

From Go to Machine Learning

Yiwei Mao

May 12, 2024

Abstract

Artificial intelligence (AI) is a widely discussed concept, evolving at an unprecedented pace and scale in the 21st century. The broadness of its definition and the cryptic technology terms often mark the concept as distant to the public. However, AI is much more approachable than many may think.

Divided into five sections, this paper aims to provide an accessible introduction to the application of artificial intelligence (AI) by majorly focusing on one of its most prominent case studies, AlphaGo, and investigating other real-life applications. In the opening section, I will show a few potential questions related to AI to justify the choice of this topic and offer an outline of the paper. I will also cover the development of AI and AlphaGo's pivotal role in it. Next, I will delve into the algorithms of AlphaGo and explain its core structure. In the following section, I will explore further applications of AI in area such as playing video games and provide related analysis. In the last section, I will reflect on discussions with peers, present my findings, and propose ideas for future research and exploration.

1 Introduction

When it comes to artificial intelligence, a myriad of questions arises. How can we effectively utilize AI? What are its advantages and disadvantages compared to human capabilities? What role do humans play in using AI? These questions have been on my mind since I noticed AlphaGo while preparing my last presentation on the board game Go. Living in an 'age of AI', such questions are closely connected to our daily life. Upon closer examination, it also becomes clear that many fundamental principles behind numerous AI algorithms are closely tied to our familiar mathematical concepts, and are deeply connected with the essence of our course: to create connections. Building upon my previous presentation on the Chinese board game Go, I decided to explore the topic further in this direction.

The following section will provide an overview of key concepts related to artificial intelligence, tracing the development of AI and the emergence of AlphaGo. In the third section, the paper will explain the central structures of AlphaGo: the Convolutional Neural Network, Monticello Search Tree and reinforcement learning, and how they combine to form a gameplay-optimizing AI. While the paper will majorly focus on explaining AlphaGo, it will not be limited to it. Moving on to the fourth part, it will explore how AI can be used to play various board games and contribute to other real-life applications.

This paper aims to provide demonstrative examples of incorporating artificial intelligence to play board games, analyze AI with respect to human sample learning, and generalize the methods to more real-life applications.

2 Artificial Intelligence

2.1 From Go and Beyond

In the popular Chinese board game Go, where the aim is simply to encircle your opponent, countless clever strategies and intriguing mathematical studies have emerged. Related research has also shown that the total number of possible Go games exceeds 10^{170} , an astronomical figure far beyond the reach of current computer processing power.[1]

As Go players gain experience through pattern memorization and continuous practice, their goal is self-improvement. Yet, scholars have never stopped researching the optimal strategy in this game. In an age of modern technology, creative ideas started to emerge: can we implement new technology such as computer

programs to play Go? Various attempts were made, but success in creating a powerful program remained unachieved. Some computer programs were able to combat with human in Go. However, even the best-designed program has never won any professional players without handicaps (pieces placed on the board previous of the game given to offset the strength difference between players of different ranks, as shown in fig.1).

Faced with the astronomical possibilities of Go, until 2013, it's widely believed then that achieving an optimal playing project is improbable.

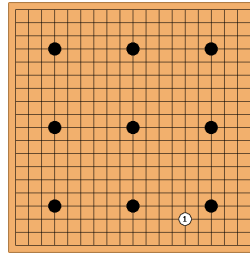


Figure 1: Handicap in situations with a higher rank White player and a lower rank Black player.[2]

2.2 Concepts and Development of artificial intelligence

Meanwhile, a new wave of technology is gradually emerging. The origins of artificial intelligence dates back to 1950, when British Mathematician Alan Turing proposed the idea “Why can’t machine use available information for problem-solving and decision-making similar to human?” However, the constraints of contemporary technology limited broader applications of this idea. Later in 1955, Logic Theorist was organized, which is considered as the first artificial intelligence program and catalyzed countless future AI researches. Over the subsequent two decades, as computers advanced in functionality and accessibility, AI research experienced rapid progress. Concepts such as natural language processing, abstract thought, and self-recognition were introduced, while their context was continually explored and developed.[3] Today, we can see the widespread and mature application of artificial intelligence across various industries. Take some most familiar things for instance: recommendation systems in music streaming apps, social media friend matching, image recognition, stock market predictions, Siri, ChatGPT, etc.

Contemporarily, the most prominent concepts of artificial intelligence are machine learning and deep learning.

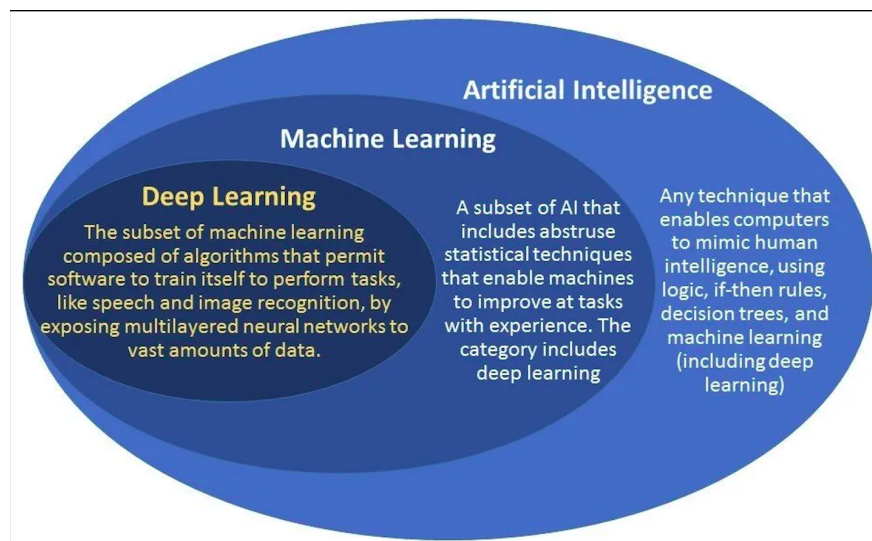


Figure 2: AI, Machine Learning and Deep Learning.

Artificial intelligence is commonly defined as a program that can sense, reason, act, and adapt like human. While machine learning is defined as a subset of AI with algorithms whose performance improves as they are exposed to more data over time, which emphasize a process similar to “human learning”. Similarly, deep learning is a subset of machine learning where the software can autonomously train itself using raw data samples, often associated with multi-layered neural networks.[4]

Understanding such concepts plays a vital importance in academic categories. However, real-life applications are often a combination of those terms. Throughout this paper, these terms will be used interchangeably depending on the context.

2.3 AlphaGo

In 2014, development of artificial intelligence intervened with the continuous study of the ancient board game Go. Google DeepMind initiated their AlphaGo project– a computer program that utilize multiple machine-learning algorithms to play Go. By October 2015, AlphaGo played against European champion Fan Hui, securing a resounding 5-0 victory, marking the first time AI triumphed over a human professional player on a full-sized board without any handicap. In 2016, AlphaGo challenged the World Champion Lee Sedol and claimed victory with a score of 4-1. Throughout this period, developers continuously refined AlphaGo’s algorithms for new challenges. In 2017, AlphaGo played against the world No.1 ranking player Ke Jie and won by a 3-0 score, solidifying its position as an exceptionally proficient gameplay program. Since then, defeating AlphaGo has been nearly impossible for humans. Presently, AlphaGo and more modified versions of it continue to selfplay to train for more optimal gameplay.[5]

3 AlphaGo

To grasp how AlphaGo works and illustrate how machine learning applies to real-world issues in artificial intelligence, it’s important to understand its approach.

In summary, the first step of AlphaGo is to learn from expert samples.

Initially, AlphaGo learns from expert game examples. Analyzing thousands of professional games reveals patterns, such as common opening moves and subsequent steps. For instance, out of roughly 20,000 professional games, around 5,541 start from the specific point shown in fig.3. While the second popular move it demonstrated in fig.4.

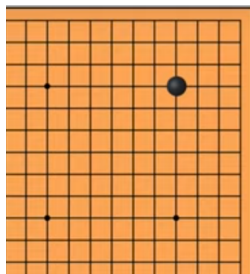


Figure 3: The most famous opening.[6]

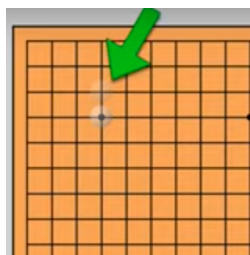


Figure 4: The most famous second step.[6]

However, merely observing and following popular moves is not feasible. In Go, each move depends on both the board's state and the opponent's move. Therefore, experts must find a method for AlphaGo to simulate human professional players' strategic decisions based on the current board configuration.

3.1 Convolutional Neural Network

Developers of AlphaGo choose to use the convolutional network, a type of machine learning model commonly employed for prediction simulations based on samples.

Coincidentally, my classmate Lizi covered neural networks in the previous presentation and demonstrated the mathematical details behind them explicitly. Therefore, this paper will provide specific introduction to the general structure of neural networks, but rather focus on the network implemented in Alpha Go—the Convolutional Neural Network.

3.1.1 Convolutional Layer

One crucial structure of the convolutional neural network is its convolutional layers. They're commonly employed in image processing, especially in tasks like image recognition and classification. Even a single convolutional layer can function as a convolutional filter, and be used to process image. These filters are typically simple 3x3 matrices, familiar to anyone who has learned linear algebra. Here is an example of using the filter shown in fig.5 to sharpen a photo.

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Figure 5: Convolutional filter to sharpen image.[7]

In the digital world, colors are represented as numbers assigned to pixels. If we zoom in on a section of the image enclosed by a red bracket in fig.6, we can see in fig.7 it appears as a 3x3 matrix. This matrix's color is demonstrated by:



Figure 6: Target picture to sharpen.[7]

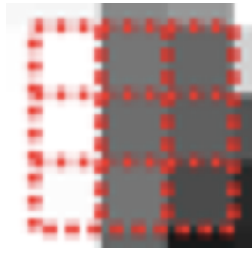


Figure 7: Image clip.[7]

$$\begin{bmatrix} 255 & 118 & 103 \\ 255 & 110 & 96 \\ 252 & 117 & 75 \end{bmatrix}$$

where larger numbers correspond to lighter color and smaller numbers to darker ones.

Then, by multiplying the rows of this matrix with the columns of the convolutional filter and summing the results,

$$255 \times 0 + 118 \times (-1) + 103 \times 0 = -118$$

$$255 \times (-1) + 110 \times 5 + 96 \times (-1) = 199$$

$$252 \times 0 + 117 \times (-1) + 75 \times 0 = -117$$

$$(-118) + 199 + (-117) = -36$$

The calculation obtain a single number -36, representing the color projected onto the center block of the matrix in the sharpened image.

After applying this process across the entire image, we can get a sharpened version of it fig.8. This process can also be understood as adjusting the colors of surrounding pixels and projecting the result onto the central pixel. Essentially, the math behind the convolutional filter is simple linear transformation and projection.



Figure 8: The sharpened image.[7]

Many other convolutional filtering matrices are also often used in image processing.

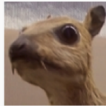

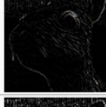

Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	

Figure 9: Convolutional filters examples.[8]

3.1.2 Convolutional neural network in AlphaGo

AlphaGo employs the convolutional neural network to simulate professional player in the following way:

1. Import the current state on the board as a 19*19 two-dimensional image through convolutional layers
2. Each single state on the board in each existing professional's games can be recorded as a sample and used to train the network
3. The input goes through 13 layers of neural networks, including the convolutional part
4. Through large amount of sample training, the network can output the positions that are considered mostly likely to be played on and their relative probability

In this convolutional network, developers have experimented with different numbers of layers—3, 6, 10, and 12. After testing, they found that 12 layers offer the best balance between prediction accuracy and computer capacity constraints. When combined with the image processing layer, this brings the total to 13 layers.[9]

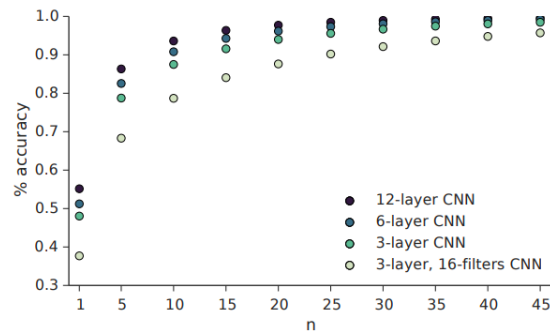


Figure 10: Probability that the expert's move is within the top-n predictions of the network. The 10 layer CNN was omitted for clarity[9]

fig.11 demonstrates a simplified version of the network. The current board state is represented by two matrices for Black and White. Each intersection is either occupied or not, so the numbers in the matrices are relatively 1 or 0. After passing through the network, it predicts the next move for White.

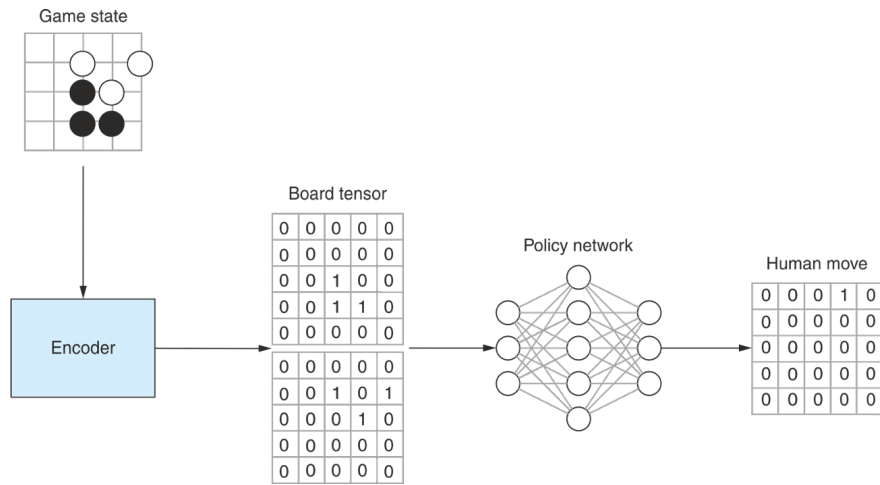


Figure 11: Simplified demonstration of predicting the next move of White

In fig.12, a clip of AlphaGo playing against FanHui, we can see the potential next moves mimicking human behavior, along with their probabilities. The highest probability move is marked in red circle with a 60 percent chance.

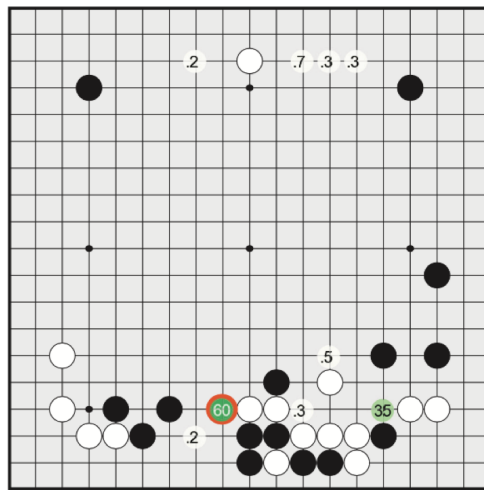


Figure 12: Output prediction in real game[10]

3.2 Monticello Search Tree

After implementing the convolutional network, AlphaGo is capable of playing, but is still limited to winning amateur Go players. The main issue lies in its inability to plan ahead. While professional Go players think several moves ahead, the neural network only focuses on the best immediate move. This can lead to unfavorable future patterns and eventual losses. To address this, a structure called the Monticello Search Tree is implemented. It looks ahead by branching out and exploring possible future moves, as shown in fig.13

However, AlphaGo is capable of only exploring a few branches of them, not all possibilities. Therefore, it is important to choose which paths to detect. First, AlphaGo aim to search the steps that might be considered proficient. Ideally, it prioritizes moves that expert players might make, avoiding paths that lead to further losses.

Again, consider the 'ladder' pattern in Go, where one player dominates the board for example. AlphaGo will have to recognize such patterns as disadvantageous and decide there is no need to further detect its future

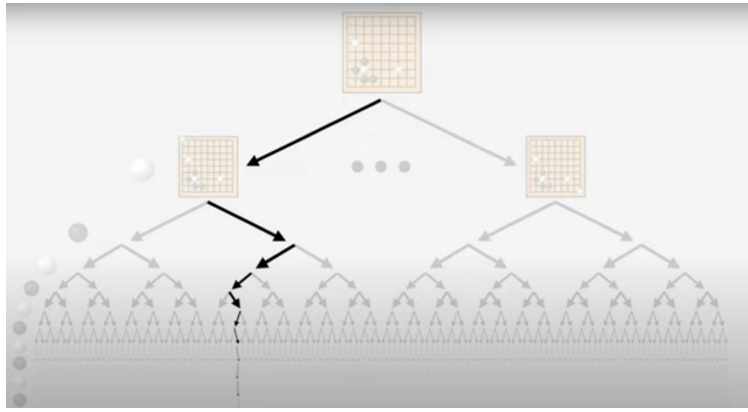


Figure 13: Search from Existing State[6]

moves. However, it is only simple to judge from human experience about this pattern, but an algorithm is needed for the computer to recognize bad patterns successfully.

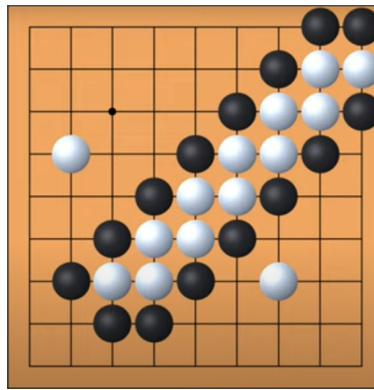


Figure 14: Ladder Pattern[6]

To achieve this, we employ a value function. By simulating numerous games from similar situations, we can get the likelihood of winning. For example, generating 100 games, if a ladder pattern emerges, with no chance of winning for White ($\frac{0}{100}$), AlphaGo marks it as a bad pattern. However, before the ladder forms, White's winning chance is generated to be $\frac{57}{100}$, indicating a more balanced position. This data helps AlphaGo make better decisions during gameplay.

3.3 Reinforcement Learning

Once AlphaGo implements the best next moves and forecasts preferred paths, it can play while thinking ahead. After that, reinforcement learning happens. Experts set two AIs to play against each other, also known as the process of self-play.

In fig.15, winning paths labeled as green is reinforced, losing red paths are penalized, and the value is updated to be 0.5. While simulations in reality produce more games. After numerous games are generated, the winning paths are reinforced, while the losing routines are penalized. By simulating a myriad of games, the outcomes might change, and AlphaGo updates the value of positions on the board accordingly. Also, the training opponent's strategies are often changed to mimic different opponents' styles.

However, when looking into future plays, though the Monticello Search Tree can generate 30 million possibilities, it is still a small number compared to the huge possible games in Go. There must be states that remain undetected that it might encounter in future games. To tackle this, experts bring in another neural network trained on the games to evaluate the value of unseen positions. This way, their value can be at least gained for further detection and analysis.

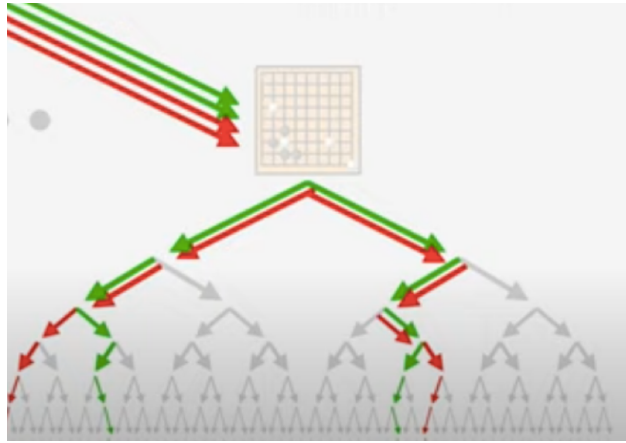


Figure 15: Reinforcement learning[6]

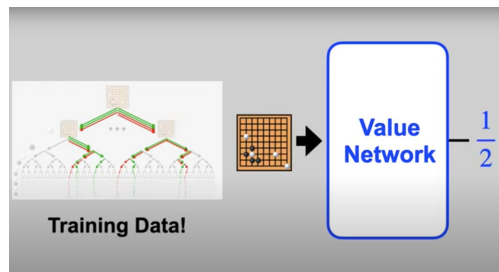


Figure 16: Value Network of AlphaGo[6]

To sum up, AlphaGo mimics skilled human players using convolutional neural networks and the Monte Carlo Tree Search to plan ahead. It prioritizes likely paths while preventing choosing less promising ones based on its value function, maximizing its detection capabilities. Through continuous self-play and reinforcement learning, AlphaGo refines its strategies and updates its value function. It also employs a separate neural network to estimate the value of potential states that remain undetected based on gathered samples. During gameplay, AlphaGo constantly generates and evaluates new paths, making its moves within a time limit ranging from a few seconds to under 150 seconds, aiming for a balance between maximal possible game detection and game time constraints.

AlphaGo demonstrates AI's potential to tackle incredibly complex games. It highlights the advantages of AI over humans in board games, showing how AI can learn from vast amounts of data quickly, unlike humans who typically require more time to learn methodically. Additionally, AI tends to make fewer mistakes once its program is established. The development of AlphaGo reflects a strategy for implementing machine learning models and combining them to solve complex problems effectively.

4 AI application

4.1 AI Platform Games

While AlphaGo represents a significant advancement in AI research, a more accessible way to utilize AI around us is use it to play video games. To offer more simple examples of applications and analyze different cases, a few attempts are demonstrated in this section.

4.1.1 Super-mario

In a video posted on June 13, 2015, a player demonstrate an astonishing way of playing Super Mario using neural evolution with an Evolving Neural Networks. [11]Surprisingly, no human input was used in this scenario, and the training system operated as follows:

1. Mario remained stationary until a limited time, after which the game reset.
2. The network randomly pressed buttons, such as continuously pressing the right button, causing Mario to move right until falling.
3. Fitness points were awarded based on how far and how fast Mario progressed, and networks that yielded the highest points were selected for further generations.
4. Neurons (nodes) were randomly generated, and axons(lines) were formed, to simulate the structure of the human brain, evolving through modification.
5. Self-evolution and training occurred.[12]

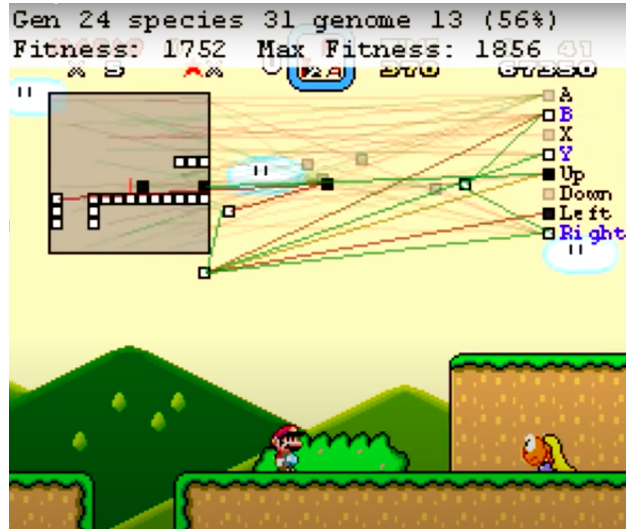


Figure 17: AI Super-mario[11]

After 24 hours and 34 generations of self-improvement (each generation consists of multiple games), Mario successfully completed the game.

The video also demonstrated a variant of a neural network capable of generating game samples by itself, indicating machine learning models can be highly adaptable depending on the required situation.

Another aspect highlighted in the video is AI Mario's implementation of spin-jumping, a technique enabling Mario to eliminate opponents while jumping. This efficient method of movement was discovered by human players during gameplay and also found by AI through self-evolution without any human input. This is what people talk about when they are referring to "AI has the potential of learning like humans."

4.1.2 Other Games

Similarly, AI is used to play different types of games, like SNAKE!, Tetris, Mario Cart, and Flappy Bird.

In SNAKE! and Tetris, the game's layout and rules are input, and rewards are given based on the snake's length and survival time, or the highest score achieved. Afterwards, the game generates its own samples, and convolutional neural networks are trained to predict the best button to press in each situation.[13][14]

In Mario Cart, the map is input as pixels, and go several transformations based on human gameplay data, eventually producing the most likely button presses by humans in those situations. When faced with the issue of AI-trained carts getting stuck in certain spots repeatedly, the player chooses to manually take over and extract his playing sample for further training to address the problem.[15]

For Flappy Bird, the approach is quite similar. However, since it's an open-source game, 100 birds are released simultaneously in one game, and the bird that survives the longest is rewarded, which can also refer to an evolutionary method.[16]

Since most of these games follow similar patterns, only their key functions are discussed here. For more detailed explanations, please refer to the relevant links.

4.2 AI Problem Solving Pattern

Despite using different setups and selecting varied types of neural networks, a common pattern emerges that can be applied to solving AI problems can be concluded. This pattern is not limited to video games or relying solely on neural networks, it provides a more typical approach to any AI problem solving.

1. Start by importing the game setup, which includes the map or rules, or the initial condition of other types of problems.
2. Choose appropriate Machine Learning models that suit the specific problem based on its characteristic. (For example, to implement a recommendation system rather than a gameplay, one may focus on the clustering model often used for categorization.)
3. Train the selected model.
4. Make observations, make necessary modifications, and return to previous steps for update.

4.3 Human Sample and Self Generated Sample

In previous examples, the use of evolutionary neural networks to play Super Mario without human input has been seen. While there exist other approaches, such as using AI trained with human input.[17] This AI learns jumping skills quicker and reaches the end of the game faster. However, different from the previous attempt, it needs a lot more human input, which is challenging to implement in some situations. This leads to an interesting question: Is it better for AI to learn from human samples or generate its own? This question is relevant not just for Super Mario, but is also closely connected to implementing AI in games like Go. For instance, after AlphaGo, scientists enhanced its structure and got an improved version with self-generated sample: AlphaZero.

AlphaZero majorly differs from AlphaGo in the following ways[18]:

1. Potential outcome: AlphaGo focuses on estimating and improving the chances of winning, assuming only two outcomes: win or loss. On the other hand, AlphaZero considers a broader range of potential outcomes, including draws and other possibilities.
2. Human sample: AlphaZero does not augment training data and does not transform the board position during MCTS. It begins with random generated games and use them to continue the training.
3. Single reinforcement network: In AlphaGo, self-play games were generated by the best player from previous rounds. After each training round, if the new player performed better than the best player by at least 55 percent, it replaced the best player. In contrast, AlphaZero continuously updates a single neural network without waiting for iterations to finish.

In just three days, AlphaZero learned and mastered the game of Go from scratch and beat the original AlphaGo in 100 games. While this shows a very successful implementation of AI without human sample over with human sample, it should not be ignored that the development of AlphaZero is largely related to the structure of AlphaGo.

To put summarize, using human data may be faster and easier initially, but it can be limited to existing strategies. On the other hand, self-generated data requires complex coding and algorithms, especially for challenging games. Regarding the ongoing debate, I believe it's not about choosing one approach over the other, but rather about smart decision making of human when setting up AI, and optimal selection based on the current resource. We should pick the method that fits and remain open to modification to the other method in the future, sometimes even based on the current attempt, as demonstrated in the case of AlphaGO and AlphaZero.

4.4 Human's Role

When talking about the advantages of AI compared to humans, people often highlight its learning speed and capacity. For instance, AlphaZero can master Go, even surpass AlphaGo's Go-playing skills in just few day once all functions are implemented, a speed humans can never achieve. Yet, people shouldn't see AI advancements as diminishing human capabilities. In fact, umans play indispensable roles in AI development.

Initially, humans must set up the game for AI input, as AI lacks consciousness or problem-initiating abilities. Moreover, constructing and modifying AI require human intervention at various stages, as been previously demonstrated through nearly every example.

While AI may replace some human functions, we must acknowledge human importance, consider utilizing the strengths of both humans and AI, and not be limited by technology, but rather make use of it wisely.

5 Conclusion

5.1 Presentation Reflection

Through this presentation, I managed to reflect clearly on many unresolved questions from previous discussions. One classmate has asked if there is potential to predict the game-winner by observing only the initial minimum steps, while the concept of the value function illustrates this idea well. It also offers an insight of possible predictions always exist, but later steps are also essential to the outcomes. AlphaGo also demonstrates a balance between using the value function or going down to detect more possibilities in certain branches. Another question also arose: can humans beat AI with unconventional strategies? The match between AlphaGo and Lee Sedol explained in this presentation showed that unexpected moves can indeed confuse AI systems, leading to potential victories for humans. However, AlphaGo's subsequent improvements, particularly in its value function and training with more data, minimized this vulnerability.

Reflecting on the presentation itself, I noticed many classmates expressed surprise at AI's ability to play games like humans and started pondering about its application in other fields. It's gratifying to see how this presentation expanded people's understanding of AI's capabilities and encouraged attempts to try. Those are also the most essential points I want to convey through this presentation. Additionally, the comparison between AI trained on human data versus AI generating its own data generated considerable interest. This prompted me to include a section comparing applications of these two approaches in this paper. It's fascinating how reflections can light up important topics that even the presenter ignored, inspiring further exploration and innovation, and I will carry out those ideas in my future investigation and presentation.

5.2 Discussions and Related Further Research

During our discussions, various ideas for furthering our project research emerged. One classmate raised an interesting question: are certain types of games exceptionally challenging for AI to master? It's fascinating to see how even action-packed shooting games, which heavily rely on quick reflexes and intricate map layouts, can be tackled by AI with today's advanced technology. On the other hand, games that involve artistic elements and design tend to be especially challenging for AI. Nowadays, AI programs can even create artwork that meets specific criteria, leading to philosophical debates about the authenticity of AI-generated art and its ethical implications. These discussions are thought-provoking and worth exploring further, while we can also aim to detect deeper into the mathematics behind those attempts.

More recently, when I learned about the presentation of Prime and how it is utilized in encryption, I couldn't help but wonder: could AI potentially crack codes? If such a scenario occurs, what measures could we take to safeguard our information against artificial intelligence? This brought me to think about the variation code used in various software and applications, verifying the identity of human. How are they specifically designed to block potential AI hacking attempts? This could be another intriguing direction for exploration.

Also, Professor Li proposed the idea about using AI, like ChatGPT, to develop methods for computer gaming or other applications based on explicit requirements. If such attempts are applicable, does this mean a further step to the autonomy of AI? This concept of internally generating AI raises astonishing questions about the autonomy of AI and whether we're getting closer to a true self-governing AI.

Under such a broad topic like AI, there's still so much left to discover, and countless challenges await for their resolution. Simultaneously, countless solvable practical problems are waiting for people to apply the existing applications already known.

References

- [1] Allis, Victor (1994).“Searching for Solutions in Games and artificial intelligence.”. Ph.D. Thesis, University of Limburg, Maastricht, The Netherlands.<http://fragrieu.free.fr/SearchingForSolutions.pdf>
- [2] “Handicap”(2020). Sensei’s Library, <https://senseis.xmp.net/?Handicap>.
- [3] Anyoha, R. (2017).“Can Machines Think?”. Science in the News,<https://sitn.hms.harvard.edu/flash/2017/history-artificial-intelligence/>.
- [4] Brennan, W. (2023).“ artificial intelligence vs. Machine Learning vs. Deep Learning: What’s the Difference?”. builtin,<https://builtin.com/artificial-intelligence/ai-vs-machine-learning>.
- [5] “ Timeline of AlphaGo.”(2023). Timelines Wiki,https://timelines.issarice.com/wiki/Timeline_of_AlphaGo.
- [6] Graphics in 5 Minutes.“Reinforcement Learning: AlphaGo”.YouTube, uploaded by Graphics in 5 Minutes, 14 Agu. 2023,<https://www.youtube.com/watch?v=4PyWLgrt7YY>.
- [7] Powell, V. (n,d).“ Image Kernels Explained Visually”. setosa,<https://setosa.io/ev/image-kernels/>.
- [8] Salerno, S. (2018).“ A Gentle Introduction To Convolution Filters.”. Medium,<https://medium.com/skylar-salernos-tech-blog/image-convolution-filters-explained-c878f1056e78>.
- [9] Maddison, Chris J., et al. (2014).“Move Evaluation in Go Using Deep Convolutional Neural Networks.”. arXiv:1412.6564 [cs.LG],<https://doi.org/10.48550/arXiv.1412.6564>.
- [10] Silver, David, et al. (2016).“Mastering the Game of Go with Deep Neural Networks and Tree Search.”. Nature, vol. 529. ResearchGate, https://www.researchgate.net/publication/292074166_Mastering_the_game_of_Go_with_deep_neural_networks_and_tree_search.
- [11] Sethbling.“MarI/O - Machine Learning for Video Games.”.YouTube, uploaded by Sethbling, 13 Jun. 2015,<https://www.youtube.com/watch?v=qv6UV0Q0F44>.
- [12] Stanley, K. Risto, M.“MEvolving Neural Networks through Augmenting Topologies.”.The MIT Press Journals Evolutionary Computation, vol. 10, no. 2, n,d, <https://nn.cs.utexas.edu/downloads/papers/stanley.ec02.pdf>.
- [13] Chrispresso.“AI Learns to play Snake!”.YouTube, uploaded by Chrispresso, 22 Sep. 2019,<https://www.youtube.com/watch?v=vhi04WsHA6c>.
- [14] Greer Viau.“Building an AI to MASTER Tetris”.YouTube, uploaded by Greer Viau, 1 Apr. 2020,<https://www.youtube.com/watch?v=1yXBNKubb2o>.
- [15] Sethbling.“MariFlow - Self-Driving Mario Kart w/Recurrent Neural Network.”.YouTube,uploaded by Sethbling,5 Nov, 2017,https://www.youtube.com/watch?v=Ipi40cb_RsI.
- [16] Code Bullet.“A.I. Learns to play Flappy Bird.”.YouTube, uploaded by Code Bullet, 21 Dec. 2018,<https://www.youtube.com/watch?v=WSW-5m81RMs>.
- [17] Chrispresso.“AI Learns to Play Super Mario Bros!”.YouTube, uploaded by Chrispresso., 19 Aug. 2020,https://www.youtube.com/watch?v=CI3FRsSAa_U
- [18] Maddison, Silver, D, et al. (2017).“MMastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm.”. arXiv:1712.01815,<https://arxiv.org/pdf/1712.01815>.