~~Using~~ Hierarchical Multiple Agent Reinforcement Learning in Unity ML-Agent

Demonstration

ABSTRACT

Hierarchical ~~Critics~~ divides the observations into different level so that could capable to solve more complicated problems in reinforcement learning. By extending the ~~original~~ trainer from Unity, we try to adopt with the hierarchical ~~critics~~ into the training to see the hierarchical structure could be applied into the multiple agent environment. In this demonstration, we modify scenarios from Unity into multiple agent environment and train agents with hierarchical ~~critics~~. There are totally four scenarios from Unity, Tennis, Soccer, Banana Collector and Crawler, are modified to test with the training algorithm. The testing result show that the modified scenarios could successfully applied with the hierarchical ~~critics~~ reinforcement learning in multiple agent environment. **The demonstration video could be viewed from this link: https://youtu.be/CVNlgnKWodg**

KEYWORDS

Hierarchical Actor-Critics;Multi-agent Reinforcement Learning;Unity MLAgents Toolkit

1 INTRODUCTION

Reinforcement Learning aims to train agent by giving certain amount of reward on a particular action. Agent has to try several rounds to gain the experience from training to solve different problems [6]. There are different ways to optimise the training performance, including policy optimisation, ~~reassign critics into different levels~~. In recent years, the Proximal Policy Optimisation (PPO) [5] simplifies the training implementation to handle more complicated scenarios in single agent environment. Nonetheless, this creates another open area when the reinforcement learning comes into multiple agent system (MAS). It is not just simply transferring the algorithms into multiple agent system, but also involving the training adjustment between learning agents under the same environment [4].

The hierarchical reinforcement learning reassigns the critics arrangement into different level. A virtual manager is added on top of all working agents in the environment to observe globally. Meanwhile, worker agents have local observation and takes an advice from the virtual manager to perform an action. There are different researches try to adopt hierarchical structure into reinforcement learning. The hierarchical critics assignment in reinforcement learning from Cao [2] demonstrate the training performance result is better than the training without hierarchical critics. This motivates us to apply hierarchical reinforcement learning into different scenarios from Unity to see the training performance could be better than the original trainer.

There are different platforms provide framework for developer to design different scenarios by using reinforcement learning. There are two mainstream platforms, OpenAI Gym [1] and Unity ML-Agent toolkit [3], provides trainers in reinforcement learning. OpenAI Gym provides various tools and framework in developing reinforcement learning and evaluating performance, ~~including rrlab and Roboschool.~~ Meanwhile, Unity also provides similar functionalities and framework in training agent, but they also provide flexibility to customise the training. ~~In addition, Unity provides examples for developer to get start the scenario development with reinforcement learning in the Unity environment.~~ In this demonstration, we pick Unity ML-Agent as our starting point to demonstrate on the idea of the hierarchical reinforcement learning since it could provides more flexibility on customising ~~the training~~.

We pick four scenarios from Unity to be the base of our demonstration. We want to demonstrate the hierarchical reinforcement learning could be applied into different situation in training. We are more focus to apply competition between multiple agents under the same environment. First and foremost, Tennis scenario is modified by expanding the number of agent to construct team-based competition. Soccer scenario also expand the team size to evaluate the attack and defence situation could be apply with hierarchical reinforcement learning. In addition, Banana collector scenario also modified from same team training into competition between teams. Agents are allocated into different team to make sure the team could get the largest reward. Lastly, The agent from crawler scenario is also modified from learn-to-walk to learn-to-fight and see if the hierarchical reinforcement learning could be applied into this situation.

2 EXPERIMENTS

The experiments on each example is conducted in two phrases, baseline and hierarchical critics. Our baseline training uses the original PPO algorithm from Unity to train the agent. After that, we setup a hierarchical critics to perform training. There are totally four different scenarios are used to demonstrate experiment results, which is Tennis, Soccer, Banana Collector and Crawler. All scenarios are slightly modified to fit with the competition setup. During the evaluation of the experiment result, we contrast the game score in testing from the baseline trainer and the trainer with hierarchical critics.

2.1 Tennis Competition

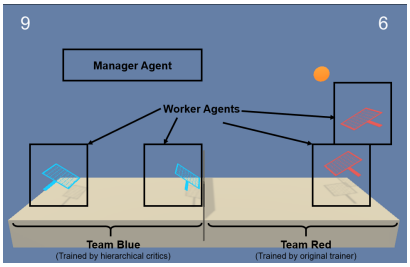
Tennis Competition is an example which is used to simulate a sport game to bounce ball into opponents' area under multiple agent environment. The agent is required control the movement of the

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 2020, Auckland, New Zealand

https://doi.org/doi

racket to make sure the ball is not dropped or out of bounds of its own area. It is a simple example to demonstrate the multiple agent training. In this demonstration, we increase the number of agents into two agents on each side to show that the hierarchy critics could work in multiple agent environment. Therefore, we make some adjustments based on the original Unity example. The virtual manager is setup on top of all worker agents to observe the global environment, while the worker agents observe the local environment to construct an action. The experiment setup are as follows,

Figure 1: Illustration of Tennis Competition in Hierarchical Structure



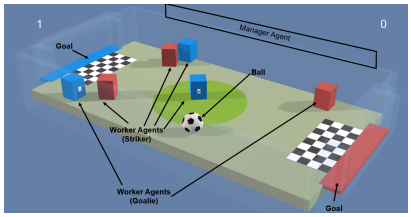
- **Objective**
  - Agent should figure a method to make sure they do not miss the ball or out of the court area during the episode by sending the ball over the net into opponents area.
- **Reward**
  - +0.1 when the ball is hit over the net
  - -0.1 when agent is missed to hit the ball or the ball out of the tennis court
- **Action Space**
  - Size of 2, which is representing the movement forward or away from the net as well as jumping.
- **Observation Space**
  - 10 variables with the position and velocity information of ball, racket and teammate.

2.2 Soccer Competition

The Soccer scenario is another example to show the worker agent work as a team to defend their goal and attack to the opponent's goal at the same time. A virtual manager agent also setup on top of all worker agents. Each team tries to attack another team's goal and defence the ball not to being kick into team's gate. Each agent has different behaviours and goal during the game. This team set up is good to show the hierarchy structure could also be applied under the Soccer scenario. It demonstrates the training in a more complicated scenario, which includes attacks and defence at the same time. The manager at here observe the global observation and coordinate the decisions between agents. The worker agent also take action base on the local observation.

- **Objective**
  - Striker agent has to figure out a method to kick the ball into opponent's gate.

Figure 2: Illustration of Soccer Competition in Hierarchical Structure

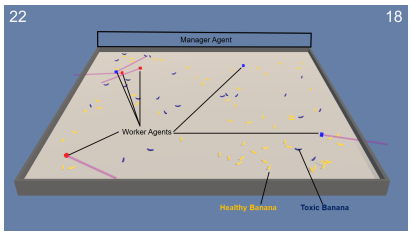


  - Goalie agent has to figure out to defence to avoid the ball go into its own gate.
- **Reward**
  - Striker: +1 when the ball enters opponent's gate, -0.1 when the ball enters own team's gate.
  - Goalie: -1 when the ball enters own team's gate, +0.1 when the ball enters opponents gate.
- **Action Space**
  - Striker: 6 actions (forward, backward, rotation and sideways movement)
  - Goalie: 4 actions (forward, backward and sideways movement)
- **Observation Space**
  - 112 variables to detect 7 types of object and also the distance. 180 degree view in front of the agent

2.3 Banana Collector

Banana Collector is another multi-agent scenario which agents aim to collect a healthy banana from the environment. The environment consist different type of banana, healthy banana and toxic banana. Agent should learn how to move and collect a correct banana since only healthy banana could give them a reward. A manager is also setup on top of all working agents to observe the whole environment. This scenario could show the training performance in making a correct decision during training.

Figure 3: Illustration of Banana Collector in Hierarchical Structure



- **Objective**
  - Agent must collect correct type of banana as much as possible.
- **Reward**
  - +1 when agent get a yellow banana
  - -1 when agent get a purple banana
- **Action Space**

  - 4 Branches of Action
    * Movement Branch - Forward, Backward or No Action
    * Side Motion Branch - Left, Right or No Action
    * Rotation Branch - Rotate Left, Rotate Right or No Action
    * Laser Branch - Emit a laser or No Action
- **Observation Space**
  - 53 variables including the velocity of agent, and the ray-based angle information of objects in front of agent (7 raycast angles with 7 measurement for each angle)

2.4 Crawler

Crawler scenario is a modified scenario which originally to let agent learn to walk from Unity. We modify the logic to let agent learn to fight to compete with each other during the training progress. The agent is required to learn to maintain their body not touching the ground, and walk to opponent's position. In addition, the agent is required to learn to fight down the opponent's agent to make the challenger loss balance to get reward.

Figure 4: Illustration of Crawler in Hierarchical Structure

- **Objective**
  - Agent must learn to maintain their body not touching the ground, and fight with opponent agent to make the challenger loss balance.
- **Reward**
  - +1 if opponent's body touch the ground
  - -1 if agent's body touch the ground
  - +0.03 times body velocity towards to opponent's direction
  - +0.01 times body direction alignment with opponent's direction
- **Action Space**
  - 20 variables corresponding to the rotations of joints
- **Observation Space**
  - 117 variables corresponding to position, rotation, velocity, and angular velocities of each limb plus the acceleration and angular acceleration of the body.

3 CONCLUSION

The above experiments show that the ~~hierarchical critics~~ could be applied into Unity ML-Agent's examples to train the agents under reinforcement learning. There are still many settings and adjustments could be made to further improve the training performance in the multi-agent environment.

For future works, this hierarchical critics could further expand into multiple level of critics to observe different observation spaces of the environment. ~~It could also further enhance to have an auto scaling on critics in different level based on the current training performance. There are different possibilities could be adjusted within this area with the hierarchical critics structure.~~

REFERENCES

[1] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. Openai gym. *arXiv preprint arXiv:1606.01540* (2016).
[2] Zehong Cao and Chin-Teng Lin. 2019. Hierarchical Critics Assignment for Multi-agent Reinforcement Learning. *arXiv preprint arXiv:1902.03079* (2019).
[3] Arthur Juliani, Vincent-Pierre Berges, Esh Vckay, Yuan Gao, Hunter Henry, Marwan Mattar, and Danny Lange. 2018. Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627* (2018).
[4] Gonçalo Neto. 2005. From single-agent to multi-agent reinforcement learning: Foundational concepts and methods. *Learning theory course* (2005).
[5] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).
[6] Karl Tuyls and Gerhard Weiss. 2012. Multiagent learning: Basics, challenges, and prospects. *Ai Magazine* 33, 3 (2012), 41–41.