Learning to Play Texas Hold'em Using Reinforcement Learning

UNIVERSITY *of* LIMERICK

OLLSCOIL LUIMNIGH

Department of CSIS

Bachelor of Science in Computer Systems

Author: Jamie Mac Manus

ID: 15147312

Supervisor: J.J Collins

Abstract

In recent years the area of machine learning has gained a lot of ground in a broad range of areas. A particularly interesting problem pertaining to machine learning is how we can develop useful AIs in a more hands off fashion. This problem is addressed by a machine learning paradigm named reinforcement learning. Reinforcement learning allows us to set up an agent in an environment after which the agent can explore the environment and begin to learn the more which actions that it should take in the different scenarios it can find itself in. This avenue of machine learning is suited to a broad range of problems but one interesting area is that of imperfect information games such as texas holdem.

Contents

Chapter 1

Introduction (10)

In this section I will introduce the subject area of this Final year Project (FYP). I will then go on to give an overview of the report, establish some goals for the project along with some of the motivations for choosing this subject area.

1.1 Overview (3) w5

Since the inception of machine learning, games have been a key problem area that has seen a lot of focus from top academics. Furthermore, the development of some machine learning strategies that can be applied to games has also lead to these strategies being applied in many different domains, many of which being very beneficial in practice.

In

1.2 Objectives (2) w4

1.2.1 Primary Objectives

Leverage Deep Reinforcement Learning to Develop a Poker Playing Agent

In the past, methods such as counterfactual regret minimization (CFR) have been used to develop agents that can play no-limit texas hold'em to a superhuman level. There have also been attempts to solve the limit version

of the game using reinforcement learning (RL). Throughout the course of this project I will be using an iterative approach to solving the problem. As such the first agent that I will develop will seek to tackle a simplified version of hold'em. Specifically I will be attempting to recreate the results outlined in[1]. This will be my initial objective with potential future iterations expanding upon this work to tackle a more complete version of the game.

Experiment with Different Reinforcement Learning Methodologies

Although all RL algorithms share certain core properties, there are a number of distinct approaches that we can take. In this project we will attempt to determine, through statistical analysis, which approach yields the most favourable results.

Develop a Web Interface For Users to Play Against the Agent

The focus of this report will largely be research. However it is also my goal to create a product that will be fun and useful for the general public. As such another objective will be to create a website that will allow users to play heads-up against the final product.

Utilise Various Techniques to Prove the Efficacy of the AI Agent Produced

In order to prove the poker playing agent's ability I hope to leverage a number of techniques. As the first version of the poker playing agent will be based on[1] I will be using similar techniques as outlined in the paper to test the agent. This means that I will test the agent against a random player, a balanced player and an aggressive player. The random player will play in a completely random manner, the balanced player will play according to the minimax algorithm and finally the aggressive player will be a variation of the balanced player that folds less regularly.

In my research thus far a prominent method of testing poker agents is to utilise previously established agents as a benchmark. This involves playing a number of games against these agents and recording the results in terms of winnings. An example of this is[2] that utilised some of the best current poker agents as established by the Annual Computer Poker Competition (ACPC).

1.2.2 Secondary Objectives

Understanding Reinforcement Learning

As this project is very specific and academic, one of the larger challenges will be to gain a strong knowledge of the domain. This means learning the history of RL, the types of problems that it has been used to solve and the specific details of different RL algorithms.

Understand the Existing Literature on Artificial Intelligence and Imperfect Information Games

A successful project will require a high degree of knowledge from the broader domain of RL. However, it is also the case that I must become closely familiar with the existing academic literature in the area of RL with respect to imperfect information games. This will allow me to avoid taking approaches that have previously shown to fail and also allow me to contribute to the existing literature without simply replicating what has already been done.

Learn about Different Approaches to Implementing Poker Agents

1.3 Contribution (1) w5

1.4 Project Plan (1) w5

1.5 Motivation (2) w5

Chapter 2

Background (18)

The aim of this chapter is to give the reader background information on the problem domain in order for them to understand the rest of the report. This will consist first of an introduction to machine learning. Then we will go into more detail on the areas of reinforcement learning and texas hold'em. Finally we will take a deeper dive on the literature surrounding how we can utilise reinforcement learning to tackle the problem of texas hold'em.

2.1 Introduction to Texas Hold'em

Texas hold'em is one variant of the family of games called poker. Poker is a group of card games that combine gambling, strategy and skill. All poker variants have three core similarities. There is betting involved, there is imperfect information (ie cards remain hidden until the end of a hand) and the winner is determined by combinations of cards. We will now discuss texas hold'em poker in more detail.

2.1.1 Game Structure

Texas hold'em consists of four betting rounds. Initially each player is dealt two private cards. These remain face down and only the person who received these cards may view them. In the next three rounds five public cards are dealt face up on the table. The second round of dealing is called the flop, where three cards are dealt. The third round is called the turn where one additional public card is dealt. Finally in the fourth round another card is

dealt which is called the river.

At each round, after the cards are dealt, the players are given the opportunity to take a number of betting related actions. We will discuss the permitted actions in the next section.

In order for players to be incentivized to continue playing in a wider array of situations, blinds are required. Blinds are a mandatory bet that must be posted by two of the players present at the game. These two bets are called the big blind and the small blind, the big blind being twice that of the small blind. As hands are played the big and small blinds are posted by different players in order to distribute the cost fairly.

The big and small blind are the first two bets that contribute to what's called the pot. The pot is the collection of all of the current chips bet by the players. When a player wins a hand then what they receive in return is the pot.

The final structural component of the game is player stacks. Each player will start the game with a certain amount of chips. If a player wins a pot then all of the chips in the pot are transferred to the winners stack.

2.1.2 Actions

2.1.3 Hand Values

2.2 Introduction to Machine Learning (1) w6

Machine learning is an area of computer science that tackles how we construct computer programs that improve with experience[3]. The term was coined by Arthur Samuel in his 1959 paper where he discussed machine learning methods using checkers as his problem area. Since then there has been a great deal of advancement in the field. Some of the notable early contributions being the discovery of recurrent neural networks in 1982 and the advancement of reinforcement learning by the introduction of Q-Learning in 1989. Recently we have seen some of this early academic work culminate in more practical achievements such as Facebook's DeepFace system which, in 2014, was shown to be able to recognise faces at a rate of 97.35% accuracy, a rate that is comparable to that of humans. Another example of recent achievement is Google's AlphaGo program which, in 2016, became the first program to beat a professional human player.

It should be becoming clear that machine learning can be a solution to a wide scope of problems and as both hardware and software continue to improve this scope will only continue to widen. We are starting to see machine learning systems become a key component of many companies business model. Since certain machine learning techniques are great at prediction, machine learning has been widely used for content discovery by companies such as Google and Pinterest. Other business applications include the use of chatbots as a part of customer service, self-driving cars and even in the field of medical diagnostics.

Since there is such a large range of actual and potential applications for machine learning it would be good for us to understand how different methodologies can be applied to solve different types of problems. In the next we will discuss just that.

2.3 Machine Learning Categories (3) w6

Supervised Learning

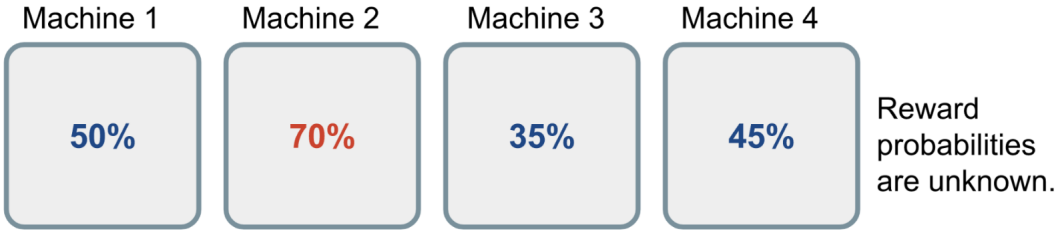
Supervised learning involves an agent which observes some example input-output pairs and learns a function that maps from input to output.[4]. This learned function can then be used on new input data, that wasn't used to train the agent and the agent should be able to give an accurate output. As such this learning task is a generalization problem. The agent must be able to identify general features of the input data and how they map to the output. Common examples of

Unsupervised Learning

Reinforcement Learning

The third category of machine learning is reinforcement learning. We will now discuss this area of machine learning in depth.

2.4 Explore-Exploit Dilemma



When it comes to reinforcement learning one of the first questions that we have to ask is how we explore the state space. An example that is often used to conceptualize this problem is the multi armed bandit problem. Let's say an agent is in a room with a number of gambling machines. Each of these machines has an arm that, when pulled will return a reward of 0 or 1 based on some underlying probability[5]. The agent has a limited number of total pulls. So the question becomes how do we distributed these pulls in order to maximise return? Well, first we have to ensure that we explore enough that we find the machine with the best reward probability and second, we must then exploit this machine to the best of our abilities.

There are a number of approaches that can be taken to solve this problem, we will now briefly discuss two of these methods.

2.4.1 ϵ-Greedy Solutions

The first approach that we will discuss is the ϵ-greedy strategy. This approach was first proposed by Watkins[6] and is a very simple and widely used method. The ϵ-greedy strategy involves choosing a random lever some proportion ϵ of the time, and choosing the lever that has been established to give the highest reward the rest of the time.

There are a number of variations of this method, the first being the ϵ-first strategy. With this strategy we take all of our random choices first, allowing us to establish the best bandit, after which we exploit this bandit. However, as stated in[7] this simple approach is sub-optimal because asymptotically, the constant factor prevents the strategy from getting arbitrarily close to the optimal lever.

This is where the ϵ-decreasing strategy becomes useful. Here, the proportion of random lever pulls decreases with time. Generally if our initial

epsilon value is ϵ_0 then our epsilon value at time t will be $\epsilon_t = \frac{\epsilon_0}{t}$.

2.4.2 Interval Estimation Strategy

Another approach that can be used is called the interval estimation strategy. With this method we initially give an optimistic estimate of the reward to each bandit within a certain confidence interval. Then we simply take a greedy approach to our exploration. Less explored bandits will have a artificially higher reward estimate and thus they will be greedily chosen, thus allowing us to evaluate each of the bandits.

In the context of reinforcement learning, state space exploration through the ϵ-greedy approach is generally sufficient.

2.5 Markov Decision Processes w7

Reinforcement learning problems are generally modelled according to what is called a markov decision process.

- S - a set of states.
- A - a set of actions.
- P - a set of state transition probabilities
- R - a set of rewards
- γ - a discount factor

2.6 Reinforcement Learning Problems w7

In reinforcement learning there are two

2.7 Reinforcement Learning Methods (9) w7

There are three primary categories of reinforcement learning algorithms. These are dynamic programming, monte carlo and temporal difference learning.

2.7.1 Dynamic Programming (3)

Dynamic programming (DP) refers to a collection of algorithms that can be used to compute optimal policies given a perfect model of the environment as a Markov decision process[8]. Dynamic programming provides a basis for many algorithms that are used in practical reinforcement learning applications. An example of such an algorithm is value iteration. This is where we alternate between policy evaluation and policy improvement in order to converge towards an optimal policy.

2.7.2 Monte Carlo (3)

In Monte Carlo, unlike dynamic programming, we do not assume complete knowledge of the environment. Monte Carlo methods require only experience-sample sequences of states, actions, and rewards from interaction with an environment[8]. Monte carlo evaluation is an episodic process this means that we only update our action values after an episode has completed.

2.7.3 Temporal Difference Learning (3)

2.8 Reinforcement Learning In Large State Spaces (1) w7

Chapter 3

Application Development (10)

3.1 Requirements

3.2 Design

3.3 Backend API

3.4 Frontend Website

3.5 Testing

3.6 Issues

Chapter 4

Empirical Studies (21)

4.1 Experiment 1 (3)

4.2 Experiment 2 (3)

4.3 Experiment 3 (3)

4.4 Experiment 4 (3)

4.5 Experiment 5 (3)

4.6 Experiment 6 (3)

4.7 Experiment 7 (3)

Chapter 5

Conclusions (6)

5.1 Summary (2) w25

5.2 Reflections (2) w25

5.3 Future Work (2) w25

Bibliography

[1] F. A. Dahl, "A reinforcement learning algorithm applied to simplified two-player texas holdem poker," in *European Conference on Machine Learning*, pp. 85–96, Springer, 2001.

[2] J. Heinrich and D. Silver, "Deep reinforcement learning from self-play in imperfect-information games," *arXiv preprint arXiv:1603.01121*, 2016.

[3] T. M. Mitchell *et al.*, "Machine learning. 1997," *Burr Ridge, IL: McGraw Hill*, vol. 45, no. 37, pp. 870–877, 1997.

[4] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[5] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[6] C. J. C. H. Watkins, *Learning from delayed rewards*. PhD thesis, King's College, Cambridge, 1989.

[7] J. Vermorel and M. Mohri, "Multi-armed bandit algorithms and empirical evaluation," in *European conference on machine learning*, pp. 437–448, Springer, 2005.

[8] R. S. Sutton, A. G. Barto, F. Bach, *et al.*, *Reinforcement learning: An introduction*. MIT press, 1998.