Designing Reinforcement learning approaches to Blackjack Game

Bibin Babu
MAI, Faculty of Computer Science and Business Information System

University of Applied Sciences Würzburg-Schweinfurt Germany
bibin.babu@student.fhws.de

***Abstract* — Twenty one also known as Blackjack is the one of the popular boardgames in betting, has undergone mathematical analysis to determine its Basic Strategy. Even though this method has been demonstrated to be beneficial, it prompts the concern of what an ideal technique can be learned via repetition. Could a gamer acquire that tactic via repetitive trials of victory and failures? Using the Monte Carlo Technique, this study answer those questions with a method to win against the casino with two rule changes from the standard game rules, concluding that this constructive approach give outcomes that are comparable to Basic Strategy and complete point count system mentioned in [1]. Effect of different learning algorithms and policy changes due to rule variations also studied in this paper.**

Keywords — Blackjack, Monte Carlo Simulation, Temporal differences learning, Q learning.

I INTRODUCTION

Twenty one appears to be a straightforward strategy game at first view. The object of the game is to go as near to 21 avoiding going over while the casino plays without plan and continues to add cards till at minimum 17. The point with the greater value triumphs as long while neither reaches 21. But there are a number of tactics that were refined through centuries to provide card gamers the best chance of winning. Frequently, amateurs are unaware of these strategies while experts completely understand them. In order to create Basic Strategy, the most well-known way to play Twenty one were merged. It was mostly developed by the simple convoluted process, even if some of its elements were drawn from mathematical thinking. So the issue is, Could a model, in other terms, simulate a new player learning the game and eventually produce outcomes that like Basic Strategy? This study would first analyse the game of Twenty one to see if this is conceivable. Despite the unambiguous approach this article has chosen to achieving the objective of Basic Strategy, with exceptions to the game rules that must be discussed before moving on to the tactics. This will set up a discussion of the Monte Carlo Method, a technique employed in this work to attain the expected learning algorithm. In order to acquire a dispersion of outcomes in which one may determine the best course of action, the Monte Carlo Method employs several iterations. The outcomes will be comparable to a player who gradually experiences victories and loses while also picking up lessons from both. This study will examine the scenarios themselves after describing the technique and goal. A beginner (Simple Strategy), a skilled (Basic Strategy), a precise card counter (Complete point count system), and Approaching Twenty One with two additional rule variants have all been modeled in the trials. The idea behind the Monte Carlo Method is that by leveraging experience to enhance the system, the player would be able to "get larger average earnings and approximately estimate the size of the state-action space.

The Monte Carlo method adds more uncertainty than the Combinatorial Framework, which computes mathematical likelihood. It computes probability rather than anticipated numbers because it considers the fact that no two hands are identical due to the unpredictability and the casino's hand. The choice that has the greatest likelihood of success is "probably," according to several trials, the right one. The approach used in this study deviates from the traditional mathematical model. The person's choices are based on a probability matrix built from prior actions rather than creating a random option and storing the outcomes. The emulator will arrive at Basic Strategy faster and more precisely if this matrix is changed in real-time rather than by evaluating stochastic outcomes. This update of the probability matrix is what makes the reverse technique seem the same. In parallel, the algorithms will conduct several models in order to arrive at an ideal decision matrix, much like the Optimization Method.

II RELATED WORKS

Originally with in mid-1960s, Bernard Widrow developed the concept of specific bootstrap acclimation, one of the earliest reinforcement learning algorithms, playing the Twenty one. He gave an example of how a reviewer helped a morphologically component learn how to play without understanding the rules or the goal of the game. Most lately, experiential learning while playing was done using neural networks and temporal differences learning. Eventually, Twenty one value-functions were studied using Monte Carlo techniques [2]. Casinos use numerous packs of cards rather than an one 52-card pack while playing to offset the edge that gamblers gain from employing card counting techniques. Gamers instinctively see that their lead drastically declines. The deficit is quantified using an assessment based on the core limit theorem and a Monte Carlo simulation while using several sets and as the casino moves through the pack. [3] In these results show that a step-wise betting approach can raise the game's anticipated wins by a factor of up to 2.6.

III Experiments and Evaluation

Considering a series of events produced by tracking and crossing through ‘s’, we want to evaluate v(s), the value of a state given policy. A visitation to ‘s’ is used to describe each recurrence of state ‘s’ during an event. Although ‘s’ could appear more than once in a single event, let's refer to the initial appearance as the first visit to ‘s’. While the every-visit MC approach averages the yields after all visits to ‘s’, the first-visit MC method assesses v(s) as the average of the yields following initial visits to ‘s’. Although the theoretical characteristics of these two Monte Carlo (MC) approaches are quite similar, they differ somewhat. If the gambler prevails, the Q price payout will be increased for all the States (all the gambler turns when he receives his hand value and the casino's visible hand value) and all the coupled chosen Moves (take or untake). One may achieve an ideal Q function that becomes finer as the number of played samples rises by repeatedly performing this sampling. A core code and two auxiliary programs set up particularly for rule variations and card counting systems make up the model environment. Before the model performs, all parameters and ranges are generated, and certain elements are then reinitialized after each card is dealt. The principal role of the original block is that of a dealer. Here, it verifies potential outcomes so that control may be sent to the appropriate function. The subroutines Monte Carlo and Forward are called to deal cards to the gambler and casino, accordingly, in the opening hand. With respect to how many times an action-state pair has been examined, the learning rate decays. It scales how much change we want to make to Q value function. We reduce our forecast from reality (seen) to the Q value. This phrase is sometimes referred to as the incorrect phrase. It proceeds to the command line to examine the casino's and gambler's hands for Twenty one after dealing each of them two cards. The metrics gathering subroutine is invoked if one or both have it.

Input: policy π, $num\_episodes$
Initialize $N(s,a) = 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
Initialize $returns\_sum(s,a) = 0$ for all $s \in \mathcal{S}, a \in \mathcal{A}(s)$
for $i \leftarrow 1$ ***to*** $num\_episodes$ **do**
  Generate an episode $S_0, A_0, R_1, \ldots, S_T$ using π
  for $t \leftarrow 0$ ***to*** $T-1$ **do**
    if (S_t, A_t) *is a first visit (with return* G_t*)* **then**
      $N(S_t, A_t) \leftarrow N(S_t, A_t) + 1$
      $returns\_sum(S_t, A_t) \leftarrow returns\_sum(S_t, A_t) + G$
  end
end
$Q(s,a) \leftarrow returns\_sum(s,a)/N(s,a)$ for all $s \in \mathcal{S}$, $a \in \mathcal{A}(s)$
return Q

Fig 1. Pseudocode of the number of events the Agent gathers is indicated in variable "num_events"

Apart for players own full cards and the casino's up card, the gambler entirely disregards every card he has ever seen. Given that basic strategy is the best technique to play with this knowledge. Important actions made by the gambler(such as splitting a pair, doubling down, standing, or drawing) should be made in the appropriate sequence only after aqcuiring some experience. Here complete point count system is implemented using high low method. The first phase is to precisely estimate the quantity of unused cards rather than just guessing. As a result, we must keep in mind two figures: the overall point, and the complete cards that are hidden. The total number of unseen cards may be calculated easily. Start counting at 52 for one deck. Minus it from the cumulative estimate each time you see one of the cards in game. Not altering the count of unknown cards if a card is played. Winning round equals one point, losing round equals -ve one point and draw round eqauls zero point. Those are the implemented criteria for reinforcement learning. Two rule variations created are stated as below. First one is, gambler has the option to double his stake and receive 1 card if a pair of card are numbered identical cards. Secondly the gambler can at the beginning split any 2 cards regardless of their number, symbol or color.

IV Results and Discussion

Rule variations from the standard game brought were affected by the policy changes observed and it's basically reversed the situation and state of the event because instead of splitting the pairs one could make the decision to double down and vise versa. Neverthless we can observe it’s not effecting the gambler's decision dramatically since all other conditions are met similarly.

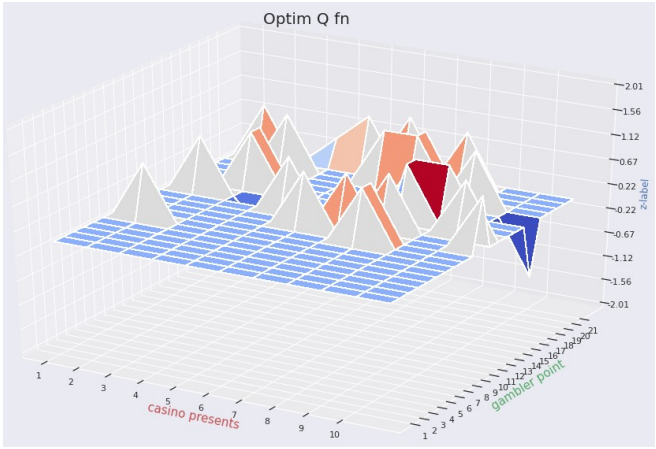

Fig 2. Visualization of Q value optimization function *when Q = 100 plays.*

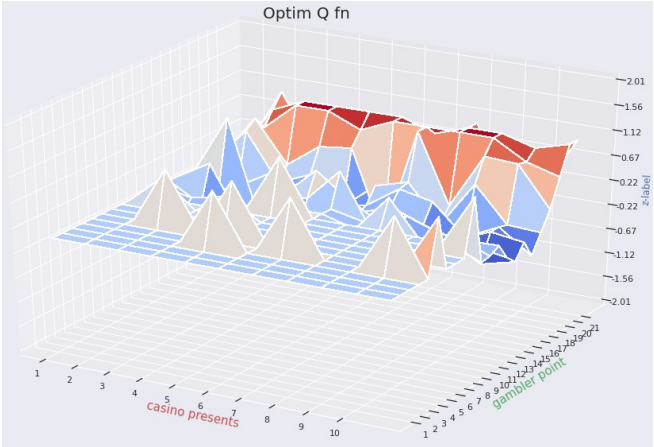

Fig 3. *Visualization of Q value optimization function when Q = 1000 plays.*

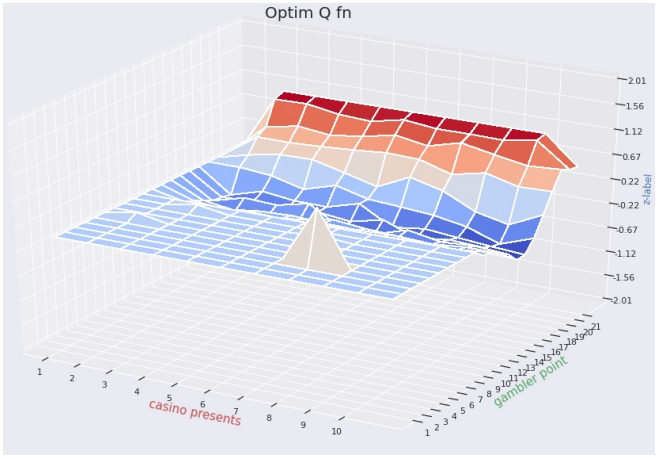

Fig 4. ***Visualization*** **of Q value optimization function** ***when Q = 100 00 plays***

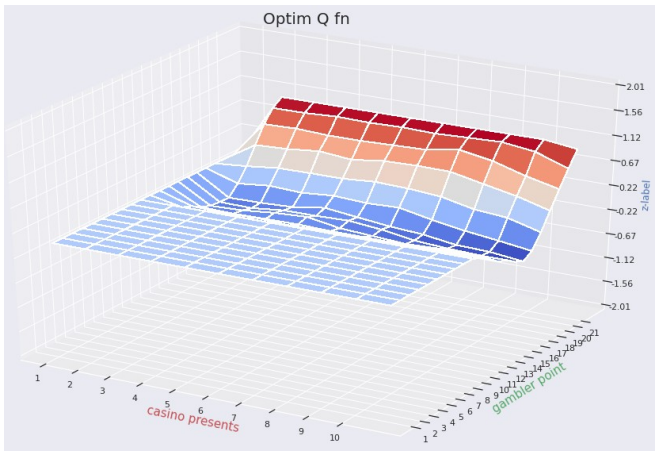

Fig 7. ***Visualization*** **of Q value optimization function** ***when Q = 100 000 plays***

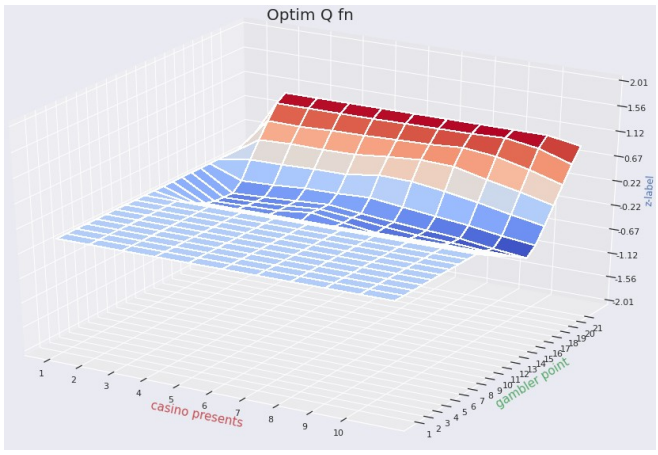

Fig 8. ***Visualization*** **of Q value optimization function** ***when Q = 1000 000 plays***

One could see from the above figures that the Q value variation diminishes and becomes more "even" as the amount of games played increases. The variation is so significant at 100 games that we can see peaks on the chart.

casino_presents	1	2	3	4	5	6	7	8	9	10	11
gambler_point											
1	take	take	take	take	take	take	take	take	take	take	take
2	take	take	take	take	take	take	take	take	take	take	take
3	take	take	take	take	take	take	take	take	take	take	take
4	take	take	take	take	take	take	take	take	take	take	take
5	take	take	take	take	take	take	take	take	take	take	take
6	take	take	take	untake	take	take	take	take	take	take	untake
7	take	take	take	take	take	take	take	take	take	take	take
8	take	take	take	take	take	take	take	take	untake	take	take
9	take	untake	take	take	take	take	take	take	take	untake	take
10	take	take	take	take	take	take	take	take	take	take	take
11	take	untake	take	take	take	take	take	take	take	take	take
12	take	take	take	take	take	take	take	take	take	take	take
13	take	take	take	take	untake	take	untake	untake	take	take	take
14	take	untake	take	untake	take	untake	untake	take	take	take	untake
15	take	take	untake	take	take	take	take	untake	take	take	take
16	take	take	take	take	untake	take	untake	untake	take	take	take
17	take	take	take	untake	take	take	take	untake	untake	untake	take
18	take	take	take	take	take	take	untake	take	untake	take	take
19	take	take	take	take	untake	untake	untake	take	take	untake	untake
20	take	untake	untake	take	take	take	untake	take	untake	take	untake

Fig 9. ***Perect move chart discovered from a 100 games***

casino_presents	1	2	3	4	5	6	7	8	9	10	11
gambler_point											
1	take	take	take	take	take	take	take	take	take	take	take
2	take	take	take	take	take	take	take	take	take	take	take
3	take	take	take	take	take	take	take	take	take	take	take
4	take	take	take	take	take	untake	take	take	take	take	take
5	take	take	take	take	take	take	take	take	take	take	take
6	take	take	take	take	take	take	take	take	take	take	take
7	take	take	take	take	take	take	take	take	take	take	take
8	take	take	take	take	take	take	take	take	take	take	take
9	take	take	take	take	take	take	take	take	take	take	take
10	take	take	take	take	take	take	take	take	take	take	take
11	take	take	take	take	take	take	take	take	take	take	take
12	take	take	take	take	take	take	take	take	take	take	take
13	take	take	take	take	untake	take	take	take	take	take	take
14	take	untake	untake	take	untake	untake	take	untake	take	take	take
15	take	untake	untake	untake	untake	untake	untake	take	take	untake	take
16	take	untake	untake	untake	untake	untake	untake	untake	untake	untake	take
17	take	untake	untake	untake	untake	untake	untake	untake	untake	untake	untake
18	take	untake	untake	untake	untake	untake	untake	untake	untake	untake	untake
19	take	untake	untake	untake	untake	untake	untake	untake	untake	untake	untake
20	take	untake	untake	untake	untake	untake	untake	untake	untake	untake	untake

Fig 10. ***Perfect move chart discovered from a million games***

Fig.9 and Fig.10 shows that smart move that gambler should take is convergence of Q value function into optimized moves and one could follow this chart table of takes and untakes inorder to achieve above average performance of gaining profit.

Different learning algorithms have specific advantages and disadvantages over the average performance and profit gainings. It is because of their different approach for the implementation part. Q-learning is a model-free algorithm [4]. Since there are no labels, a reward system is employed instead by establishing the Game Rules and specifying what constitutes a prize and what does not. When our model is first being trained, it will begin to collect randomised moves (the randomization technique may also be chosen) from various states and examining the various

results. Based on real point value, a learning rate, the highest predicted reward available, and the actual reward discovered, a score is established and modified for each (Position, Move) pair. Temporal Difference (TD) is a modification of the Q-Learning process in which the periodicity of updating the Q value depends on a factor known as the qualification traces. The trace factor, which ranges from 0 to 1, indicates how quickly traces degrade (closest to 0 means a faster fading). Because a previous estimate is used to upgrade the Q value, this strategy is known as the "bootstrapping technique." The upgrade phase became the complete event, returning us to the precise Monte-Carlo condition when we take an Eligibility Traces number of 1. We may infer from a few studies that a smaller Trace Factor takes longer to learn (to balance), but produces reduced error comparing to Monte-Carlo.

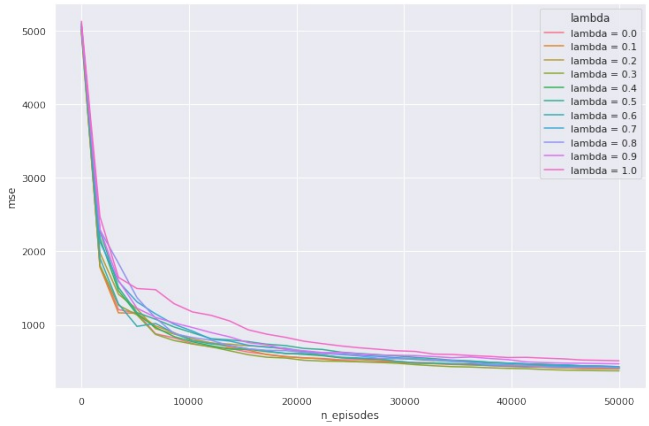

Fig 10. ***Test case for Plotting and Evaluating Alternative Lamda values for 50 000 Plays***

V CONCLUSIONS

By the conclusion of this paper, it will be clear that using the Monte Carlo Method does offer a close imitation of a gambler gradually learning how to play Twenty one. The mechanism used in this model, which forces players to decide between taking and untaking, allows it to come close to Basic Strategy. There are several limitations to this paper's improvements that may have enhanced the experiments and hence raised the profit %, even though the models were able to precisely meet the paper's objectives. As previously stated, treating hard and soft totals independently would significantly enhance the experiment. It would provide greater model performance and probably lead to a larger profit %. The problem with this is that it would not only require more trials to attain a stable condition, but it would also double the duration of each run. These trials were selected to have an acceptable balance with the trade-off among precision and performance. A thorough exploration of different learning algorithms could make in the future. So that combination of hybrid learning algorithms like integrating a Deep learning model to this may increase the chance of gaining more advantage in beating the casino. Also experimenting in the perspective of increasing the chance of winning for Casino also interesting topic to study later eventhough all Casino sets their rules for their advantage.

REFERENCES

1 Edward O. Thorp. Beat the Dealer . Vintage, New York, 1966.

2 Vaidyanathan, Atul. "Monte Carlo Comparison of Strategies for Blackjack." (2014).

3 Golden, Leslie M. "An analysis of the disadvantage to players of multiple decks in the game of Twenty-one." Mathematical Scientist 36.1 (2011).

4 Richard S. Sutton and Andrew G. Barto. Reinforcement Learning: An Introduction . MIT Press, Cambridge, MA, 2 edition, 2018.