Correlated Q

May 20, 2021

Abstract

In this project, the results of Figure 3 in Greenwald and Hall (2003)[1] are reproduced. Four variants of Q learning algorithms are implemented based on Soccer, a Markov game example. Similar to the original paper, CE-Q and FF-Q learners all converge while Q-learning agent does not.

1 Introduction

Markov games are stochastic games that have the property of Markov transitions. Q learning does a good job in convergence and finding optimal policies in MDPs, but it does not always perform well in multi-agent Markov games. In this study, an experiment was conducted to compare the convergence results of four variants of Q learning methods: Q-learning, friend-Q, foe-Q and correlated Q.

Friend-Q and Foe Q, or FF-Q, are introduced in Littman (2001)[2]. In a two-player environment, as the name suggests, the two methods update the Q values by considering the opponent as a friend or a foe, respectively. It has been proved that FF-Q converges in general-sum Markov games

Correlated Q (CE-Q) is proposed by Greenwald and Hall (2003)[1] to generalize both Nash-Q and FF-Q and seek the correlated equilibrium. Four variants of CE-Q are introduced, including utilitarian (uCE-Q), egalitarian (eCE-Q), republican (rCE-Q), and libertarian (lCE-Q), each with different objective functions. In this experiment to solve the soccer game, uCE-Q is used. Utilitarian CE-Q learning method tries to maximize the sum of total rewards of all players. The update equation that uCE-Q follows will be explained later.

The second section of the study introduces the general rules of the soccer game. Section 3 explains the algorithm with more details in implementation. Section 4 presents the results of the four agents and discusses some differences. The last section discusses some conclusions and also pitfalls encountered during the implementation.

2 Soccer Game

The environment of the soccer game is built based on Litterman (1994)[3] and Greenwalk and Hall (2003)[1]. It is an excellent example of a zero-sum Markov game with no deterministic equilibrium policies.

The game has two players, player A and player B in this case. The field is a grid with 2 rows and 4 columns. The most left column and most right column are the winning goal area for player A and player B, respectively. Figure 1 shows one of the possible states of the game. In this state s, player B owns the ball (the circle). Each player has five possible actions: north (N), south (S), east (E), west (W) and stick (St). In each step, the two players execute actions in random order. When of the the player landed on one of the goal cells, the game ends. If the player moved into the his own winning goal cell, he scores +100 and the opponent scores -100. However, if the player moved into the opponent's goal cell, he scores -100 and the opponent scores +100.

The game also has an important collision rule. If a player's action execution would take him to the cell which is occupied by the opponent, the move does not take place. Meanwhile, the ball would always move to the stationary player. Taken the state SS as an example. If A moves first to the west, the move will not take place and the ball stays at B because B is not moving. If B moved first to the east, the move will not take place either but the ball will move to A.

Because of such stochastic actions, the game will not have deterministic policies for any player. Under the environment, four Q learning algorithms will be implemented to see if convergence exists for them.

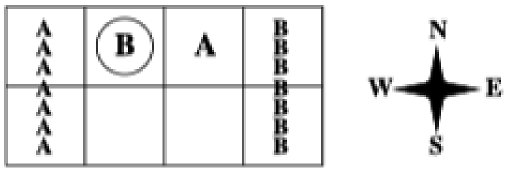

Figure 1: Soccer game. State SS. In this state, player B owns the ball.

3 Design of experiments

The four Q learning variants are all implemented with multi-agent Q learning algorithm. As shown in the pseudo code algorithm 1, the general procedure of the learning is the same for CE-Q, FFQ and Q learnings. There are some differences that are worth to mention.

Firstly, as pointed out in the original paper, in this study, CE-Q and FF-Q are off-policy learners and Q learning is on-policy. In general, off-policy means that the agent learns independently of the agent's actions while on-policy learners learn from the actions of current policy. However, the definitions does not apply in the original paper. Based on the followup paper, here, on-policy and off-policy are defined on the action profiles. That is, on-policy actions are determined from current policy. In this experiment, to make things simple, for off-policy learners such as CE-Q and FF-Q, the actions are selected with full exploration; for the on-policy learner of Q learning, the actions are decided with ϵ-greedy algorithm.

Algorithm 1: Multi-agent Q Learning

```
Input: state batches
Initialize action-value functions $Q_i$;
Initialize state-value functions $V_i$;
Initialize special state for plotting with state $SS$ and actions $\vec{A}$ =(S,St);
for time=1 to T do
    if game ends then
        reset the game
    end
    record $q = Q_a(SS, \vec{A})$;
    choose and execute action $a_i$;
    observe $s'$, $R_i$;
    update $V_i(s') = f_i(Q_1(s'), ..., Q_n(s'))$;
    update $Q_i(s, \vec{a}) = (1-\alpha)Q_i(s, \vec{a}) + \alpha[(1-\gamma)R_i + \gamma V_i(s')]$;
    if s=SS, $\vec{a}$ = (S,St) then
        record $q' = Q_a(s, \vec{a})$;
        record difference between $q$ and $q'$;
    end
    s = s';
end
```

Secondly, another major difference is how the state-value tables V_i are updated. For friend-Q, the agent treats the opponent as a friend and will try to coordinate the game. The update follows the equation:

$$V_i(s) = \max_{\vec{a} \in A(s)} Q_i(s, \vec{a}) \tag{1}$$

The agent will update the table only for the player itself. The equation can be considered a ordinary Q-learning in a two-player action space with $\vec{a}$ instead of a.

For foe-Q, the opponent is considered a foe. The learner will then seek an adversarial equilibrium, or in other words, will try to maximize its values (rational) when the expected values are minimized (adversarial).

$$V_1(s) = \max_{\sigma_1 \in \sum_1(s)} \min_{a_2 \in A_2(s)} Q_1(s, \sigma_1, a_2) = -V_2(s) \tag{2}$$

For CE-Q, the experiment follows the utilitarian algorithm (uCE-Q) which is in consistent with the original paper. The objective of the learner in each step is to maximize the sum of all players' rewards under some constraints.

$$\sigma \in \arg\max_{\sigma \in \text{CE}} \sum_{i \in I} \sum_{\vec{a} \in A} \sigma(\vec{a}) Q_i(s, \vec{a}) \tag{3}$$

In equation 3, σ is the probability distribution table we would like to solve in each iteration. Since the soccer game is a zero-sum game, the sum of the two players' rewards should always be zeros. The constraints can be considered from the perspective of equilibrium. Since we want to achieve a correlated equilibrium for the two players, any player should pursue the highest expected rewards given the opponent's action decision. For example, in the equilibrium, the expected reward for player A to take action N in state s is $\sum_{a \in A} \pi(N, a) Q(s, N, a)$ should always be no less than $\sum_{a \in A, b \in A, b \neq N} \pi(N, a) Q(s, b, a)$ for all other action b which is not N. (Technically the probability here should be conditional probability$\pi(a|N)$; I used joint probability to make it consistent with code implementation. By multiplying $\pi(N)$ on both sides, the results are the same) Such inequality should hold for all players and all possible actions.

In implementation, the discount factor γ is set to be 0.9, which encourages the player to end the game sooner. Learning rate α is decaying exponentially from 0.1 to 0.001. Such sequence satisfies the condition of convergence (square summable but not summable). In methods that ϵ-greedy algorithm is applied, the ϵ is also decaying from 1 to 0.001, encouraging more explorations at the beginning.

4 Result

4.1 Correlated Q Learning

As shown in figure 2 and figure 3, the two curves have the general similarity of convergence. However, the scores in my experiment at first 50 thousand iterations is increasing instead of random fluctuations. The possible reason could the setting on the initial state of each game. In my implement, the soccer games start with random non-terminal states. Although the original study did not tell us how they initialized the game, it is possible that the game could always start in state SS as pictured in figure 1.

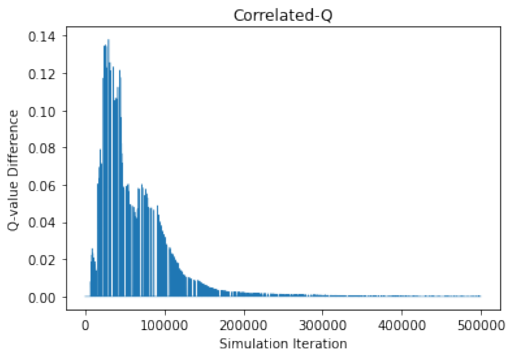

Figure 2: Replicated CE-Q Results

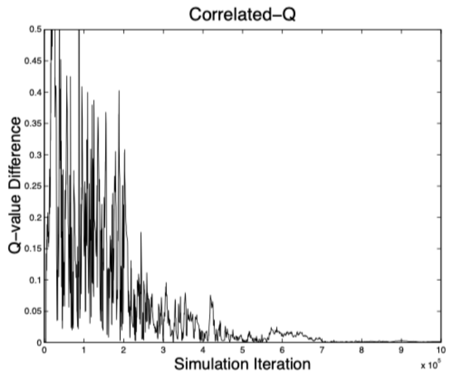

Figure 3: Original CE-Q Results

4.2 Foe Q Learning

The replicated error curve of Foe Q learning method is very similar to the original one. The learning agent also converges. However, unlike the original paper, it is not identical to the one in CE-Q. The two agents have performed differently in the game. I have not figured out the reason, but I have some assumptions. One reason could be the initialization of the game. Difference initializations may cause the difference in curves. Another reason could come from the linear programming. As mentioned before, Foe-Q focuses on the objective function (max-min) to find the optimal policy, while CE-Q focuses on the constraints of joint probability distribution because the objective function value is always zero. The optimization problem in CE-Q may not have just one solution each time. Difference solutions can lead to difference errors, but the impact will diminish when the learner is converging. Therefore, although the two learners do not have the identical start, the tails of the curve is similiar.

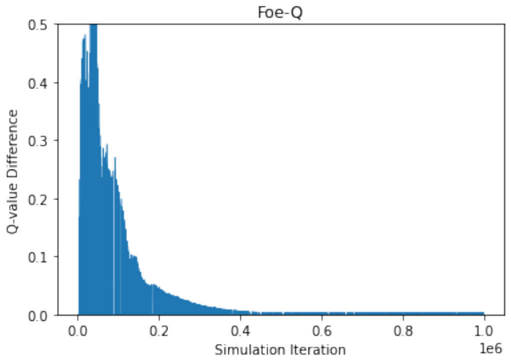

Figure 4: Replicated Foe Q Results

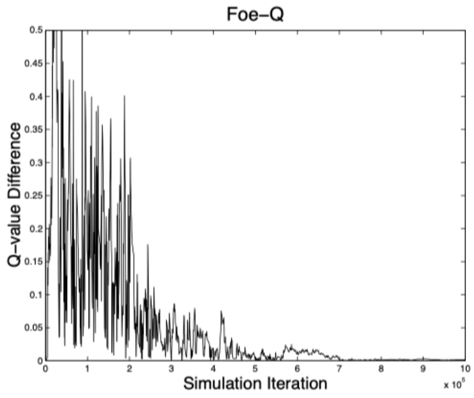

Figure 5: Original Foe Q Results

4.3 Friend Q Learning

Friend Q learning converges quickly with only round 10 thousand iterations. The result is basically identical to the one in the original paper. The actions for player B in state S will converge to East with expecting player A also executing East. The ball will eventually pass to A while A is in B's goal cells, making B scoring $+100$. The expectation of B is not the same as the original paper, but the conclusion is the same. Both my implementation and the original paper indicate that friend-Q will converge to a deterministic policy for player B.

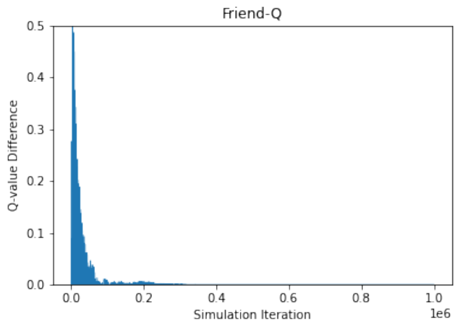

Figure 6: Replicated Friend Q Results

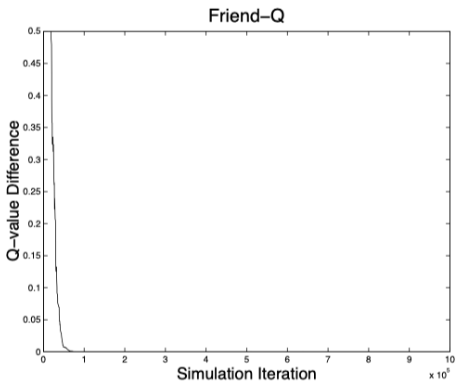

Figure 7: Original Friend Q Results

4.4 Q Learning

The Q learning agent does not converge as the original study. Although the curve looks like converged at the end of iterations, it is mainly because of the decaying of learning rate α. In the multi-agent algorithm, α decides how much the state-action pair value $Q(s, \vec{a})$ will change from next reward and state-value estimations. A decaying learning rate will always lead to convergence given enough time or iterations. In fact, by changing the decaying rate, the result can be quite different. In the implementation, the learning rate in algorithm 1 decays exponentially from 0.1 to 0.001. However, since the original paper did not tell us the speed of decaying, or whether they choose to decay α exponentially or linearly, it makes sense that there could be some differences on the final result.

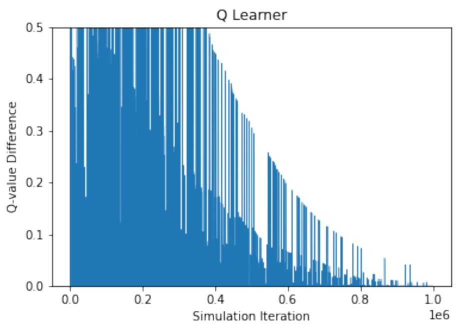

Figure 8: Replicated Q Learning Results

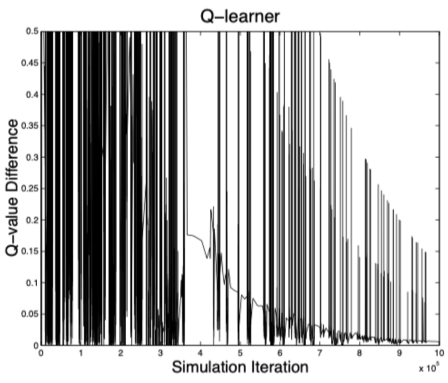

Figure 9: Original Q Learning Results

5 Discussion and Difficulties

Although Q learning has been proved to a reliable method to find optimal policy in MDPs, the method fails to converge in the stochastic soccer game. The Q agent only learns and explore actions based on the player's scores and fully ignoring the opponents' states and actions. The original paper has shown that Q learning can still converge in a deterministic multi-agent Markov game where a unique equilibrium exists. In an environment that has multiple equilibria, correlated Q-learning algorithm can have more reliable result.

During the implementation, some pitfalls were encountered and caused confusions in understanding how the multi-agent learners work.

Firstly, the original paper fails to explain how to initiate the game and experiment. It also does not tell us how they defined "simulation iteration" which is the x axis of the four figures. Since only the value in state SS is captured and not all epochs pass though the state, it raises the difficulty to reproduce the results.

Secondly, some settings of the environment are not explained with details. For example, the original paper tells us that both learning rate α and epsilon ϵ decays over time to 0.001, but not further information was introduced. I have experimented with several decaying methods and decaying rates, the results diff to a great extent.

Finally, another pitfall is the aforementioned definitions on on-policy and off-policy. The definitions determine how the players will choose their actions. But as Q-learning is generally considered as off-policy learner, the "on-policy Q-learning" saying is undoubtedly confusing to the audience.

References

[1] Amy Greenwald, Keith Hall, and Roberto Serrano. Correlated q-learning. In *ICML*, volume 3, pages 242–249, 2003.

[2] Michael L Littman. Friend-or-foe q-learning in general-sum games. In *ICML*, volume 1, pages 322–328, 2001.

[3] Michael L Littman. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier, 1994.