Proceedings of the 2nd International Conference on Computing Innovation and Applied Physics
DOI: 10.54254/2753-8818/19/20230542

An evaluation of reinforcement learning performance in the iterated prisoner's dilemma

Yifan Sun[1]

[1]Department of Mathematics, University of Toronto, Toronto, Ontario, M5R0A3, Canada

robert.sun@mail.utoronto.ca

Abstract. This paper uses recurrent neural network-based reinforcement learning to play Iterated Prisoner's Dilemma against different game theory strategies. Multiple experiments are carried out to compare the performance of the reinforcement learning agents, e.g., RL-agent vs. Tit for Tat, RL-agent vs. Grudge, RL-agent vs. Tit for Tat then Defect, RL-agent vs. Cooperate or Defect. It shows that both DQN and PPO agent would receive the highest reward by playing against a Tit for Tat agent in Iterated Prisoner Dilemma. Furthermore, DQN agent would perform better, by receiving higher mean episode reward compared to PPO agent.

Keywords: reinforcement learning, game theory, deep q-network, proximal policy optimization.

1. Introduction

Reinforcement learning adapts behavior through exploration and exploitation, and eventually maximize the agent's expected cumulative discounted reward [1]. In a usual reinforcement setting the action from an agent will not change the environment, but in a prisoner's dilemma game, an agent's reward depends on all the previous actions that were made.

Table 1. The payoff matrix for the Prisoner's Dilemma. The tuples are the rewards that each player able to receive based on their choice of actions. For instance, if Player 1 plays defect and Player 2 plays cooperate, then Player 1 receives a reward of 5 and Player 2 receives a reward of 0.

		Player 2	
		Cooperate	Defect
Player 1	Cooperate	(3, 3)	(0, 5)
	Defect	(5, 0)	(1, 1)

In this paper, author would demonstrate the experiments via iterated Prisoner's Dilemma (IPD), where two players play the game more than once in succession and the players will observe their opponent's previous actions and change their strategy based on that. The paroff matrix for the Prisoner's Dilemma is shown in Table 1. This work offers a complete comparison between the performance of DQN algorithms as well as the performance of PPO algorithms trained by different strategies, such as Tit or Tat Strategy, Grudge Strategy, Cooperate or Defect Strategy and Tit for Tat then Defect Strategy, to find a player strategy that let the reinforcement learning agent performs the best. Various experiments are

DOI: 10.54254/2753-8818/19/20230542

conducted via iterated prisoners dilemma (IPD), since in a single game of prisoners dilemma, the Nash equilibrium would be both agents defect [2].

2. Related Work

This paper is inspired from1996 paper by Sandholm and Crites [3] and the 2017 paper by Harper [4]. The first paper is about evaluations of reinforcement learning agents' performance on the IPD limited to Q-learning. In the second paper, several powerful strategies for the Iterated Prisoner's Dilemma were created using reinforcement learning techniques, in particular evolutionary and particle swarm algorithms. In comparison, this project used DQN and PPO algorithms to see given different strategies as opponents, which strategy would train the RL agent to give out the best performance in IPD.

3. Method

3.1. Parameters Setting

In the experiment we will use two Reinforcement Learning algorithms: Deep Q-Networks (DQN) and Proximal Policy Optimization (PPO). DQN is a value – iteration-based method and PPO is a policy – gradient type of method. The implementations of DQN and PPO were from the Ray Rlib Library [5], and the default configurations were used, with some little modification: DPN used noisy network to aid exploration, set the N-step for Q-learning to 3, minimum value estimation and maximum value estimation to -10 and 10 respectively. DQN has a hidden layer network of [256, 256, 32, 8]. PPO used a minibatch with size of 32 and trained on 20 epochs for each batch. PPO has a hidden layer network of [1024, 512, 512, 256, 256, 32, 8]. By the default of RLib configurations, one training iteration is 1000 timesteps for DQN, and 4000 timesteps for PPO.

3.2. Game Setting

The experiment is based multiple games, there are 8 experiments in total, 4 experiments for each Reinforcement Learning agent to evaluate the agent's performance. We call each game an episode, and there are 70 episodes for an experiment. Each episode consists of 100 prisoner's dilemma games. A prisoner's dilemma game has two players, and each player can choose one of the options: cooperate (C) or Defect (D) [6]. There is a payoff matrix that has four different kinds of reward: T, R, P, S. Where T = 5, R = 3, P = 1, S = 0. In our experiments, one player will be a Reinforcement learning agent and the other player will play different strategies.

3.3. Reinforcement Learning Algorithms

In this paper, two reinforcement agents would be used, which are DQN and PPO. Both reinforcement learning agents have memory, in which the agent makes his actions based on the previous actions of the opponent and itself. In DQN, a random minibatch is sampled per update, but in PPO all the experience in memory is used.

3.3.1. Deep Q Learning Algorithms. The DQN algorithm is a model-free, online, off-policy reinforcement learning algorithm [7]. DQN is a variant of Q-learning, instead of using a Q-table in Q-learning, DQN uses a deep neural network to approximates a state-value function in a Q-learning framework. The DQN algorithm has two key enhancements compare to the Q-learning algorithm:

1. A experience replay buffer used for storing the episode steps in memory for off-policy learning, where samples are collected from the replay memory at random.
2. The DQN is optimized towards a frozen target network that is periodically updated with the latest weights every k steps, in order to address the instability caused by chasing a moving target.

These changes allowed a successful training process of a DQN, an action-value function approximated by a convolutional neural net, on the high dimensional visual inputs.

DOI: 10.54254/2753-8818/19/20230542

3.3.2. Proximal Policy Optimization Algorithms. The Proximal Policy Optimization, or PPO, is a policy gradient method for reinforcement learning, that provides an improvement on Trust Region Policy Optimization (TRPO) [8]. The motivation was using only first-order optimization while keeping the data efficiency and reliable performance. Let $r_t(\theta)$ denote the probability ratio $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$, so $r(\theta_{old}) = 1$. TRPO maximizes a "surrogate" objective:

$$\mathrm{L^{CPI}}(\theta) = \hat{\mathrm{E}}_\mathrm{t}\left[\frac{\pi_\theta(\mathrm{a_t}|\mathrm{s_t})}{\pi_{\theta_{\mathrm{old}}}(\mathrm{a_t}|\mathrm{s_t})}\hat{\mathrm{A}}_\mathrm{t}\right] = \hat{\mathrm{E}}_\mathrm{t}\left[\mathrm{r_t}(\theta)\hat{\mathrm{A}}_\mathrm{t}\right] \quad (1)$$

Where CPI means a conservative policy iteration.

However, in the process of maximizing L^{CPI}, the policy update would be too large, in the case where there is no constraint. PPO modifies the objective, to penalize changes to the policy that move $r_t(\theta)$ away from 1:

$$J^{CLIP}(\theta) = \hat{E}_t\left[\min\left(r_t(\theta)\hat{A}_t,\ clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right] \quad (2)$$

Where ϵ is a hyperparameter. The first term inside the minimum function is L^{CPI}. The second term, $clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t$ removes the incentive for moving r_t outside of the interval $[1-\epsilon, 1+\epsilon]$. By taking the minimum of the clipped and unclipped objective, the change in probability ratio would be ignored when it improves the objective, and the change in probability ratio would be included when it makes the objective worse.

3.4. Fixed Strategies

In each experiment, a reinforcement learning agent would play against a player that plays one of the four different game-theory strategies, which are Tit for Tat, Cooperate or Defect, Tit for Tat then Defect and Grudge. A Tit for Tat strategy is implemented when the player cooperates with another player in the very first interaction and then mimics their subsequent moves [9]. A Cooperate or Defect strategy is simply played by always defect or cooperate in each episode. A Grudge player would start by cooperating, but defect forever if opponent defects. A Tit for Tat then Defect strategy is similar to Tit for Tat strategy, but then on a random turn begins defecting forever.

4. Result

4.1. DQN Agent vs. Fixed Strategy

The performance of the DQN agent is showed by checking the episode's mean reward in Figure1. DQN agent received the highest reward when playing against player using Tit for Tat strategy, compared with player using Grudge or Tit for Tat then Defect strategy. A DQN agent would receive a mean episode reward closer to 300 as the number of episodes increase, when playing against Tit for Tat strategy. Although DQN agent would receive a higher episode max reward when playing against player using Cooperate or Defect strategy, there is a large gap between its maximum and minimum episode rewards, the mean episode rewards gradually increase and eventually bouncing around 280 as the number of episodes increase, with a lot of fluctuation in the mean episode reward. Therefore, the mean episode reward from Tit for Tat is the optimal strategy for a DQN agent to play against with.

DOI: 10.54254/2753-8818/19/20230542

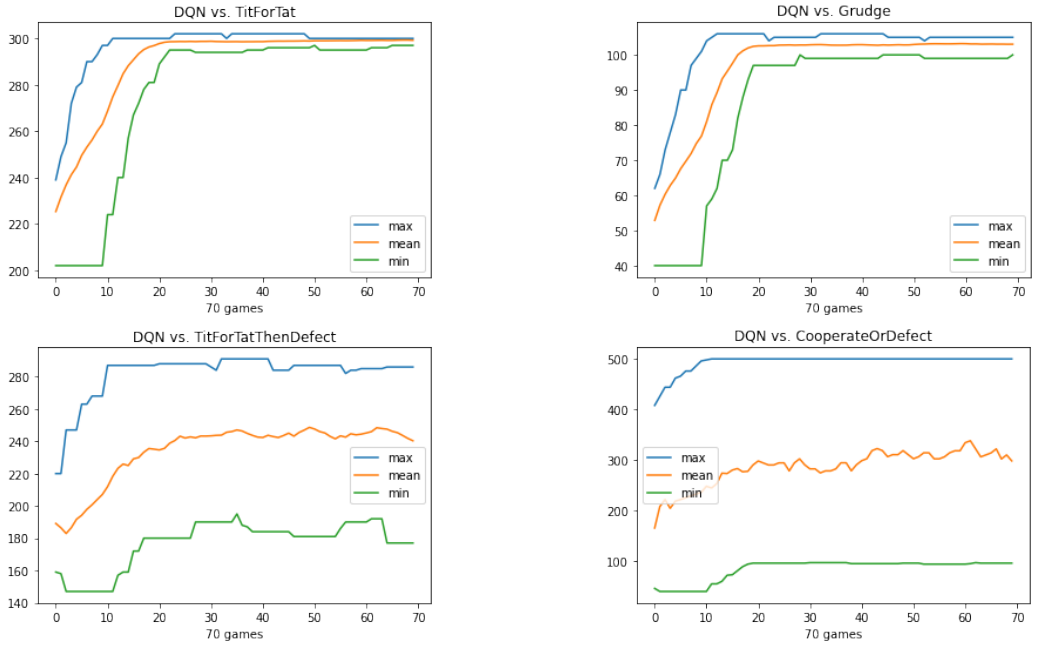

Figure 1. DQN agents play against Tit for Tat, Grudge, Tit for Tat then Defect and Cooperate or Defect. The maximum, mean and minimum episode rewards of DQN agents are showed.

4.2. PPO Agent vs. Fixed Strategy

The PPO agent received the highest reward when playing against player using Tit for Tat strategy, compared with player using Grudge or Tit for Tat then Defect strategy as shown in Figure 2. When a PPO agent plays against Cooperate or Defect strategy, the mean episode reward is fluctuating between 200 and 300 as the number of episodes increase, and most of the time the mean episode reward is below 240. In comparison, When the agent plays against Tit for Tat, the mean episode reward keeps increasing. Due to the fact that each experiment was restricted to 70 episodes, the reward achieved just a little bit above 240, if there are more episodes been played the reward will be higher, and much stable.

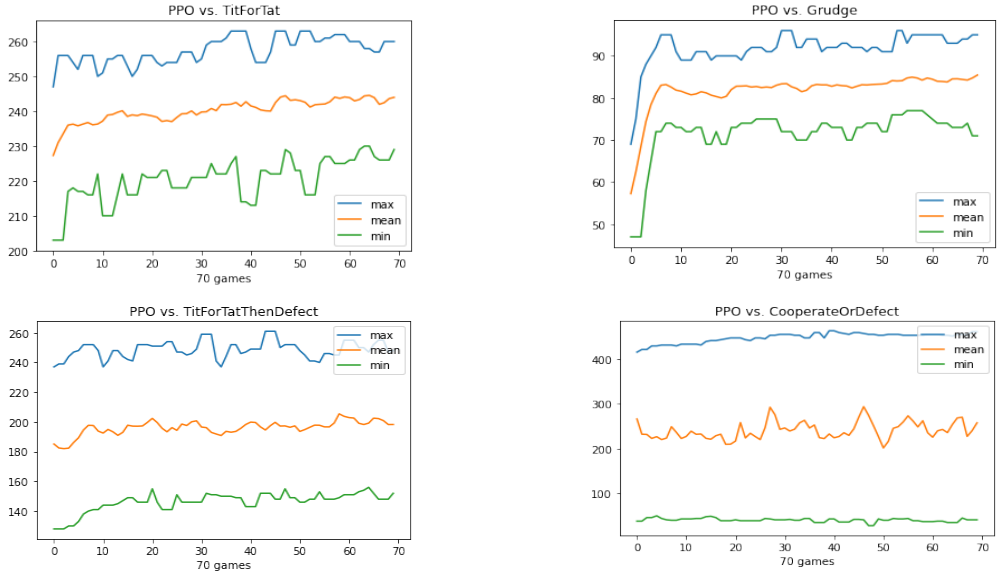

Figure 2. PPO agents play against Tit for Tat, Grudge, Tit for Tat then Defect and Cooperate or Defect. The maximum, mean and minimum episode rewards of PPO agents are showed.

DOI: 10.54254/2753-8818/19/20230542

5. Conclusion

This paper presents an evaluation focusing on the performance of reinforcement learning agents in the Iterated Prisoner's Dilemma. It shows that both DQN and PPO agent would receive the highest reward by playing against a Tit for Tat agent in Iterated Prisoner Dilemma. Furthermore, DQN agent would perform better, by receiving higher mean episode reward compared to PPO agent. There was variance across experiment. For DQN agent, one solution would be to average previously learned Q-values estimates, which leads to an improvement on the performance by reducing approximation error variance in the target values [10]. Furthermore, there are many ways to continue this work, one way is to first train either a PPO agent or DQN agent, then use it to train the other reinforcement agent, and vice versa. It will be interesting to see if the reinforcement learning agent performs better when trained by another reinforcement learning agent.

Acknowledgments

This work was done as part of the Deep reinforcement Learning Course taught by Prof. Pietro from the department of Computer Science and Technology of the University of Cambridge. I would like to thank Prof. Pietro and Jacky for the support.

References

[1] Sutton RS, Barto AG. Reinforcement learning: An introduction. MIT press; 2018 Nov 13.
[2] Rapoport A, Chammah AM, Orwant CJ. Prisoner's dilemma: A study in conflict and cooperation. University of Michigan press; 1965.
[3] Sandholm TW, Crites RH. Multiagent reinforcement learning in the iterated prisoner's dilemma. Biosystems. 1996 Jan 1;37(1-2):147-66.
[4] Harper M, Knight V, Jones M, Koutsovoulos G, Glynatsi NE, Campbell O. Reinforcement learning produces dominant strategies for the iterated prisoner's dilemma. PloS one. 2017 Dec 11;12(12):e0188046.
[5] Liang E, Liaw R, Nishihara R, Moritz P, Fox R, Goldberg K, Gonzalez J, Jordan M, Stoica I. RLlib: Abstractions for distributed reinforcement learning. In International Conference on Machine Learning 2018 Jul 3 (pp. 3053-3062). PMLR.
[6] Luce RD, Raiffa H. Games and decisions: Introduction and critical survey (new york, 1957). Chs. vi and xiv. 1957;4.
[7] Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller M. Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602. 2013 Dec 19.
[8] Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347. 2017 Jul 20.
[9] Axelrod R. Effective choice in the prisoner's dilemma. Journal of conflict resolution. 1980 Mar;24(1):3-25.
[10] Anschel O, Baram N, Shimkin N. Averaged-dqn: Variance reduction and stabilization for deep reinforcement learning. InInternational conference on machine learning 2017 Jul 17 (pp. 176-185). PMLR.