CSCI 5622: Final Project Proposal
Construction and Training of a Poker Bot Using Reinforcement Learning

Team Members
Osman Asif Malik
Mike Ramsey
Kwan Ho Lee
Erik Johnson

Our Project

Our project is to construct and train a poker bot. The goal is for the bot to play as well as possible against human players. To start off, we will consider a simplified version of heads up fixed limit Texas Hold'em. If time permits, we will consider more advanced versions of the game.

Why This Project

Our project falls within the long history of research into training computers to play games like chess, Go, and poker. Poker is an especially interesting game to investigate because, unlike chess and Go, there is randomness (e.g., the cards one is dealt), hidden information (e.g., the competitor's hand), and the goal of not just winning but maximizing one's winnings. Although these factors increase poker's complexity and the size of the underlying state space, there are many different types of poker and simplifying assumptions one can make to reduce the complexity and the size of the state space. Thus, training a poker bot is accessible if we choose a simple type of poker but can also be made arbitrarily challenging by choosing a more difficult type of poker.

Moreover, poker is a game that people find both fun and intellectually interesting. Thus, this project offers an interesting problem which many people are interested in and on which we can test and further develop our machine learning chops. There is also a potential for monetary reward. There are ongoing poker bot competitions. Moreover, casinos now use machine learning software in their poker games.

To train our poker bot, we will use reinforcement learning which is a topic we won't cover much in CSCI 5622. Thus, this project also offers an opportunity to add more tools to our machine learning toolkit. In summary, like the many teams in academia and industry who for decades have been studying how to best train a poker bot, we think this project will be interesting, accessible, and challenging (and potentially lucrative).

Why Machine Learning Techniques are Appropriate

The main reason why machine learning is appropriate to train the poker bot is due to complexity and randomness. For example, AlphaGo—a computer program that plays the board game Go—is a good example of how machine learning techniques are appropriate for the application to poker. Unlike chess in which there are a relatively small number of moves, in the game Go there are a vast number of moves. Similarly, poker has a very large number of states as well as incomplete information. Reinforcement learning is an ideal tool for exploring and learning in such a setting.

Our Approach

We plan to use reinforcement learning to train the poker bot. Reinforcement learning is different from supervised and unsupervised learning in that the agent makes multiple subsequent decisions which each may have an associated cost or reward. The agent is concerned with how to take *actions* in different *states* in order to maximize a *reward*. In our case, the agent is the poker bot. The actions are to check, bet, call, or fold. The state is the specific cards that the bot and player have together with the results of the previous actions by the bot and its opponent. The reward is the total amount won or lost in one hand. As you can see, the reinforcement learning formulation of training a poker bot poses the problem in a sensical manner.

But how will the poker bot know what is the best action to take? This question addresses one of the characteristic features of reinforcement learning: the *exploitation* and *exploration* trade-off. For a given hand, the poker bot must exploit its learned knowledge in order to win the hand. However, it also must explore in order to learn the best possible actions to make. The poker bot would be unsuccessful if it was unable to explore new moves. For this reason, we must decide on how to address the trade-off described above. We plan to implement epsilon-greedy methods and upper-confidence-bound action selection. We will additionally have to perform parameter studies to find the optimal parameters.

Notice that the poker bot does not have complete knowledge of his environment, i.e., the bot does not know the cards that its opponent has. For this reason, the bot will not have all of the information that it needs to make the optimal action. To address this, we plan to train the bot using a Monte Carlo approach. The bot will weigh his decisions based on the expected return of taking a certain action in a given state. For example, if the state (King, King) followed by a bet action generally has a high reward over the course of learning, then the bot will have a tendency to bet if that state appears in the future. Therefore the poker bot will only make decisions based on learned experience, and not on any prior, pre-learned information.

Dataset(s)

Since we will be using reinforcement learning to train our poker bot, we will not need any datasets. To train the bot, we will train it against instances of itself. Self-play is an established technique in reinforcement learning, and has been used to teach machines to play poker. We may also use some of the existing bots others have trained and put on the internet to train our bot.