B-109

On the Performance of DQN in 2D and 3D Game Environments

ZHANG YIMING[1] is0386es@ed.ritsumei.ac.jp Ritsumeikan University
GAO RUOYU[1] is0489pr@ed.ritsumei.ac.jp Ritsumeikan University
THAWONMAS RUCK[1] ruck@is.ritsumei.ac.jp Ritsumeikan University
HARADA TOMOHIRO[1] harada@ci.ritsumei.ac.jp Ritsumeikan University

1. Abstract

This paper discusses the performance of Deep Q-Network (DQN) in 2D and 3D game environments. In this paper, we analyze reasons for the poor performance of DQN in the 3D game environment. We then propose to use Inverse Reinforcement Learning (IRL) to solve the detected issues.

2. Introduction

Recent years, reinforcement learning methods are applied to some 2D-games and perform well. For example, Random Network Distillation (RND) [1] scores over 10000 points in an Atari game called Montezuma's Revenge, which is beyond the average of human players. Hierarchical Reinforcement Learning (HRL) with feature control [2] also gets outstanding performance in Montezuma's Revenge and Zaxxon.

However, we speculate that those methods are only optimal for 2D environments and do not demonstrate whether the outstanding performance can be similarly achieved in 3D environments, which is our research question in this work. To answer the aforementioned research question, we choose DQN [3] with the TensorFlow framework for both environments to compare the DQN's performance in them.

As a 2D-game environment, we used CartPole (Fig.1) [4] of OpenAI Gym and set the learning rate to 0.001, maximum training time steps to 100000 (about 700 episodes), discount rate to 1.0, and neural network to Multi-Layer Perceptron.

(a)

(b)

Figure 1: CartPole—(a) gets reward and (b) loses reward because the pole has inclined

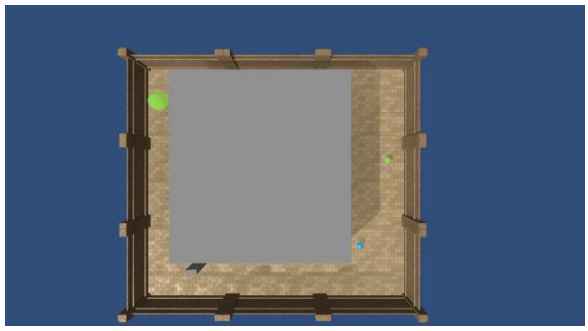
[1]Intelligent Computer Entertainment Laboratory, College of Information Science and Engineering

The reason why we choose this problem is that CartPole is a classic and excellent problem for DQN, and it is always used into the research of DQN and related methods.

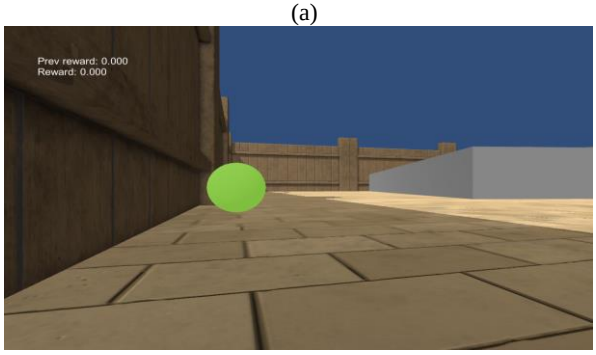
For a 3D-game environment, we used Animal-AI (Fig.2) [5] with our config having two positive rewards (a big green ball and a small green ball), one agent (a blue one) and two obstacles.

We choose this environment because it is a typical simulated 3D animal playground with sparse rewards. Therefore, we can use it to test the performance of DQN in a sparse reward 3D environment.

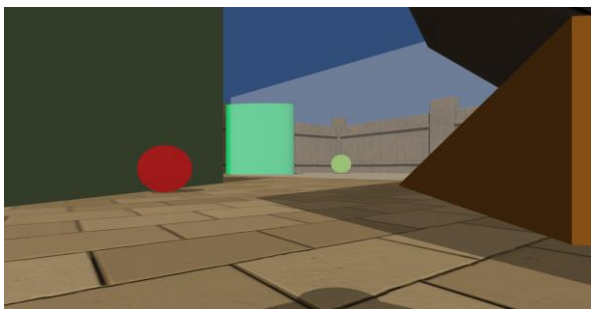
In the 3D environment, we test two versions of DQN: one with the training time step of 100000 (about 100 episodes) and the other of 400000 (about 400 episodes); the other parameters are the same as those in the 2D environment.



(a)



(b)



(c)

Figure 2: Animal-AI—(a) default camera, (b) agent camera aiming at a goal (the big green ball), and (c) agent camera aiming at a wrong object (the green box)

3. Results

After training in the CartPole environment, the DQN model could make the trained agent always get the max reward of 200.0 in every episode among 10 test-run episodes. In other words, the agent performed well in all the episodes.

After training in the Animal-AI environment, DQN with 100000 steps could not find any rewards in among 10 test-run episodes. DQN with 400000 steps could find the max reward of 3.0 (the big green ball) in episode 1 and the reward of 1.0 (the small green ball) in episode 2, but could not find any reward in the subsequent episodes. The agent always became confused in the arena, which is the game space.

4. Analysis

The first reason why the agent can not find more rewards in the 3D Animal-AI environment is that the environment is complex and that rewards are sparse. The agent might, for example, touch a reward when it is falling back in the first training episode, and it will learn that the falling back action is a rewarding action and will always do the action in subsequent episodes. Therefore, in future episodes the agent cannot learn the fact that the real rewards are the two types of green balls, but not the falling back action.

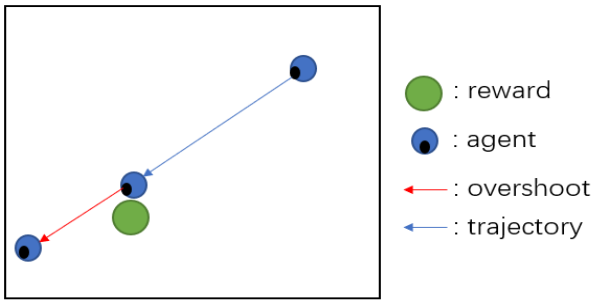


Figure 3: A scenario where an overshoot occurs

The second reason is that each action—among go forward, fall back, turn left and turn right—has inertia, so if the agent takes an action for a long period of time, it will easily overshoot (Fig.3) and reach an unknown area. Therefore, if the agent cannot accurately always aim the current target, it will get lost in the arena and cannot obtain more rewards.

5. Discussion

To solve these problems, our idea is to introduce IRL [6] into DQN. Figure 4 outlines our idea.

IRL can find a reward map or a reward function from the environment and the behavior of the agent, and it is also used in video games.

From Fig.4, in step (1), we use DQN and spend enough time to train the agent to find rewards. Then, we record the agent's trajectories where rewards are found and select the best one as the expert trajectory for step (2).

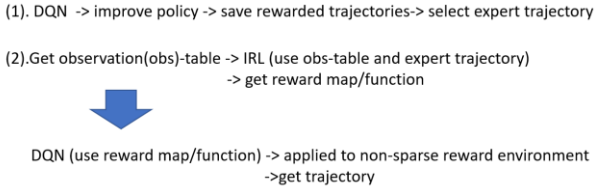


Figure 4: DQN with IRL

In step (2), we use the expert trajectory from step (1) and the observation-table from the camera input to train IRL. Then, we can get a dense reward map with the access frequency or reward function to solve the sparse reward problem. We can get better trajectories where the agent can find more rewards quickly.

With this idea, we can expect that the agent will not do excrescent actions, which solves the first problem, and that the proposed idea will make the agent's movement more accurate, which solves the second problem.

6. Future work

We will continue optimizing DQN in the Animal-AI environment, and actually combine it with the IRL method for the agent to perform well in sparse reward environments. We will also test the efficiency and practicability of this idea in more 3D-game environments and try to also apply it to robot navigation [7] [8].

Acknowledgement

The authors wish to thank all the members in our seminar for their fruitful discussion and support, in particular, the members of the Animal AI project. This research was supported in part by Strategic Research Foundation Grant-aided Project for Private Universities (S1511026), Japan, and by Grant-in-Aid for Scientific Research (C), Number 19K12291, Japan Society for the Promotion of Science, Japan.

References

[1] YuriBurda, Harrison Edwards, Amos Storkey, and Oleg Klimov. EXPLORATION BY RANDOM NETWORK DISTILLATION. arXiv:1810.12894v1 [cs.LG] 30 Oct 2018.

[2] Nat Dilokthanakul, Christos Kaplanis, Nick Pawlowski, and Murray Shanahan. Feature Control as Intrinsic Motivation for Hierarchical Reinforcement Learning.
DOI: 10.1109/TNNLS.2019.2891792

[3] Volodymyr Mnih, et al. Human-level control through deep reinforcement learning. Nature volume 518, pages 529–533, 2015.

[4] Ádám Budai and Kristóf Csorba. Deep Reinforcement Learning: A study of the CartPole problem. Cscs2018, page 17.

[5] Animal-AI Olympics: http://animalaiolympics.com/

[6] AaronTucker, Adam Gleave, and Stuart Russell. Inverse reinforcement learning for video games. arXiv:1810.10593v1 [cs.LG] 24 Oct 2018.

[7]. Liulong Ma, Yanjie Liu and Jiao Chen. Using RGB Image as Visual Input for Mapless Robot Navigation. arXiv preprint arXiv:1903.09927, 2019 - arxiv.org.

[8]. Liulong Ma, Yanjie Liu, Jiao Chen and Dong Jin. Learning to Navigate in Indoor Environments: from Memorizing to Reasoning. arXiv preprint arXiv:1904.06933, 2019 - arxiv.org.