Multiagent Coordination Solution based on Q-Learning Algorithm

Hyuntaek Oh
ohhyun@oregonstate.edu
Due. Oct 15, 2024

I. Introduction

In recent years, autonomous agents have emerged as a crucial area of research in the fields of robotics and artificial intelligence. These agents are designed to operate independently in certain environments, learning to achieve their own goals. A significant challenge in this research topic is how multiple agents with their individual and collective objectives can cooperate in shared environments to obtain optimal outcomes.

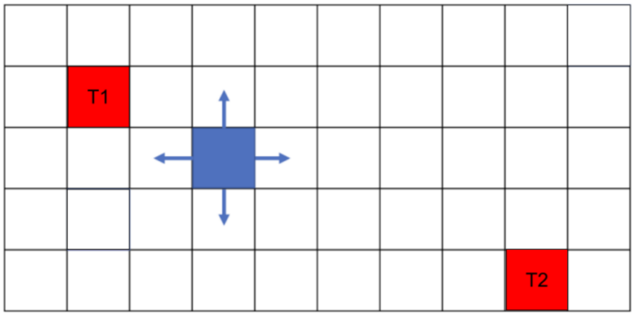

Fig. 1: A 5x10 grid-world, with agents in blue and two targets (T1 and T2) in red

As can be seen in Fig. 1, in a 5x10 gridworld, which is an environment, agents start at a fixed location, and move four possible directions. Agents receive a reward of 20 for capturing either target and a reward of -1 for every time step an agent moves in this gridworld without reaching and capturing a target.

In this assignment, there are three problems to be dealt with: single agent and single target, two agents and two targets without collaboration, and two agents and two targets with cooperation. These problems represent fundamental concepts in multi-agent systems, providing key questions that how to achieve individual and collective goals with single or multiple agents and how agents behave when acting independently but in shared spaces with cooperation.

Reinforcement learning (RL) is based on the simple observation that rewarding desirable behavior and discouraging undesirable behavior leads to behavioral change and has been utilized to handle the multiagent system problems. Q-learning in RL is the most commonly used examples of model-free temporal difference learning algorithms. Q-learning updates the Q-values of the state action pairs based on the ϵ-greedy action selection.

Our approach is based on introducing Q-learning to optimize Q-table. With Q-learning, we solve three problems in different ways respectively since the requirement and condition is not as same as they are. In method section, we introduce three different ways to handle each problem. Next, in experiments and results section, we focus on accumulative rewards of three different algorithms to clarify which method has better performance when two agents cooperate or not. After that, in discuss section, we discuss and analyze how agents work to obtain maximum accumulative rewards and the differences between the second and third solutions in terms of cooperation. In conclusion section, we summarize our algorithms and explain key findings from the experiments.

II. Method

In this section, we introduce approaches how to solve three problems respectively. From the first problem to the last problem, we develop our algorithms based on Q-learning with respect to the requirements and conditions step by step.

A. Problem Setup

The environment is gridworld, which has the size of 5 row and 10 columns, where agents and targets are placed in Fig. 1. At first, the whole gridworld initialized to the value of -1 since every time step agents move produce a reward of -1. Then, we set two targets at the position (1, 1) and (4, 8) with a reward of 20, which are target 1 and target 2 respectively.

In this grid world, the locations of agents are state expressed as row column pairs such as (2, 3). agents have four possible movements: up, right, down, and left. These movement can be expressed as *ACTIONS['up', 'right', 'down', 'left']*, and we use index-accessing to get a certain action. Each action is based on the current state of agents, which is an input of the function to yield an action from action space.

We define Q-table, which is an output of the algorithm, to identify which path agent can maximize total rewards from the start location. The Q-table is a matrix that the expected reward of four directions agents move in each grid, and all the elements of the matrix initialized to zero.

When agents capture their target, the current episode terminates, and then the location of agents initialized to initial location (2, 3), starting a new episode.

B. One agent and one target

In this problem, the objective of an agent is to reach T1 as known as target 1. As described above, an agent receive a reward of -1 whenever it moves each grid every time step. To maximize its reward and achieve its goal, the agent should move as few times as possible. With single agent Q-learning, an agent accomplishes its task and can find optimized route to capture target 1. For training, every action is affected by the current location of the agent as an input and ϵ, which is a value of 0.1 for the possibility of taking random actions. The chosen action and current state point to the next possible location of the agent. During this process, the agent receive a reward of -1 or 20 from the environment. Based on the reward agent obtained, the formula of temporal difference is used to update the current Q-table. The temporal difference formula is:

$$TD = reward + (\gamma \times \arg\max_{a} Q(s', a'))$$

where γ is discount factor and has a value of 0.9. With the product of α, which is learning rate and has a value of 0.9, and temporal difference, we can replace old Q-table with newly updated Q-table. After 20 times of iterations, updated Q-table as an output of Q-learning shows which path maximize total reward the agent receive.

C. Two agents and two targets with independent learning

The problem requires to introduce one more agent and target. The additional task is that the second agent captures target 2. We use the same method as previous one did but independent learning. Each agent has its own Q-tables to achieve its objective, for this, they have individual reward systems, not sharing, meaning that agent 1 optimizes its path to capture target 1 and agent 2 also optimizes its path to capture target 2. However, since there are two agents that prioritize their own goals, we create a new condition that the episode terminates when one of them captures its target. For Q-learning, the system parameters are identical to previous setting. $\alpha = 0.9$, $\gamma = 0.9$, and $\epsilon = 0.1$. The number of episode is 50,000 times.

D. Two agents and two targets with cooperation

Unlike previous problem, there are two main differences: global reward system and shared Q-table. The concept of global reward system is that both agents share a reward system and they receive the same reward, meaning that each agent's action affects other agent's action. Previously, we define the condition that prioritize their own goals, leading two agents to take self-greedy actions, maximizing its own reward. On the other hand, in this case, both agents take each agent's actions into consideration to have a maximum global reward in the shared reward system. The state of each agent, which are the locations of each agent, and ϵ are input for next actions of each agent, and the next actions of each agent move them to next locations respectively.

For fulfilling the requirements and conditions above in this problem, we create a global reward and one shared Q-table. Global reward is a variable storing the values that both agents receive when they take an action. We set an additional condition that both agents receive a reward of 40, leading agents to cooperate and maximize global rewards, when both agents capture their own target respectively at the same time.

Based on the Q-learning, we update one shared Q-table with global rewards in the same method previous temporal difference formula. The system parameters of Q-learning is identical to previous setting. $\alpha = 0.9$, $\gamma = 0.9$, and $\epsilon = 0.1$. The number of episode is also 50,000 times.

III. Experiments and Results

In this section, we experimented on three different Q-learning algorithms. The figures in the section are based on the system parameters in method section.

A. One agent and one target

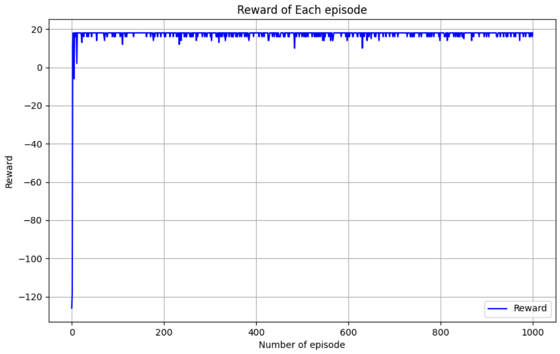

Fig. 2: Reward of each episode during training

As shown in Fig. 2, it plotted the rewards of single agent while training. During this process, the number of episode is 1,000 since agent optimize the path quite quickly. At the very beginning, agent received a reward of about -120, meaning that agent did not know about which action is best to capture target 1. After some episode, updated Q-table leads the agent to select optimal action to achieve its goal.

B. Two agents and two targets with independent learning

As can be seen in Fig. 3, it showed accumulative rewards of two agents using independent learning. In the figure, the reward of agent 2 gradually decreased until being about -25,000. Since the termination condition is that one of the agents captures its target, agent 1 took advantages of the proximity to target 1. On the other hand, agent 2 was far away from target 2, meaning that the probability of reaching target 2 is lower. Thus, due to the agent 1's self-greedy action, agent 2 receive relatively less rewards than agent 1 without any cooperation.

Fig. 4 showed the sum of two agents' rewards. The sum of rewards was calculated by each reward two agents received

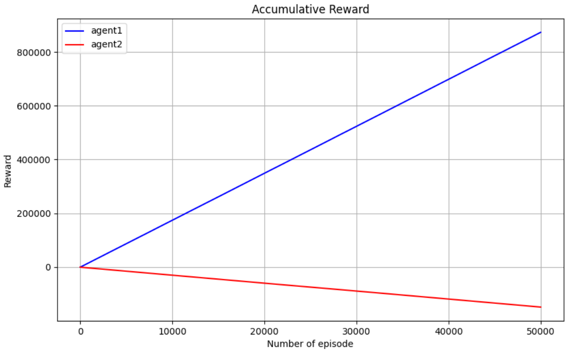

Fig. 3: Accumulative rewards of two agents using independent learning

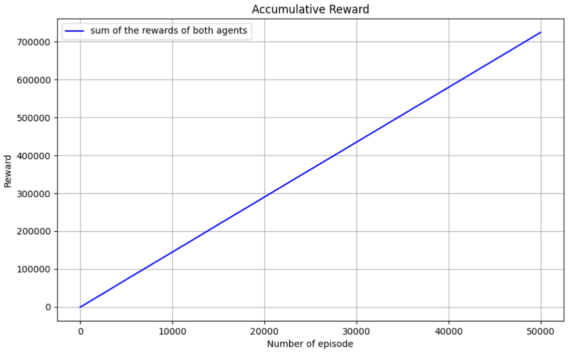

Fig. 4: Sum of both agents' rewards

at the same episode. It was for comparing with cooperation case since the total rewards of two agents can be used for the performance of independent works even though they independently learned and interfered each other.

C. Two agents and two targets with cooperation

The figure above displays the accumulative reward of three different cases. From the long-term respective, two agents with cooperation have superior performance than others after over 40,000 episodes, meaning that two agents struggled to receive the highest reward in each episode by accomplishing both objectives simultaneously to maximize global rewards.

IV. Discussion

A. Two agents and two targets with independent learning

With independent learning, two agents struggled to capture their own targets without considering the other's action. It seems the agents did not benefit each other, and even they interfered the other's task since the episode terminates when one of them captures its target earlier. Both agents actually did not collaborate with each other to maximize their total rewards in this problem.

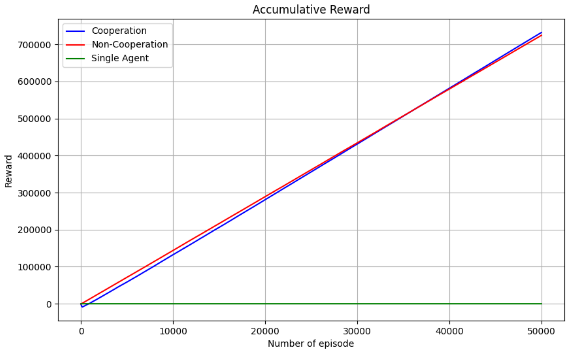

Fig. 5: Accumulative rewards of three different algorithms

B. Two agents and two targets with cooperation

While training, two agents with cooperation prioritized to maximize global rewards, and they waited for capture their own targets simultaneously even one of them located near the target. Fig. 5 above prove that both agents pursue the highest rewards at the same time. It was intentional that one agent moving around the target for the other. The cooperative behaviors between agents were explicitly manifested while accumulating total rewards.

V. Conclusion

In this assignment, we explored the performance of multiagent with three different Q-learning algorithms. Single agent quickly optimized the path to capture target 1. After introducing one more agent, without cooperation, two agents optimized their action selection to achieve its own objectives. On the other hand, with cooperation, both agents selected their actions based on global reward, maximizing total rewards. By comparing performance with independent learning with one with cooperation learning, cooperation between agents could bring greater benefits in terms of long-term rewards. If the tasks require team-working rather than individual tasking, cooperation between agents is possibly better than other techniques.