CSCI 4156 – Reinforcement learning

Assignment 2

Temporal Difference Agent for Backgammon game

Noof Al Shehhi

B00832852

**Abstract**

In this Assignment, our goal is to approach TD- gammon using custom built MLP as our agent and trained MLP using neural backpropagation to train a backgammon game. First aproach was to train Agent with MLP using loss calculated with TD algorithm.

**Approach**

I would say approach started mainly with understanding the theoretical and math part of TD and the equation to implement it in the MLP model. My MLP parameters were 250\*16 and 1\*250 and 2 parameters for bias, so in total 4252 parameters.

1-I will go through how I went through parameterizing my MLP agent:

For MLP, I used input size of 16 as observation and for output I used value of 1 as return for preprocessing of data. Then normalizing data step was based on subtracting mean and dividing standard deviation with it. Furthermore, I initialized weights to have mean of 0 and their standard deviation is almost 0.5.

For forward pass propagation I used weight matrices and matrix multiplication from NumPy library and kept the activation function to be sigmoid function that has a gradient of (sigmoid(x)(1-sigmoid(x))). The reward function even after discount ranges from 0 to 1. Therefore, I could say sigmoid was the best function to consider in this case for the last layer and for backward propagation. Moreover, instead of using optimizer, I have used the matrix multiplication for gradients calculation and updated weights by simply multiplying gradients with learning rate of 0.003. I chose my learning rate to be 0.003 after manually checking my accuracy graph if I increased the learning rate. Then I noticed that bigger than 0.003 do not give a good performance. Then I add it to weights for weights using the loss function mean square error loss for weights update it is implemented in train function of MLP class.

**Discussion**:

**Prediction value Learning changes:**

For the prediction value I started my MLP model with random values as weights of MLP then I update weights with the Temporal difference learning (TD) algorithm concept is Markov decision processes of discounted sum of reward on the next state added with a reward at the current state gives us predicted value for error calculation but in TD algorithms we keep old value and new value with some ratio Lets say we have lambda value of 0.7 that means it will save old value 30% and new value 70% which can be seen by algorithm equation

here is an example equation of TD algorithm.

**value\_updated <- value + lambda (0.7) \*(reward + discount (0.95) \*next\_state\_value - value)**

I applied this equation to calculate predicted value and then calculated error which gets less and less over time. so, this model trains more and more and predicts accurate value.

**Findings:**

**How accurate is MLP.**

A screenshot of a computer program

Description automatically generated with low confidence

**Figure1: My Model accuracy while testing.**

According to Figure1, printing the accuracy showed an almost 65% of predicting the win of white in multiple backgammon games. This accuracy refers to good sign that the model is learning and is able to find a pattern as it is higher than 50%. The MLP was quite accurate but as it is built with numpy it is not that accurate to get a good accuracy compared to pytorch model because I think pytorch has advance optimizers that increase MLP performance. other than that accuracy can be improved if we give model visulaization of full boad state to model as input or we use state and action combined to predict value with input shape of 32 which is combined 16 16 values of current board state and action. We can use other ways like we can try other reward functions or if we train multiple agents one for moves of white and one for balck moves then we can even get better results.

Another finding is:

A graph of blue and orange lines

Description automatically generated with low confidence

**Figure2: Visualizing the winning rate of white and black Team.**

In figure2 as the model have been played for the white team, we will see that the selected player(white) team wining rate increases, it went from 50% to 70%. And the black team was decreasing which means losing the game. This finding make sense as the prediction should be fair prediction with 50% for white and 50% chance for black. If we have the white wining prediction, then the black prediction should be located under 0.5 to represent losing prediction. So, one player can win the game. The prediction seemed to be consistent and stable which is good for the model.

**Conclusion:**

Temporal difference improved MLP performance for backgammon game. In addition, it was helpful in doing two tasks at the same such as board evaluation and prediction.