Adaptive Blackjack Player Using Monte Carlo Reinforcement Learning

Ranjitha Umesh
MAI, Faculty of Computer Science
Technical University of Applied Science, Würzburg-Schweinfurt
ranjitha.umesh@study.thws.de

***Abstract*—Blackjack is a classic casino card game enjoyed by millions, where players aim to have a hand value closest to 21 without exceeding it, competing against the dealer. A major challenge in Blackjack is developing strategies that can adapt to the dynamic nature of the game and varying rules. In this research, we implemented a self-learning Blackjack player using reinforcement learning techniques to tackle this problem. Our goal is to create an intelligent agent capable of dynamically adapting its strategy to maximize profits. We used a Monte Carlo control algorithm to train the agent through extensive simulated gameplay. The agent learns to make optimal decisions and adjusts actions based on game scenarios. This study shows how reinforcement learning can create effective agents for strategic card games like Blackjack, demonstrating significant advancements in adaptive strategy development.**

***Index Terms*—Blackjack, Reinforcement learning, Monte Carlo control, Hand value**

I. INTRODUCTION

Blackjack, also known as 21, is among the most popular card games in casinos worldwide. The game's goal is straightforward: players aim to get a hand value as close to 21 as possible without exceeding it and compete against the dealer. Each player starts with two cards and can choose actions like hitting, standing, or doubling down. Face cards are worth 10 points, Aces can be 1 or 11 points, and all other cards are worth their face value.

In this study, we developed a self-learning Blackjack player using reinforcement learning techniques, specifically the Monte Carlo method. We aimed to create an intelligent agent that adapts its strategy through continuous gameplay to maximize profits. The agent begins with the basic strategy, which provides optimal moves based on the player's hand and the dealer's visible card. We then integrated the Hi-Lo card counting method to help the agent track high and low cards, adjusting its strategy and bets based on the remaining deck composition. Additionally, we explored the impact of rule variations like Dealer Hits Soft 17 and Double Anytime, requiring the agent to adapt its strategy accordingly.

Traditional strategies didn't adapt to changing game conditions. Our study aims to address critical questions such as: How do rule changes like Dealer Hits Soft 17 or Double Anytime affect learning agents? How well can these agents adapt to changing conditions compared to static strategies? By exploring these questions, our study aims to better understand how reinforcement learning can be used in Blackjack and develop successful gameplay strategies.

Through these implementations, our self-learning Blackjack player learns to make optimal decisions and adapts to various game conditions, showcasing the power of reinforcement learning in complex, strategic environments like Blackjack. This study highlights how machine learning can enhance strategic decision-making in dynamic environments, offering practical insights for both casual and professional Blackjack players. The findings also contribute to broader applications of reinforcement learning in other domains where adaptive strategies are crucial for success.

Furthermore, the integration of reinforcement learning with card counting techniques represents a significant advancement in automated gameplay. By continuously analyzing game outcomes and refining its strategies, the agent can effectively respond to real-time changes in the game environment. This level of adaptability is particularly beneficial in Blackjack, where the dynamic nature of the game can render static strategies suboptimal. Our research not only demonstrates the feasibility of using machine learning in card games but also provides a framework for future studies to explore more sophisticated algorithms and techniques in similar settings.

Ultimately, our study serves as a stepping stone towards more advanced and intelligent gaming agents capable of outperforming traditional human strategies. As technology continues to evolve, the potential applications of such self-learning systems extend beyond gaming, offering valuable insights into fields like finance, healthcare, and autonomous systems where decision-making under uncertainty is paramount. Through rigorous experimentation and analysis, we aim to push the boundaries of what is possible with reinforcement learning and pave the way for innovative solutions in various industries.

II. Related Work

To develop our self-learning Blackjack player, we drew from several important studies that greatly influenced reinforcement learning and the strategy.

Our implementation of card counting strategies in the Blackjack agent by [1]. Thorp introduced the Hi-Lo card counting system, a revolutionary method that significantly improved player's ability to keep track of the ratio of high to low cards remaining in the deck. This technique has become a cornerstone of Blackjack strategy and was crucial in developing our reinforcement learning agent. By integrating Thorp's Hi-Lo system, our agent is better equipped to make informed decisions, dynamically adjusting its strategy to optimize performance. In developing our self-learning Blackjack player, we utilized fundamental principles from [2]. We used the Monte Carlo control algorithm to optimize the agent's strategy by learning from complete game episodes. The book explained that the epsilon-greedy policy, which balances exploration and exploitation, was key to enhancing our agent's performance. This knowledge helped us to implement and refine our Blackjack agent, ensuring a robust and effective learning process. Our research builds upon the work [3] They used Monte Carlo reinforcement learning techniques to develop optimal Blackjack strategies through extensive simulations. Their research provided a foundational framework that guided our enhancements, enabling us to create a more robust and adaptive self-learning Blackjack agent.

III. Methods and Implementation

A. Introduction to the Implementation

We created a simulated Blackjack environment to develop our self-learning Blackjack player. We defined the possible actions and card values and incorporated a learning mechanism to train the agent. The agent learns from its experiences by adjusting its strategy based on the outcomes of each game, gradually refining its approach to converge on an optimal policy. Rewards play a crucial role in this learning process, with positive rewards for winning hands and negative rewards for losing hands. This helps the agent to identify and reinforce successful strategies while minimizing losses. Key aspects of this implementation included integrating the basic Blackjack strategy and the Hi-Lo card counting system, exploring rule variations.

B. Blackjack Environment Setup

To create a self-learning Blackjack player, we started by setting up a simulated environment for the game. This environment uses a standard deck of 52 cards, where face cards are worth 10 points, ace cards can be either 1 or 11 points, and all other cards are worth their face value. The agent can perform three actions: hitting, which involves drawing another card; standing, which means keeping the current hand and ending the turn; and doubling down, which involves doubling the bet and drawing one final card. The game starts by dealing two cards each to the player and the dealer. The agent then decides on its actions based on the game's current state. The environment is responsible for shuffling and dealing cards, evaluating hand values, and executing the chosen actions.

C. Reinforcement Learning Techniques

Our approach uses Monte Carlo control to train the Blackjack agent. It involves estimating the value of each state by averaging the returns (rewards) received after visiting that state across multiple games. This approach includes both First-Visit Monte Carlo, which averages returns after the first visit to a state, and Every-Visit Monte Carlo, which averages returns every time the state is visited. Monte Carlo Control is then employed to refine the policy by evaluating state-action pairs, ensuring the agent explores all possible actions and avoids getting stuck in less effective strategies.

We update state-action values, or Q-values, after each game episode with the following formula:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left(G - Q(s,a) \right)$$

Q(s,a) is the action-value function, which represents the expected cumulative reward of taking action a in state s G represents the total discount reward, and alpha is the learning rate.

$$G = \sum_{t=0}^{T} \gamma^t R_{t+1}$$

T is the final time step of the game. Gamma is the discount factor, which we set to 1 in this case, meaning there is no discounting of future rewards.R_{t+1} is the reward received at time step $t+1$.

At the end of each game, the agent receives a reward based on the game outcome. These rewards help to adjust the value of each state.

Balancing exploration and exploitation is essential, and our agent achieves this through an epsilon-greedy policy. This policy helps the agent balance between exploiting known best actions and exploring new ones, occasionally making random choices to discover potentially better strategies. In practice, the agent plays numerous games of Blackjack, generating episodes by recording sequences of states, actions, and rewards. After several episodes, the agent updates its policy based on cumulative rewards, adjusting strategies that frequently lead to winning outcomes.

D. Rule Variations

Our implementation also examines the impact of two rule variations Dealer Hits Soft 17 and Double Anytime. In the Dealer Hits Soft 17 rule, the dealer must take another card when holding a soft 17 (a hand with an Ace counted as 11). This increases the dealer's chances of improving their hand and raises the risk of busting. The Double Anytime rule allows players to double their bet at any point during their turn, not just on the initial two cards. This added flexibility can result in higher profits if used wisely. The agent learns to recognize the best moments to double down, maximizing potential gains.

IV. Performance Assessment and Validation

This section aims to assess how well our self-learning Blackjack agent performs. We will measure the agent's effectiveness and accuracy by examining important metrics like the win rate, loss rate, tie rate, and average reward.

We ran comprehensive simulations with 100,000 training games to evaluate our Blackjack agent's performance. Using a standard deck of 52 cards, we tested the agent's ability to make decisions such as hitting, standing, and doubling down. We examined its performance under different strategies and rule variations in a controlled environment. After training, we assessed the agent's effectiveness and consistency by running an additional 10,000 games.

TABLE I
PERFORMANCE METRICS FOR DIFFERENT BLACKJACK STRATEGIES

Configuration	**Win Rate (%)**	**Loss Rate (%)**	**Tie Rate (%)**
Basic Strategy	41.50	50.74	7.75
Card Counting	42.78	48.30	8.92
Rule Variation (Soft 17)	41.99	50.89	7.31
Rule Variation (double anytime)	42.00	50.94	7.05

From the above table comparing the basic strategy with card counting reveals that card counting significantly enhances the agent's performance. The agent achieves a higher win rate and a lower loss rate, demonstrating the advantage of integrating the Hi-Lo card counting method into its decision-making process. Comparing the basic strategy to the

Dealer Hits Soft 17 variation, the win rate increased slightly while the loss rate stayed similar, indicating a modest impact on performance. The Double Anytime variation showed a more significant win rate improvement with a similar loss rate. This shows that giving the agent the flexibility to double its bet at any point during the game leads to higher profitability.

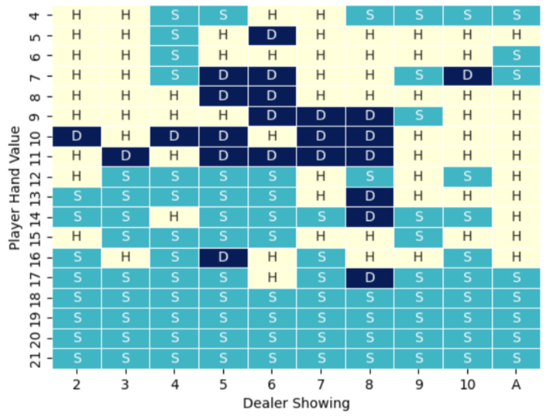

Fig. 1. Optimal policy for basic strategy (Ace As 1)

In (Fig. 1), the optimal policy for a Blackjack player is shown when an Ace is counted as 1. Each cell in the heatmap indicates the best action for the player to take in that particular scenario. It is determined by the expected outcomes calculated during the training phase. The strategy varies depending on the dealer's visible card and the player's hand value, adjusting to minimize losses and maximize gains.

The strategy in (Fig. 2) shows how the player's optimal decisions change when an Ace is valued at 11, offering more flexibility in hand values and potential actions.

(Fig. 3) presents a 3D visualization of the maximum Q-values for each state in the Blackjack game after 100,00 games. The Q-value represents the expected cumulative reward that the agent can achieve from a given state.

(Fig. 4) provides a 3D representation of the top Q-values for each state in the Blackjack game after 100,000 games. The Q-value reflects the expected

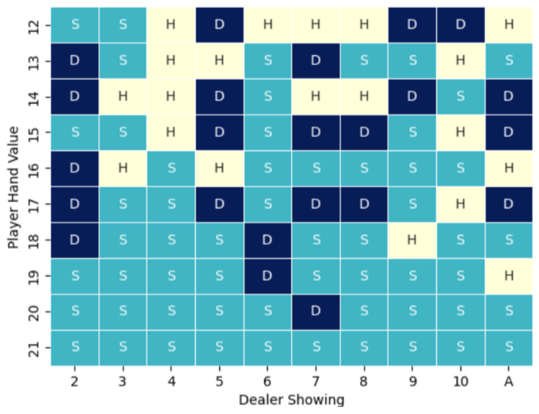

Fig. 2. Optimal policy for basic strategy (Ace As 11)

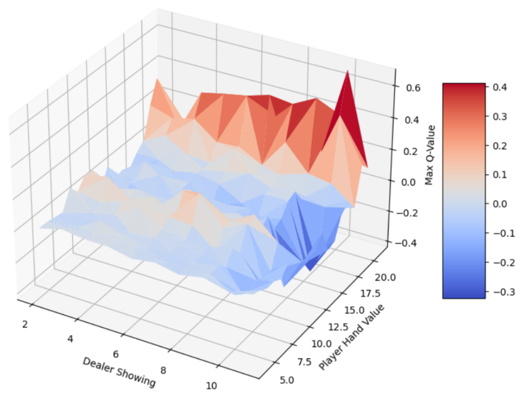

Fig. 3. Visualization of maximum Q value optimization function across 10,000 games for Each State

total reward that the agent can achieve from a specific state by following the best possible strategy.

In (Fig. 5), the results indicate that both rule variations lead to effective learning of Blackjack strategies, with the Dealer Hits Soft 17 and Double Anytime rule providing a marginally better performance.

In Figure 6, we analyze the effect of different Gamma (γ) values on the learning performance of the Blackjack agent. γ is the discount factor in reinforcement learning, determining the importance of future rewards. As the number of games increases, the win rates for all gamma values stabilize, indicating that the agents have reached a steady

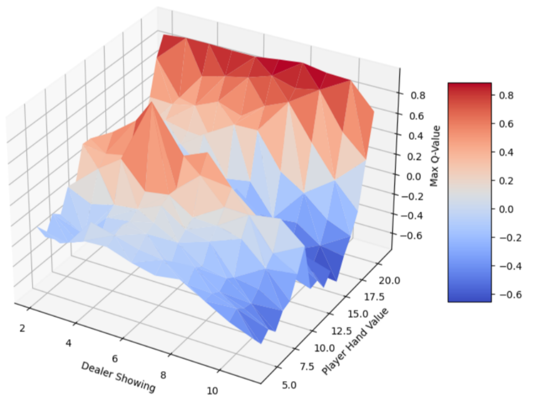

Fig. 4. Visualization of maximum Q value optimization function across 1,000,00 games for Each State

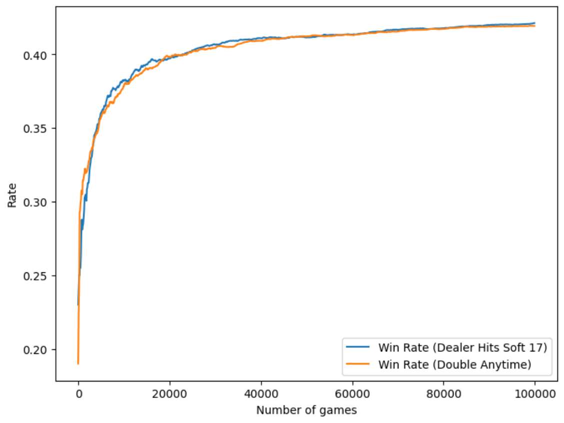

Fig. 5. Visualization of Win Rates vs Number of games for rule variations (Dealer Hits Soft 17 and Double Anytime)

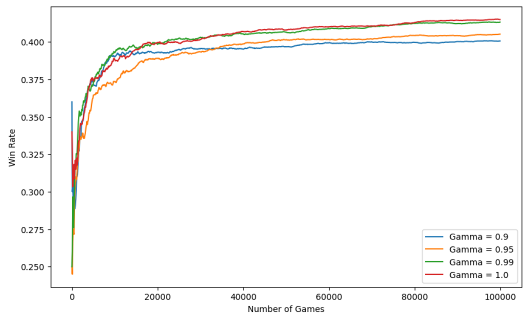

Fig. 6. Visualization of Win Rates vs Number of Games for Different Gamma Values

performance state.

V. Conclusion and Future Work

Through these implementations, our self-learning Blackjack player learns to make optimal decisions and adapts to various game conditions, showcasing the power of reinforcement learning in complex, strategic environments like Blackjack. This study highlights how machine learning can enhance strategic decision-making in dynamic environments, offering practical insights for both casual and professional Blackjack players. The findings also contribute to broader applications of reinforcement learning in other domains where adaptive strategies are crucial for success.

Furthermore, the integration of reinforcement learning with card counting techniques represents a significant advancement in automated gameplay. By continuously analyzing game outcomes and refining its strategies, the agent can effectively respond to real-time changes in the game environment. This level of adaptability is particularly beneficial in Blackjack, where the dynamic nature of the game can render static strategies suboptimal.

Ultimately, our study serves as a stepping stone towards more advanced and intelligent gaming agents capable of outperforming traditional human strategies. As technology continues to evolve, the potential applications of such self-learning systems extend beyond gaming, offering valuable insights into fields like finance, healthcare, and autonomous systems where decision-making under uncertainty is paramount. Through rigorous experimentation and analysis, we aim to push the boundaries of what is possible with reinforcement learning and pave the way for innovative solutions in various industries.

References

[1] Thorp, Edward O. Beat the Dealer: A Winning Strategy for the Game of Twenty-One. Random House, 1966.
[2] Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA, 2 edition, 2018.
[3] Srinivasaiah, R., Biju, V. G., Jankatti, S. K., Channegowda, R. H., and Jinachandra, N. S. “Reinforcement learning strategies using Monte-Carlo to solve the blackjack problem.”International Journal of Electrical and Computer Engineering (IJECE) 14.1 (2024): 904-910.