Deal me in: a neural network approach to blackjack

Matthew Traino
Department of Computer Science
Florida Southern College
Trainomatt1@gmail.com

***Abstract* — Training a neural network on the game of blackjack and have the network decide what to do from the state of the game. Currently, the network can only make assumptions on if the correct move is to hit or stand.**

I. Introduction

Blackjack is a “simple” card game. With the overall objective to get as close to 21 without going over and beating the dealer. Each card is worth its face value. Aces are worth either one or eleven and king, queen, jack are all worth 10. The goal of this project is to train an algorithm to play out games of blackjack with a decent win to loss ratio.

II. Approach

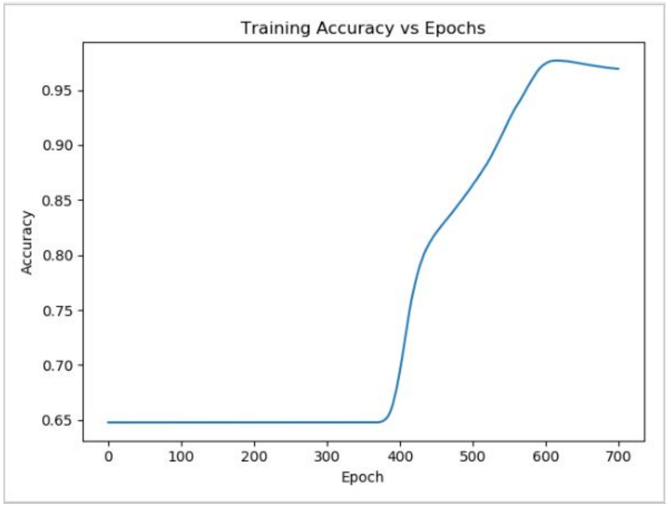
Two thoughts came to mind for completing this task. The first, but not used in this program, is a decision tree. A decision tree probably would have been easy to implement, but I did not want to give the algorithm a method of playing the game. I wanted to see what could be learned with little human interaction. So for my approach, I implemented a neural network using a 104-100-100-256-1 layout. The network uses the sigmoid function on all the hidden layers and the relu function on the output layer.

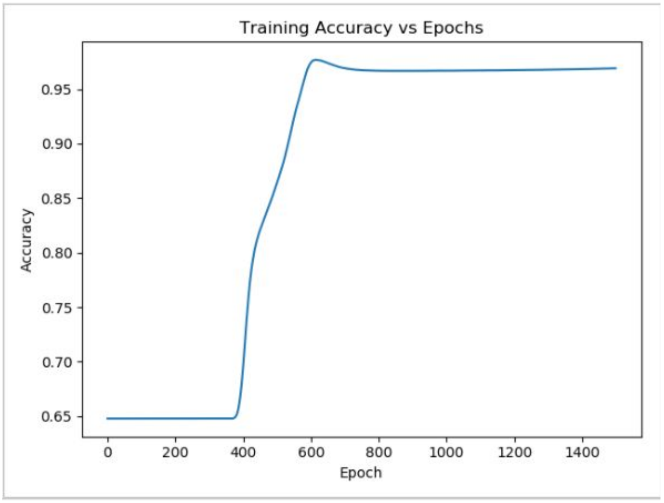
III. Challenges

The first challenge faced was deciding the best way to store and represent the data. The solution to this problem was to create an array of size 104. The first 52 portions of the array are used to represent the cards the network has, the last 52 portions are used to represent what card of the dealer's hand the network can see to base all decisions from. The next issue is creating a sufficient amount of data for the network to learn from. Creating this amount of data by hand was not an option because even with 100,000 plus training samples the network still needed to run over 500 epochs in order for the accuracy to improve and the loss to decrease. The solution for this was having a script that ran through games of blackjack and writing the array of size 104 to a file and creating a label to go with each sample.

IV. Results

An untrained neural network when playing 100,000 games of blackjack on average had a 0.00% win rate. This was due to the network not understanding the basic rules of the game and would keep trying to hit no matter the value of the hand. Upon training the network with 500,000 training samples on 700 epochs with a learning rate of 0.001, had on average a 37.82% win rate when playing vs the dealer who hits on anything that is less than a 17 until the dealer’s hand is equal to or greater than 17. Allowing the network to train for 1500 epochs with the same parameters only marginally improved the win rate to 40.90% on average.





V. Conclusion

The trained network compared to the untrained network was massively better in winning at blackjack. Overall, the choice of a neural network or decision tree for this case is still a tough one to make. Without testing a decision tree for this dataset I am curious as to how it would perform compared to the trained neural network. Another algorithm that strikes my eye is one that could also be used is a genetic algorithm. Having the algorithm teach itself how to play the game from just playing out of the game over and over and using a survival of the fittest algorithm to decide which children pass their genes onto the next generation. This is something I hope to revisit in the future and compare all of these results.