Empirical Analysis of Reinforcement Learning Algorithm via OpenAI gym environments

Donggyu Lee
20170783
KAIST
qwqw3535@kaist.ac.kr

Sungwon Yang
20170817
KAIST
yyang3314@kaist.ac.kr

Yourim Shin
20170361
KAIST
shinyourim@kaist.ac.kr

Abstract

This empirical study compares and analyzes the performances of Deep Q-Network (DQN), Advantage Actor-Critic (A2C), and Proximal Policy Optimization (PPO) algorithms in a combination of Atari game environments and classic control environments. Agents trained with each algorithm were exposed to diverse tasks, learning from pixel inputs in Atari games and state-based observations in classic control tasks. Performance evaluation encompassed learning speed, sample efficiency, stability, and task-solving capabilities. DQN excelled in certain Atari games with its replay buffer, while A2C's learning was unstable and often unsuccessful. PPO demonstrated competitive performance across both environments due to its stability and sample efficiency. The analysis highlights algorithmic strengths and weaknesses, such as A2C's potential variance issues, and the problem of convergence to local optimum by on-policy algorithms. This research contributes to understanding the performance characteristics of DQN, A2C, and PPO, aiding algorithm selection and advancements in reinforcement learning for complex real-world problems.

1 Introduction

Reinforcement learning (RL) algorithms have gained significant attention for their ability to learn optimal policies through trial and error. This empirical study focuses on comparing and analyzing the performances of three widely used RL algorithms: Deep Q-Network (DQN), Advantage Actor-Critic (A2C), and Proximal Policy Optimization (PPO). The evaluation takes place in a combination of Atari game environments and classic control environments.

In recent years, RL algorithms have been successfully applied to various domains, including game playing. Atari game environments provide diverse and challenging scenarios, while classic control environments offer a different set of tasks with distinct characteristics. Understanding the performance of RL algorithms in both these environments is crucial for assessing their suitability across different problem domains.

Agents trained with DQN, A2C, and PPO algorithms were exposed to a range of tasks, learning directly from pixel inputs in Atari games and utilizing state-based observations in classic control tasks. The evaluation criteria encompassed learning speed, sample efficiency, stability, and task-solving capabilities.

The results of the study reveal notable differences in the performance of the algorithms. DQN showcased exceptional performance in specific Atari games, leveraging its replay buffer mechanism to achieve superior outcomes. Conversely, A2C exhibited unstable learning and limited success, suggesting potential issues with its variance and optimization process. PPO demonstrated competitive performance across both Atari and classic control environments, benefiting from its stability and sample efficiency.

By identifying algorithmic strengths and weaknesses, this study provides valuable insights for algorithm selection and the advancement of RL methods. Notably, the findings shed light on the potential variance issues associated with A2C and the challenges of convergence to local optima faced by on-policy algorithms. Such understanding contributes to the development of more effective RL approaches for addressing complex real-world problems.

2 Related Works

2.1 Reinforcement Learning algorithms

The three algorithms we are focusing on—DQN, PPO, and A2C—have emerged as prominent contenders in the field. DQN, introduced by Mnih et al.[1], utilizes a deep neural network to approximate the action-value function and has shown promising results in Atari game environments. PPO, proposed by Schulman et al. [2], employs a policy gradient optimization approach with a surrogate objective to ensure stability and sample efficiency. A2C, introduced by Mnih et al. [3], combines the advantages of actor-critic methods with synchronous parallel training, making it a popular choice for RL tasks.

2.2 Atari games and Gym environment

Atari games, a classic arcade game which was popular in 1970s are widely used for the comparison and evaluation of the new reinforcement learning algorithms because of its simple structure from 2013.[4] For example, Google's one of the famous model and successor of AlphaGo, MuZero[5] showed its performance through Atari games. Also, paper using World model had noticeable result in solving the Atari games. [6] OpenAI Gym environment[7] offers All of the atari games and other popular environments which are used for RL algorithms such as MuJoCo[8] and Classic control.

3 Methods

Advantage Actor-Critic (A2C)[3], Deep Q-Network (DQN)[1], and Proximal Policy Optimization (PPO)[2] algorithms were trained with 3 random seeds and 5 million steps in various OpenAI Gym environments.

3.1 Environments

3.1.1 Atari Games

For Atari games, Atari v4 without frame skip was used. For the input of the neural networks, four frames of observations were stacked together to include the information about the velocity of moving objects in the scene.

3.1.2 Classic Controls

Classic Controls have 5 environments. They are simple tasks which state, action and rewards are neatly defined for the reinforcement learning algorithms can easily use. In contrast to the Atari games, the input of the classic controls are float arrays, not the result of the CNN layer. Three of the games (Acrobot, CartPole, MountainCar) have discrete action space and all algorithms were applicable, while two of the games (MountainCarContinuous, Pendulum) have continuous action space and thus DQN is not applicable for these two environments.

3.2 Experimental Setups

3.2.1 Evaluation of Algorithms

For the evaluation of algorithms, we focused on 4 aspect, which are final result of model, learning speed, learning stability, and variance.

3.2.2 Implementation Details

Stable-Baselines3, a Python reinforcement learning library, was used to help the training process. For all algorithms and environments, a vectorized environment was used for faster training and less correlation within batches.
For all policy and value networks of A2C, DQN, and PPO, 2 linear layers each with 64 units were used. For Atari games, CNN feature extractor is shared by the policy and value networks like in many other literature. The architecture of the CNN feature extractor module was the same as in the "Nature CNN"[1].
The exploration in DQN was done using epsilon-greedy policy and that in A2C and PPO was done by action noise, after which the action was clipped to prevent out of bound error.
In this experiment, 4 environments were run in parallel. All of the Atari games were trained with A100 and V100 GPUs, and the classical controls were trained with CPU or T4 GPU.

Table 1: Hyperparameter settings for the RL algorithms

hyperparameters	**A2C**	**DQN**	**PPO**
discount factor	0.99	0.99	0.99
max grad norm	0.5	0.5	0.5
learning rate	7×10^{-4}	10^{-4}	3×10^{-4}
replay/rollout buffer size		10^6	8192
batch size	20	32	64
exploration ratio		1.0 - 0.05	
clip range			0.2

4 Results

4.1 Result of Atari games

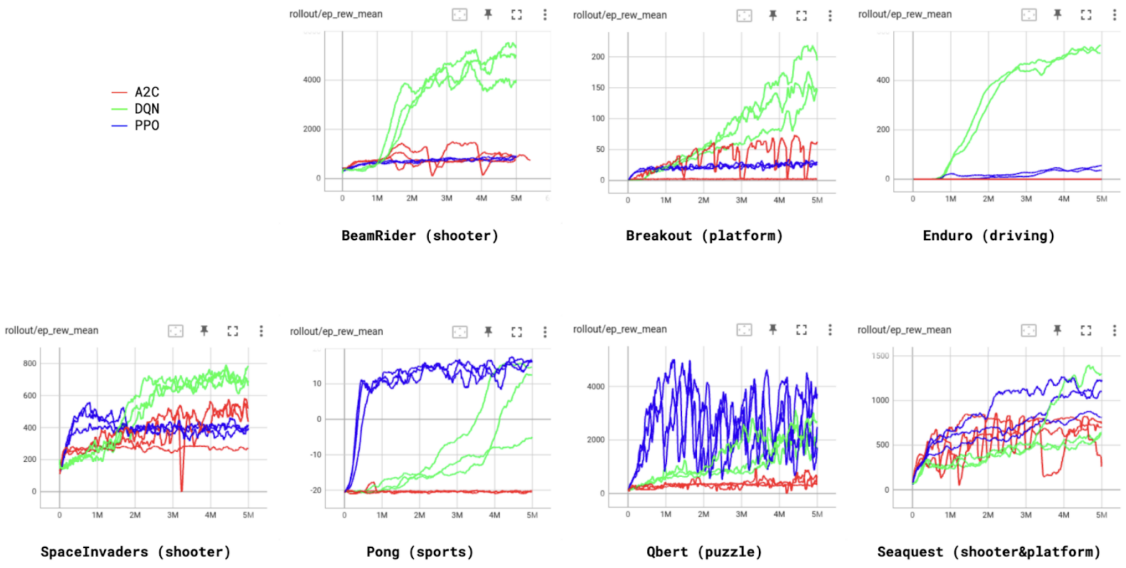

Figure 1: The learning graph of Atari games.

4.2 Analysis of Atari games

Convergence in the early stages of training was observed in Atari game results, often led to a local optimum, where the algorithms reached suboptimal policies. Specifically, PPO demonstrated quick learning of local optimal policies and converged, while DQN continued to improve and eventually outperformed PPO.

Table 2: Mean and standard deviation of 3 repeated Atari Games

mean±SD	A2C	DQN	PPO
BeamRider	797.0±73.5	4791.2±697.0	843.5±87.9
Breakout	22.5±34.9	162.1±27.9	28.1±1.7
Enduro	0.0±0.0	532.9±19.9	30.3±27.9
SpaceInvaders	410.2±128.8	709.0±67.2	407.6±10.3
Pong	-20.3±0.2	7.5±10.9	16.4±0.1
Qbert	558.5±204.9	2212.2±484.4	2640.6±1009.1
Seaquest	568.5±267.4	856.1±382.6	1045.3±217.3

DQN demonstrated superior performance in games with dynamic scenes, leveraging its replay buffer to handle rare observations effectively. PPO, on the other hand, excelled in games with more static scenes, exhibiting quick convergence to local optimal policies. However, it was DQN that ultimately outperformed PPO over time due to its continuous improvement and exploration capabilities. This observation highlights the advantage of off-policy algorithms in exploration compared to on-policy algorithms. Moreover, DQN's replay buffer makes it deal better with rare observations while on policy algorithms may underfit to rare observations. In addition, Atari environments don't require probabilistic policies unlike multi-agent environments where deterministic policies can be exploited by other agents (ex. rock-scissors-paper). Hence, DQN performed exceptionally well in these contexts.

The instability of A2C was another notable observation. A2C exhibited large fluctuations in the training curve, indicating its sensitivity to hyperparameters and random seeds. Possible explanations for A2C's learning failure revolved around two sources of error: the value function (V) and the policy (pi). We found that clipping changes in policy, as implemented in PPO, worked better than directly clipping the gradient itself. These factors made it challenging to achieve consistent performance with A2C.

4.3 Result of Classic controls

Figure 2: The learning graph of Classic control

Table 3: Mean and standard deviation of 3 repeated Classic control

mean±SD	A2C	DQN	PPO
Acrobot	797.0±73.5	4791.2±697.0	843.5±87.9
CartPole	22.5±34.9	162.1±27.9	28.1±1.7
MountainCar	0.0±0.0	532.9±19.9	30.3±27.9
MountainCarCont	410.2±128.8	709.0±67.2	407.6±10.3
Pendulum	-20.3±0.2	7.5±10.9	16.4±0.1

4.4 Analysis of Classic control

In classic control, PPO emerged as the top performer. It demonstrated superior results, better stability, and significantly faster convergence speed compared to the other algorithms. However, in the MountainCar and MountainCarContinuous environments, both PPO and A2C failed to achieve the goal and remained near the starting point, while DQN continuously explored the environment. A2C and PPO tended to stay close to the ground, adopting a cautious approach to minimize penalties.

In contrast, DQN displayed an aggressive exploration strategy, continuously attempting to navigate the environment and reach the goal. This also suggests that PPO and A2C might have fallen into a local optimum, while DQN adopted a more exploratory strategy.

4.5 Advantages of Deep Q Network

While it is difficult to definitively claim that one algorithm is universally "best" among the others, DQN has demonstrated superior performance in several cases. Here are a few possible explanations why DQN has outperformed PPO and A2C in some gym environments:

Experience Replay DQN utilizes experience replay, which involves storing transitions (state, action, reward, next state) in a replay buffer. This buffer allows the agent to break the temporal correlation of the data and sample from a more diverse set of experiences during training. Experience replay has been found to stabilize the learning process, prevent catastrophic forgetting, and improve sample efficiency.

Q-Learning and Value Iteration DQN is based on Q-learning, a model-free method that estimates the optimal action-value function (Q-function). By iteratively updating the Q-function based on the Bellman equation, DQN aims to find the optimal policy. This approach is particularly effective in environments with discrete actions, such as Atari games.

Exploration-Exploitation Trade-off DQN balances exploration and exploitation through the use of an epsilon-greedy policy, gradually decreasing the exploration rate over time. This allows the agent to initially explore the environment and gradually exploit the learned policy as training progresses.

Transfer Learning and Generalization DQN's architecture and learning approach enable transfer learning between different games. By pretraining on a set of games and fine-tuning on a target game, DQN can leverage learned knowledge and generalize across similar game environments more effectively.

5 Conclusion

In conclusion, our analysis of deep RL algorithms highlights the absence of a universally superior algorithm across all environments. Each algorithm possesses its own strengths and weaknesses, and their performances can vary based on the complexity and dynamics of the environment. Furthermore, these algorithms exhibit differences not only in their final performance but also in their robustness to hyperparameters and randomness. To select the most suitable algorithm for a given situation, it is crucial to consider various factors, including sample efficiency.

The limitations of our study include the need for further exploration of hyperparameter settings and the requirement for more extensive data collection, especially in games with varying complexity. Overall, this analysis contributes valuable insights into the performance and characteristics of deep RL algorithms, aiding in their application, development and choosing appropriate reinforcement learning algorithm in new environment.

6 Contribution

- Donggyu Lee(DL), Sungwon Yang(SY), Yourim Shin(YS)

All authors conceived and planned the experiments and contributed to the interpretation of the results. DL and SY carried out the experiments. All authors provided critical feedback and helped shape the research, analysis and manuscript. All authors discussed the results and contributed to the final manuscript.

References

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen,

Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015.

[2] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[3] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning, 2016.

[4] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, jun 2013.

[5] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, dec 2020.

[6] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models, 2022.

[7] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[8] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.