., Gong, Q., Sengupta, S., Chen, Z., Pinkerton, J., & Zitnick, L. (2019). ELF OpenGo: an analysis and open reimplementation of AlphaZero. *Proceedings of the 36th International Conference on Machine Learning*, **97**, 6244-6253.

[9] Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P. (2012). A survey of monte carlo tree search method. *IEEE Transactions on Computational Intelligence and AI in Games,* **4(1)**, 1-43.

[10] Arulkumaran, K., Deisenroth, M. P., Brundage, M., & Bharath, A. A. (2017). Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, **34(6)**, 26-38.

[11] Lee David/Leela-chess-to-Chinese-Chess Available: https://github.com/leedavid/leela-chess-to-Chinese-Chess.

[12] Bupticybee/icyChessZero. Available: https://github.com/bupticybee/icyChessZero.

[13] OpenAI. Openai five. (2018). Available at: https://blog.openai.com/openai-five/.

[14] Online chess games database. Available at: https://www.365chess.com/.

[15] Allis, V. (1994). Searching for Solutions in Games and Artificial Intelligence. *PhD thesis, University of Limburg*, Netherlands.

[16] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G. (2016). *Mastering the game of Go with deep neural networks and tree searc*h.