ISSN 2959-6157

The Advantage of Board Game with Deep Reinforcement Learning and Causal Inference

Juncheng Ming[1, *]

[1] MAJESTIC INTERNATIONAL COLLAGE, Foshan, China
*Corresponding author: junchengming@gmail.com

Abstract:

With the continuous progress of artificial intelligence technology, the deep reinforcement learning method combining behaviourism and connectionism has shown fantastic performance far beyond the human level in chess games. This paper systematically introduces two mainstream chess games, including the standard algorithms of artificial intelligence technology in chess and Go, including Monte Carlo and deep reinforcement learning, and expounds on their algorithm principles and the core reasons for their strong learning ability. In addition, this paper discusses the limitations of existing methods in interpretability and puts forward the possibility of applying causal reasoning to deep reinforcement learning to solve interpretable problems. This study will provide a valuable reference for practitioners in related fields.

Keywords: Board Game; Deep Reinforcement Learning; Causal Inference; Artificial intelligence.

1. Introduction

Board games are vital applications to test the development level of artificial intelligence (AI) models, especially chess and Go. For chess, the randomness on the chessboard is lower than that of Go, but the complexity of the rules is higher than that of Go, so the game depends on heuristic functions. So when it comes to making AI, algorithms like open book are used, and the reliance on algorithms written for randomness, like machine learning or Monte Carlo trees, is weaker than in Go. For Go, the rules are more straightforward than chess, but the board is more extensive, and the possibility and randomness of moves are enormous, so the heuristic algorithm is not very helpful to Go AI. Therefore, when making Go AI, algorithms such as machine learning or Monte Carlo trees that are good at handling randomness are more critical than heuristic algorithms [1].

In chess, a common AI algorithm is the minimax algorithm. The Minimax algorithm is a two-person zero-sum game algorithm that uses decision trees to help players make the best decisions. This algorithm is widely used in artificial intelligence designed for chess. All possible game states are treated as nodes in the game tree, and each node will have a different value to determine whether this move is the most profitable. The root node is the game's initial state, and the leaf node is the result of the game. Two players each have a mathematical goal. The most significant player will try to retrieve the leaf node containing the maximum value in the current decision tree, while the most minor player will do the opposite. The Minimax algorithm is often combined with Alpha-beta pruning to complete optimization. Alpha-beta pruning can delete some nodes that are unlikely to be applied by evaluating the weight of each leaf node. This method can reduce the number of nodes in the decision tree and thus reduce the calculation of AI to improve efficiency. The chess engine uses chess step sorting to improve the efficiency of search algorithms such as minimax and Alpha Beta pruning. Checking the most promising moves first will increase the possibility of early pruning, thus reducing the number of nodes to be evaluated in the strategy tree, enabling the chess engine to go deeper into the strategy tree quickly. The chess step sequence techniques commonly used in chess engines include the main change (PV) moves, eating and ascending, killer moves, historical moves, shifting table moves and static exchange evaluation (SEE). These methods determine the priority of chess moves according to the best action sequence, eating and promoting chess pieces, previous beta truncation and material gain and loss [2].

The AI algorithm commonly used in Go is based on Monte Carlo tree Search (MCTS). MCTS is an algorithm used in the decision-making process, especially for games like Go, which contains many possible moves and states to consider. The algorithm consists of four main steps: selection, expansion, simulation and return. The selection step starts at the root node and selects child nodes along the path of the tree until an end node or unexpanded node is reached. This step usually uses some heuristic strategy to determine the selected child node. The scaling step starts with the newly expanded node and performs a series of random simulations. The simulation results are used to evaluate the value of the nodes. The simulation step starts with the newly expanded node and performs a series of random simulations. The simulation results are used to evaluate the value of the nodes. The return step returns the simulation data along the exploration path to the root node, where the data for each node on the path, including parameters such as winning percentage, is updated during the return process. This step is used to adjust the parameters of each node so that the result is positive [3].

The algorithm is very suitable for Go AI because of its excellent handling of complex and changeable game states and giant prediction steps, and it can run efficiently without relying on heuristic functions. Quiescence search, as an optimization technique, extends the specific type of move (called' quiet' move) of strategy tree search beyond the standard search depth. A' quiet' move will not significantly impact the whole board's state, such as eating or putting a child. In chess, a quiescence search is performed after the standard search algorithm (such as minimax search) reaches a predetermined depth. When the static search is started, the algorithm will focus on the subset of non-static start, such as eating seeds or eating seeds, and then recursively search until it reaches a quiet position or a set static depth. Then, the static evaluation function will evaluate the quiet state, and the evaluation value will be passed up the policy tree. Finally, static search results will be combined with conventional search results for more accurate location evaluation results. This algorithm is not an independent search algorithm but an enhancement of the existing algorithm, which mainly improves the position evaluation in the search tree by considering tactical sequences beyond the standard depth.

With the development of deep learning, especially reinforcement learning, the model of deep reinforcement learning, represented by AlphaZero, shows the level of board games far beyond human beings. Reinforcement learning is an algorithm in machine learning. The best strategy is learned by training agents, which will be realized by combining with algorithms such as neural networks. The agents will interact with the environment to obtain superparameters, and the evaluation function will evaluate the returned superparameters and give feedback to the agents. If the returned parameters are positive, the agents will be rewarded. The agents will constantly adjust internal parameters and update the database by receiving feedback to optimize decision-making, and some algorithms will explore deep strategies through self-game [4,5].

This paper introduces the progress of deep learning in the application of board games and discusses the critical role of interpretability in applying board games based on deep learning. This paper hopes to provide a valuable reference for researchers in related fields through review.

2. Board Game with Deep Reinforcement Learning

The principle of Deep Reinforcement Learning, DRL) is to combine Deep Learning, DL) with Reinforcement Learning, RL), use the powerful perceptual ability of the deep neural networks to deal with complex environmental conditions, and combine the decision-making ability of reinforcement learning to optimize behaviour strategies, thus realizing intelligent decision-making. Its core framework can be summarized as follows: 1) Environmental observation. At every moment, the agent obtains a high-dimensional observation from the environment, including images, sounds and other information. 2) Environmental observation: At every moment, the agent obtains a high-dimensional observation of the environment, including images, sounds and other information. 3) Action selection: Based on the current state representation, the agent selects an action according to a Policy or Value Function. The strategy function directly outputs the probability distribution of actions, while the value function evaluates the value of each action and selects actions by maximizing the value. 4) Environmental feedback: After acting, the environment will give new observations and reward signals. The reward signal is used to evaluate whether the agent performs this action in the current state. 5) Strategy update: The agent updates the parameters of its strategy function or value function according to the reward signal fed back by the environment to make a better choice in the next decision.

Unlike other algorithms, deep reinforcement learning can continuously generate new training data and optimize its neural network model, thus achieving self-improvement. This is the core reason it quickly catches up with the human level, and its self-optimization strategy is shown in Figure 1. Figure 1 shows a self-game played by the program. The sequence of game states is $S_1,\ldots,S_T$. At each location S_T, MCTS will be executed using the network whose parameters are currently updated. Actions taken $a_t\,\pi_t$. The MCTS is used when training data is generated in each game, so the quality of training data generated is higher [6].

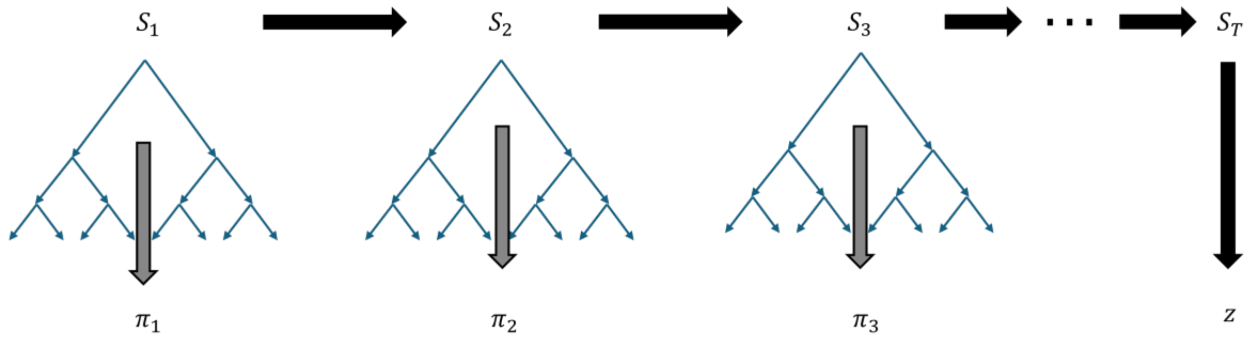

Fig.1 Self-play reinforcement learning in AlphaGo Zero

3. Interpretability and Causal Inference

Large open-source models such as AlphaZero and Stockfish. These models can make decisions in complex chess games using deep neural networks and reinforcement learning algorithms. However, due to the complexity of the internal mechanism of the model, understanding its decision-making process has become an essential but challenging task. Previous work has proposed identifying the features that have the most significant influence on decision-making by evaluating the contribution of each input feature to the model output and analyzing the middle layer activation. In the chess model, this can reveal which chessboard features (such as the layout of chess pieces and control areas) are crucial to model decision-making. However, this kind of method is inherently restrictive because of the black box problem of the deep learning model.

Applying structural causal reasoning to deep reinforcement learning is an important research direction to solve its interpretability problem. Pearl et al. put forward the structural causality model (SCM), and a graphical production model describes the causal mechanism. This model contains the mathematical relationship between variables and defines the causal direction and path between variables, as shown in Figure 2.

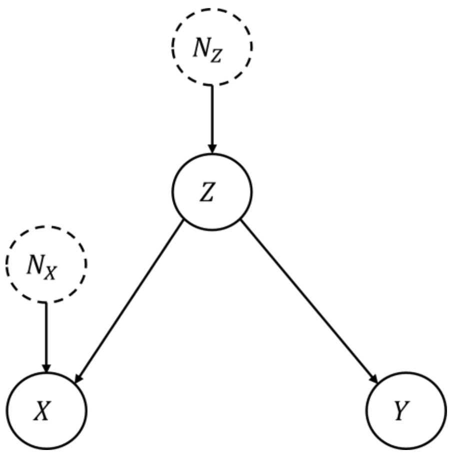

Fig. 2 the Structural causal Model with exogenous variable

SCM can clearly show which variables are the causes, which are the results, and how the causal relationship between them is transmitted. In the competitive process of board games, the overall decision-making process can be considered a hidden Markov decision-making process. For each specific decision, there are many influencing variables. In the structural causal model, a directed acyclic graph (DAG) can represent the relationship between influencing variables. Nodes in the causality diagram represent variables, and directed edges represent causality between variables. In the causality diagram, if all paths between two variables are blocked (d- separated) by a set of variables Z, then under the condition of given Z, the two variables are independent in probability. This is a fundamental concept used to judge the independence between variables in causal reasoning. In SCM, given all the direct causes (parent nodes) of a node (variable), the node is independent of all other nodes that are not its descendants. This basic assumption in SCM simplifies the complexity of causal reasoning [7].

Through causal reasoning algorithms to explain the prediction results and decision-making process, people can better understand how the model plays chess games and find potential deviations and problems. Related work has proved the possibility of improving the interpretability of deep learning by combining causal inference. In the medical field, researchers can use deep learning models to predict patients' disease risks and use causal reasoning algorithms to explain these prediction results. By combining these two methods, doctors can better understand the patient's condition and treatment effect to make a more reasonable treatment plan. In autonomous driving, researchers can use a deep learning model to identify road obstacles and pedestrians and explain how the model makes obstacle avoidance decisions according to these goals through a causal reasoning algorithm. This helps to improve the safety and reliability of self-driving cars. This paper holds that the combination of structural causal reasoning and deep reinforcement learning is the leading direction of chess-related research in solving the interpretability problem of models.

4. Conclusion

The board game is where artificial intelligence technology shows its powerful learning ability. With the continuous progress of artificial intelligence technology, from the initial method based on symbol system to the neural network of connectionism, and then to the deep reinforcement learning method combining behaviourism and connectionism. The continuous progress of artificial intelligence in board games represents the peak of artificial intelligence technology. This paper systematically introduces two mainstream chess games, including the standard algorithms of artificial intelligence technology in chess and Go, including Monte Carlo and deep reinforcement learning. Furthermore, the possibility of applying causal inference to deep reinforcement learning to solve interpretable problems is discussed. The research in this paper will provide a valuable reference for practitioners in related fields.

References

[1] Jingjing Bai, Xin Guo. The application of chessboard game based on integrated learning and UCT algorithm in mental health and emotional regulation. Entertainment Computing, 2024, 51: 100722.

[2] Shruti Priya, Shubhankar Bhadra, Sridhar Chimalakonda, et al. ML-Quest: a game for introducing machine learning concepts to K-12 students. Interactive Learning Environments, 2024, 32(1): 229-244.

[3] Florian Richoux. Injecting Combinatorial Optimization into MCTS: Application to the Board Game boop. arXiv preprint arXiv:2406.08766, 2024.

[4] Jung-Kuei Yang, Shi-Jim Yen, Serkan Kavak. Deep learning approaches to the game of Connect6. Entertainment Computing, 2024, 50: 100707.

[5] Sotetsu Koyamada, Shinri Okano, Soichiro Nishimori, et al. Pgx: Hardware-accelerated parallel game simulators for reinforcement learning. Advances in Neural Information Processing Systems, 2024, 36.

[6] Ti-Rong Wu, Hung Guei, Pei-Chiun Peng, et al. MiniZero: Comparative Analysis of AlphaZero and MuZero on Go, Othello, and Atari Games. IEEE Transactions on Games, 2024.

[7] Guido W. Imbens. Causal inference in the social sciences[J]. Annual Review of Statistics and Its Application, 2024, 11.