Reinforcement Learning for Blackjack

Saqib A. Kakvi[1]

Goldsmiths, University of London, SE14 6NW, London

Abstract. This paper explores the development of an Artificial Intelligence system for an already existing framework of card games, called SKCards, and the experimental results obtained from this. The current Artificial intelligence in the SKCards Blackjack is highly flawed. Reinforcement Learning was chosen as the method to be employed. Reinforcement Learning attempts to teach a computer certain actions, given certain states, based on past experience and numerical rewards gained. The agent either assigns values to states, or actions in states. This will initially be developed for Blackjack, with possible extensions to other games. Blackjack is one of the simpler games and the only current game in the SKCards package which needs an Artificial Intelligence agent. All the other games are single player. To test the performance of the Reinforcement Learning agent, several experiments were devised and run.

1 Background

1.1 Current System

The current system is a framework for all card games. There is a reresentation of Cards and gameplay is represented by the interdependant Interfaces Player and UI. The Blackjack Player has an attempted AI implementation, which is a number of nested if-else if-else clauses. This does not work.

1.2 Blackjack

[1] The object is to get upto 21 points in a maximum of 5 cards. This is achieved is by assigning values to each of the cards in the deck: 2-10=Face value; Courts=10; Ace=1/11. The player may then take another card(hit) or end thier turn(stand).

1.3 Reinforcement Learning

Reinforcement Learning learns from rewards for taking a sequence of actions in an evirnoment, based on its knowledge. This will lead to a change and eventually a reward. The agent will accordingly adjust its knowledge. Agents can act based upon values of states or actions in states. It learns based on past rewards for an action, A, and the reward received for action A. For simplicity, we take the average of the previous rewards and update it and also allows the agent to know how good a reward is.

The agent implemented is a softmax selection agent. It choses actions based on thier probability, from the equation below:

$$\frac{e^{Q_t(a)\backslash\tau}}{\sum_{b=1}^{n} e^{Q_t(b)\backslash\tau}} \quad (1)$$

[2]

This strikes a balance between exploitation and exploration. The parameter $0 < \tau < 1$ is called the temperature and decides how much the agent explores.

The basic value method uses discounted. The agent passes back the reward multiplied by a constant $0 < \gamma < 1$, known as the discount rate. Multiplying the reward by the step-size parameter, $0 < \alpha < 1$, mitigates the effect of noise in the data. By calculating these values, the agent learns a policy, P, which is the action the agent will take in a given state. We aim to learn a policy $P \simeq P^*$.

2 Experiments & Results

4 basic tests were devised. Each test uses different values of rewards to achieve a different purpose. The first test uses the normal values: Win = +10; Push = +5; Lose = -10; Bust = -10. The subsequent tests involve doubling one reward value and keeping the rest at the norm. The dealer hits until a set value. The constants used were: $\tau = 0.5$; $\alpha = 0.6$; $\gamma = 0.75$; A sample of the results are given in Table 1.

2.1 Conclusion

The results show that by altering the rewards, alters the policy of the agent. It is seen that altering negative rewards has a larger effect than altering positive rewards. From the net wins, we can see that policy alone cannot win.

References

1. D. Parlett, *Teach Yourself Card Games* (Hodder Headline Plc, Euston Road, London, NW1 3BH, 1994).
2. R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction* (MIT Press, Cambridge, MA, USA, 1998).

DEALER POLICY	WIN REWARD	LOSE REWARD	BUST REWARD	FINAL POLICY	WIN (%)	LOSE (%)	BUST (%)	NET WINS(%)
11	10	-10	-10	17	44.62	17.6	30.66	-3.64
17	10	-10	-10	18	19.08	17.4	38.74	-12.86
11	20	-10	-10	16	45.98	23.1	22.02	0.86
12	20	-10	-10	17	42.76	19.54	29.82	-6.6
15	10	-10	-20	12	19.48	58.88	0	-26.26
16	10	-10	-20	12	18.3	58.26	0	-25.72
16	10	-20	-10	20	20.46	5.8	59.86	-31.12
17	10	-20	-10	20	16.14	6.04	59.02	-25.7

Table 1. Final Results