DM887 Assignment 1: Q-Learning for Tic-Tac-Toe

Ugurcan Ozalp

October 2022

1 Problem Definition

The first component of the problem is first to devise an tic-tac-toe game environment for playing. At each time step, opponent should have already played its move as optimal move. The environment has to have discrete state and discrete action space to allow tabular reinforcement learning methods applicable.

Second part is to create a learning agent which uses tabular q-learning using varying ϵ (for ϵ-greedy policy) and λ (for TD-λ) hyper-parameters. To make it possible, it is necessary to define a reward function. Then, results will be plotted for different hyper-parameters and results will be discussed.

2 Solution Method

2.1 Environment Definition

To implement tic-tac-toe game environment, we first defined a tic-tac-toe board which is allowed to be interacted by both players. In addition, all possible board states are analyzed and non-terminal valid board configurations are indexed as state. End-game configurations are indexed to single state according to the result, like $\mathcal{S}_{terminal} = \{x - win, o - win, draw, error\}$, where error is result of wrong actions (such as playing to a cell which is already filled). At the end, there are 4524 possible game states, although there are $3^9 = 19683$ possible board configurations (but most of them are invalid).

The opponent strategy is adapted to play the move move that makes it win at instant time step. If there is not such move, it checks if there is a move which prevents the main player to win on next time step. Otherwise, it randomly plays on valid cell.

2.2 Q-Learning Experiments

We run the tabular q-learning algorithm for $\epsilon \in \{0, 0.1, 0.2, 0.3\}$ and $\lambda \in \{1, 2, 3\}$. We run the same experiment 10 times and results are plotted according to these repeated tests on same hyper-parameter set. For all experiments, we adopted learning rate $\alpha = 0.8$ and discount factor $\gamma = 0.9$. Results are summarized in Figure 1 and 2.

3 Discussion

Although the problem seems easy, obtained results are not so good as expected. The main reason is that state space dimension is too high and discrete, so each board state is perceived as if it is unique and do not similar to any other board state. In addition to that, agent has to learn to play valid action and it has to try it for all board state separately.

For player X, the agent best performance when $\lambda = 1$ compared to the others, and performance decreases as ϵ increases. However, higher λ values yield best performance for player O unlike player X. One possible and simple explanation is that the transition function and optimal Q functions are completely different and not alike for player X and O, so the problem is almost different. Remember that the longest possible episode has 5 steps if everything goes well, so it is natural to be sensitive to λ.

Lastly, some experiments where $\epsilon = 0$ are stuck to single behaviour. It means that Q table is not updated and same behavioral policy is applied all time.

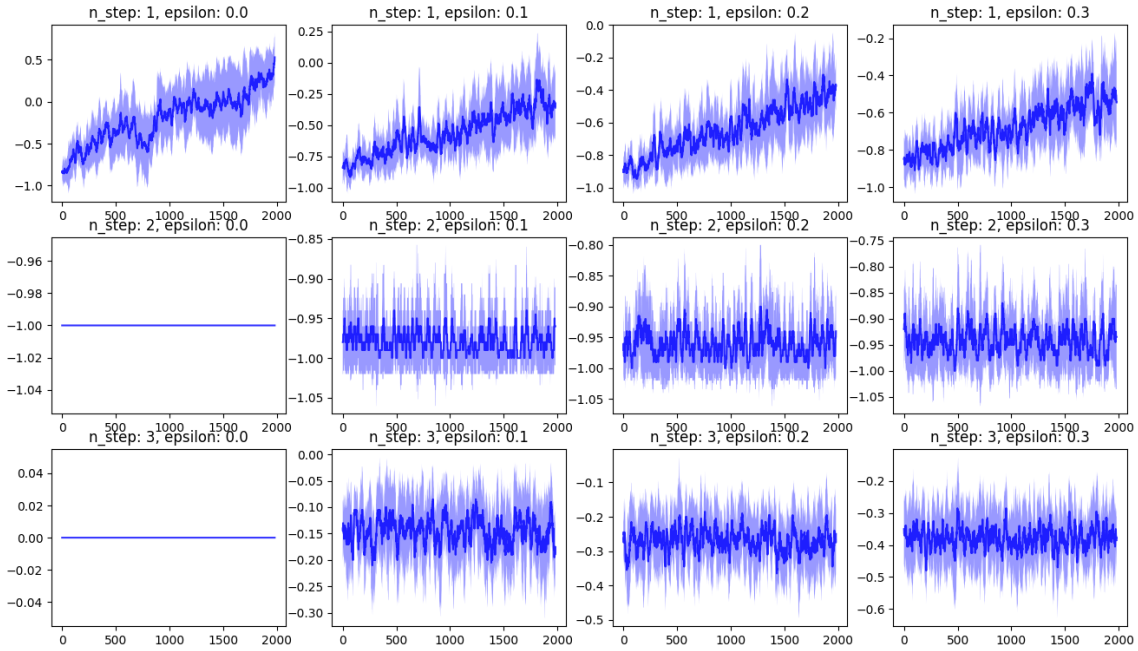

Figure 1: Learning history when player is X

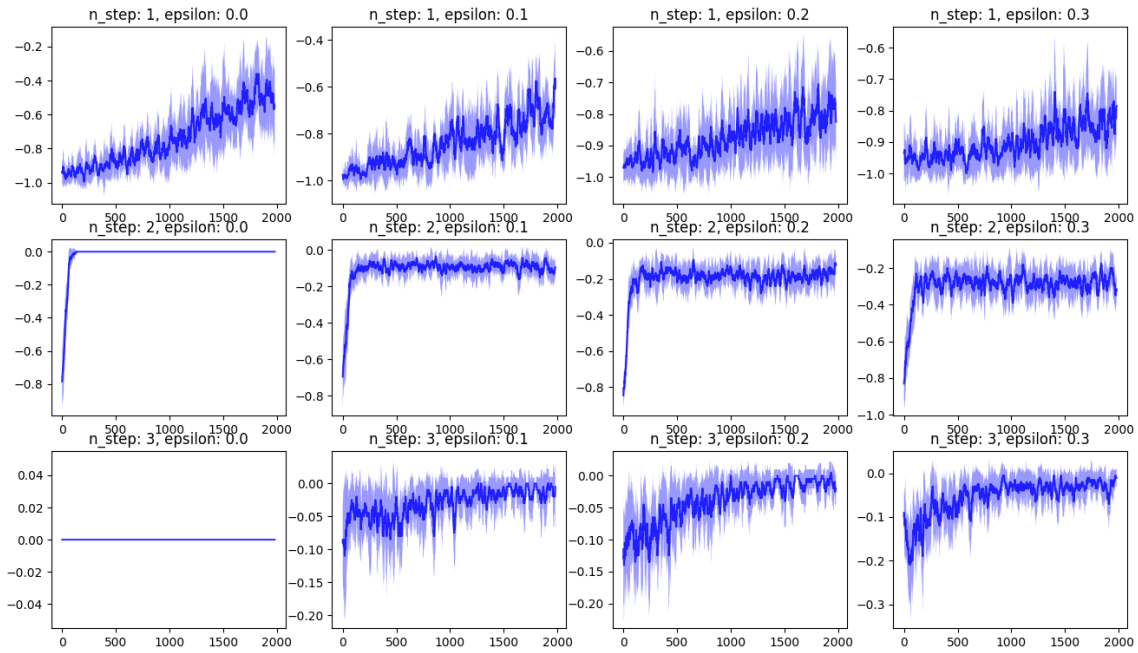

Figure 2: Learning history when player is O