CPEN 502 Assignment-b: Reinforcement Learning (Look Up Table)

Ali Asgari Khoshouyeh (Student #24868739)

December 15, 2021

Team Members

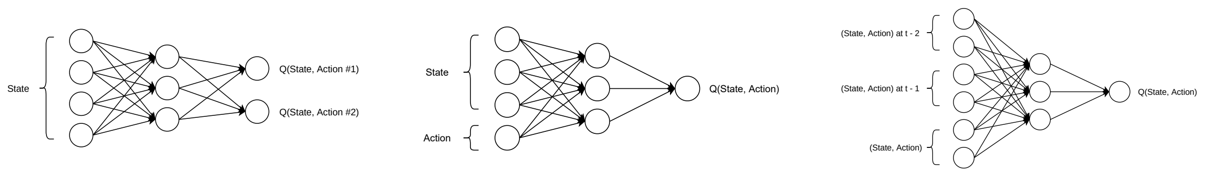
We are a team of three sharing the same code base.

- Christina Sun
- Husna Kalim
- Ali Asgari Khoushouyeh

It is noteworthy to mention that close to the extended deadline we realized that our code is orders of magnitude slower on my teammates' machines. So we sharing the plot data too.

(4) The use of a neural network to replace the look-up table and approximate the Q-function has some disadvantages and advantages.

a) There are 3 options for the architecture of your neural network. Describe and draw all three options and state which you selected and why. (3pts)



(a) Action as output (b) Action as input (c) Adding time dimention

Figure 1: Different architectures considered for the neural network as the function approximator for the Q values of different game actions.

FIg. 1 shows the different architectures we considered for our neural network to be plugged in instead of the lookup table. The first two architectures shown in Fig. 1a and Fig. 1b are the ones discussed in the class. The first one codes the Q value for different possible actions at each step as different neurons of the output. In the second architecture the action is coded as some input neurons and the Q value corresponding the the encoded action as the input at a certain state is coded as the single output neuron.

The novel architecture that we used is the one shown in Fig. 1c which is essentially same as the second architecture, but, it also receives the state-actions of the two past time-steps as input. This is following our intuition used in the second part of the assignment where we included the past two state-actions in the lookup table keys to capture the delay of rewards as it may take some time for a bullet to hit the enemy and get reflected in the reward policy.

0.1 b) Show (as a graph) the results of training your neural network using the contents of the LUT from Part 2. Your answer should describe how you found the hyper-parameters which worked best for you (i.e. momentum, learning rate, number of hidden neurons). Provide graphs to backup your selection process. Compute the RMS error for your best results. (5 pts)

Learning Rate	Momentum	Hidden Neurons	RMSE
1.00E-04	0	5	9.666091389
1.00E-04	0	15	9.660078042
1.00E-04	0	30	9.628909289
0.001	0	5	7.632740399
0.001	0	15	6.943443362
0.001	0	30	6.711401519
0.01	0	5	7.985192317
0.01	0	15	7.018144655
0.01	0	30	6.530952131
1.00E-04	0.5	5	9.148578254
1.00E-04	0.5	15	8.993952348
1.00E-04	0.5	30	9.051175005
0.001	0.5	5	7.297405961
0.001	0.5	15	**6.335407802**
0.001	0.5	30	**6.289893926**
0.01	0.5	5	7.919711744
0.01	0.5	15	7.75168296
0.01	0.5	30	8.222388904
1.00E-04	0.9	5	7.713839065
1.00E-04	0.9	15	6.790881731
1.00E-04	0.9	30	6.754064308
0.001	0.9	5	7.772037228
0.001	0.9	15	7.099569645
0.001	0.9	30	6.728784713
0.01	0.9	5	11.50926595
0.01	0.9	15	13.85358874
0.01	0.9	30	18.89859967

Table 1: RMSE after 100 epochs under different hyperparameter values for training the neural network on the static lookuptable data.

We performed a grid search on different possible values of number of hidden neurans, momentum and learning reate. The results are shown in Table 1. As the difference between 15 hidden neurons and 30 hidden neurons (shown bolded in the table) is negligible, we keep 15 nerons as it is much faster.

We show the convergance of the selected settings in Fig. 2. It shows that our neural network is able to reduce the error fitting on the lookup table static data by decreasing the error from more than 6 to around **4**.

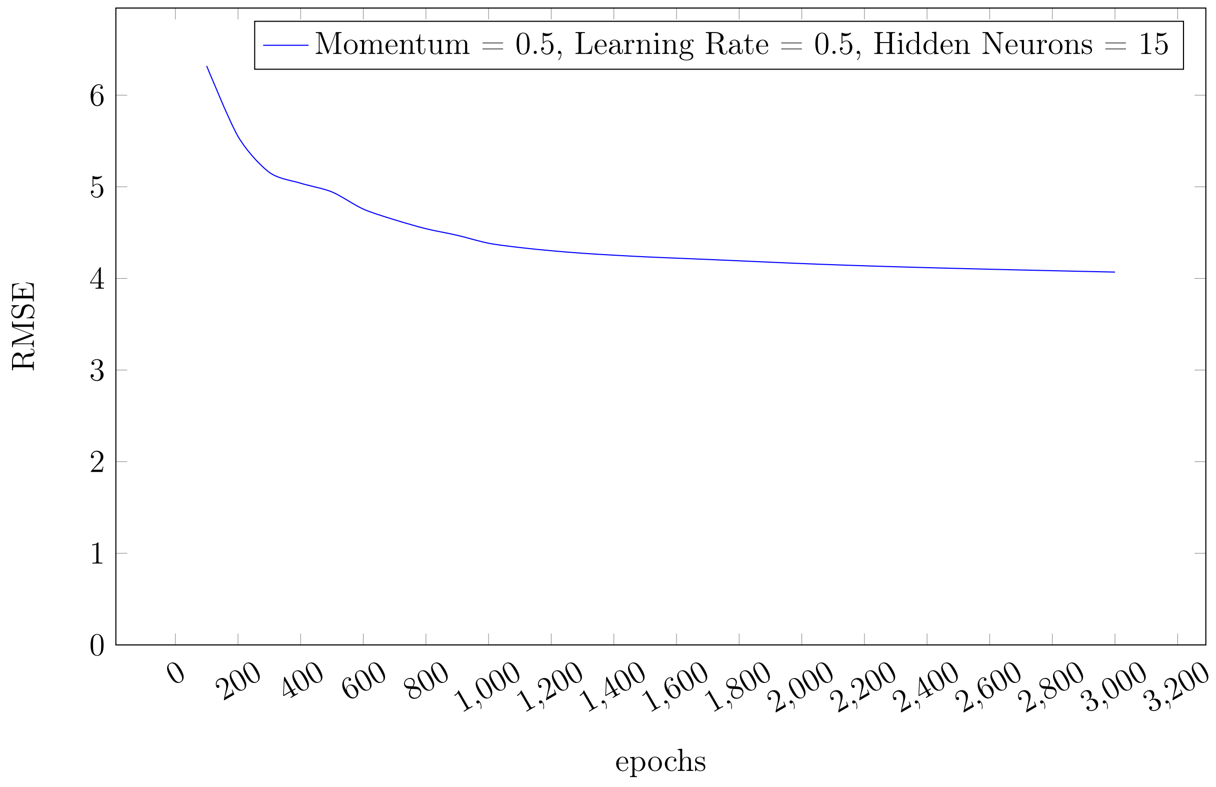


Figure 2: The convergence of the select hyperparameters over static lookup table data.

c) Comment on why theoretically a neural network (or any other approach to Q-function approximation) would not necessarily need the same level of state space reduction as a look up table. (2 pts)

As an example, we are using the distance to the enemy as one of the dimensions of the states. When using a lookup table, it matters to decrease the number of possible values to few, so the number of entries in the lookup table do not blow up and the get revisited often. But as the neural network treats the input as a real number and does not do exact-match like the lookup table, close values also let the states be somehow recalled by the neural network. Thus, using the large number of possible values for the distant to enemy is still tractable by the neural network. That said, it is still important to limit the range of that value to something close to the other inputs so the weight initialization would fit this dimension like the other dimensions.

(5) Hopefully you were able to train your robot to find at least one movement pattern that results in defeat of your chosen enemy tank, most of the time.

Identify two metrics and use them to measure the performance of your robot with online training. I.e. during battle. Describe how the results were obtained, particularly with regard to exploration? Your answer should provide graphs to support your results. (5 pts)

We measure the following metrics:

1. **Win rate**: the number of rounds won among 100 rounds.
2. **Win turns**: the average number of turns it takes our robot to defeat the enemy. We calculate this average only for the won rounds. Lower values are better as it means our robot is able to kill its opponent more quickly.

We interleave battles of training with a robot with $\epsilon = 0.8$ and test battles of 100 rounds with a robot with ϵ set to 0.05. We only report the metrics for the test robot (i.e. ϵ set to 0.05).

Appendices

A Source Codes