Adaptive Trust Criteria Clipping in Enhanced PPO Algorithm for Ms. Pacman Learning

Shaozhen Pan[1*], Xinhan Qiu[2*], Yusi Gong[3*], Zhuoyue Tan[4*], Shu Wang[5*]

[1]36920231153226 class for Institute of AI
[2]36920231153229 class for Institute of AI
[3]23020231154183 class for Institute of Information
[4]36920231153232 class for Institute of AI
[5]36920231153236 class for Institute of AI

Abstract

Reinforcement learning algorithms have demonstrated significant potential in addressing complex decision-making tasks. However, effective exploration of the action space remains a challenge, particularly for the Proximal Policy Optimization (PPO) algorithm, which can lead to suboptimal performance and convergence issues. In this study, we introduce an adaptive regional trust criterion shearing mechanism to improve the exploration-exploitation balance of the original PPO algorithm. By dynamically adjusting the clipping range within the trust region, our method aims to enhance the algorithm's ability to explore and exploit action spaces effectively. Experimental evaluations, including comparative analyses with the DQN and SAC algorithms, as well as the original PPO algorithm, demonstrate the efficacy of our approach in learning the Ms. Pacman game in the Atari environment. The results indicate that our optimized PPO algorithm achieves improved performance and sampling efficiency, highlighting its potential for addressing the exploration challenges inherent in traditional PPO algorithms.

Introduction

Deep learning models have demonstrated remarkable advancements, attaining state-of-the-art outcomes in fields such as vision (Krizhevsky, Sutskever, and Hinton 2012; Szegedy et al. 2015; He et al. 2016) and speech recognition (Amodei et al. 2016). This success can be attributed to the proficiency of models like convolutional neural networks in deriving high-level features from extensive datasets. Simultaneously, reinforcement learning has also made significant strides, particularly in addressing its historical challenges by enabling a more adaptable state representation. Deep reinforcement learning (DRL) has found widespread application in developing agents capable of mastering complex control tasks within computer games. Notable achievements including an architectural proposition for solving 3D first-person shooter game environments using deep learning (Lample and Chaplot 2017) , an introduction of the "Neural Fictional Self-Game" approach for learning approximate Nash equilibrium strategies (Heinrich and Silver 2016), and a groundbreaking work in generalizing the AlphaGo Zero program to achieve superhuman performance spanning various domains (Silver et al. 2017).

The Atari 2600 platform, popularly known as Atari games, was home to several hundred video games developed during the 1980s. As a pioneer and substantial influence in the early years of the video game industry, it introduced a wide array of game genres, ranging from arcade-style action, adventure, racing, to puzzle games. These games present diverse and increasingly intricate environments, thereby offering a plethora of challenges that steer the formative development of Artificial Intelligence (AI), specifically in the field of reinforcement learning. And today, Atari games have emerged as a standard benchmark for the evaluation of reinforcement learning algorithms.

In recent times, remarkable strides in AI and Machine Learning (ML) have yielded astounding outcomes, surpassing human performance in diverse domains, such as ATARI games. Notably, Ziyu et al. introduced an innovative reinforcement learning approach, leading to substantial enhancements in policy evaluation and performance within the renowned Atari 2600 game, MsPacman (Mnih et al. 2015). Additionally, Ding et al. introduced the Deep Spiking Q-Network (DSQN), a more resilient and efficient reinforcement learning approach that demonstrates exceptional proficiency in ATARI games (Van Hasselt, Guez, and Silver 2016). These achievements have sparked widespread interest in AI and neural networks, particularly in the realm of Deep Reinforcement Learning (RL). However, Poor initialization can cause Proximal Policy Optimization (PPO) to suffer from a lack of exploration, increasing the risk of training failure or entrapment in suboptimal local optima. This limitation can hinder the algorithm's performance, particularly in complex and high-dimensional environments where finding optimal policies becomes increasingly challenging. Therefore, there is a critical need to develop innovative methods that can dynamically adjust the exploration-exploitation trade-off within the trust region to improve the algorithm's ability to navigate and learn from complex action spaces.

In this work, we explore a novel approach, in which we enhances the original PPO algorithm by employing an adaptive region trust criteria clipping mechanism, which adap-

*These authors contributed equally.

tively adjusts the clipping range based on the trust region. We also conducted a series of comparative experiments, involving the application of the DQN algorithm, the PPO algorithm, the SAC algorithm, and our optimized PPO algorithm to learn the Ms. Pacman game in the Atari environment. Our contributions are as follows

- we proposed a novel enhancement proposed for the PPO algorithm. The experimental findings demonstrate that this unique improvement strategy exhibits slightly superior performance compared to the conventional PPO method.
- We conduct a comprehensive comparison of three reinforcement learning algorithms, namely DQN, PPO, and SAC, in application to the Atari game, MsPacman. The experimental results illustrate that amongst these, PPO not only demonstrated superior performance, but also achieved the result in the shortest time.

Related Work

Game-playing algorithms have a rich history, with some even surpassing human players. In this section, we will delve into related works that explore self-playing algorithms. We categorize these works into three main areas: non-deep learning methods, deep reinforcement learning methods, and multi-agent deep reinforcement learning methods.

Non-deep learning

Early research in this area mainly used some non-deep learning methods. Wender et al. (2012) evaluated the suitability of reinforcement learning (RL) algorithms for performing combat unit micromanagement tasks in the commercial real-time strategy (RTS) game StarCraft: Broder War. After that, Crespo Wichert (2020) presented a comprehensive review on reinforcement learning applied to games, discussing both classic and recent developments. They highlighted the importance of self-play, where the algorithm learns by playing against itself without requiring any direct supervision. In another study, Bai Jin (2020) introduced a self-play algorithm,Value Iteration with Upper/Lower Confidence Bound (VI-ULCB),and showed that it achieves regret after playing T steps of the game. This work presents the first line of provably sample-efficient self-play algorithms for competitive reinforcement learning. These studies demonstrate the effectiveness of non-deep learning methods in game applications, providing a strong foundation for further research in this area.

Deep Reinforcement Learning

Given the intricate nature and inherent challenges of multiplayer games, multi-agent reinforcement learning has surfaced as a potential solution. A landmark development in this field was the advent of the Deep Q-Network (DQN) algorithm, pioneered by Mnih et al. (2013). This marked the inaugural successful fusion of deep learning and reinforcement learning, specifically Q-learning, culminating in a comprehensive learning system. Building upon the foundational work of then, Ivan Sorokin et al.(2015) have applied the DQN algorithm to a variety of gaming contexts with notable success. For instance, in the realm of Atari games, DQN has been utilized to approximate state-value functions, yielding impressive results such as achieving human-like accuracy after approximately 110 games and superior performance after 300 games. This application of DQN demonstrates its potential in tackling complex, high-dimensional problems within the gaming sphere. Another significant contribution to this field is the Proximal Policy Optimization (PPO) algorithm, proposed by Schulman et al. (2017). This algorithm represents a novel category of policy gradient methods that alternate between data sampling through environmental interaction and the optimization of a "surrogate" objective function via stochastic gradient ascent. In the realm of game-playing applications, PPO algorithm has been successfully implemented by Chao Yu et al.(2018). They demonstrated that PPO-based multi-agent algorithms achieve surprisingly strong performance in four popular multi-agent testbeds: the particle-world environments, the StarCraft multi-agent challenge, Google Research Football, and the Hanabi challenge, with minimal hyperparameter tuning and without any domain-specific algorithmic modifications or architectures. Importantly, compared to competitive off-policy methods, PPO often achieves competitive or superior results in both final returns and sample efficiency. This application of PPO demonstrates its potential in tackling complex, high-dimensional problems within the gaming sphere. The Soft Actor-Critic (SAC) algorithm proposed by Haarnoja et al. (2019), introduces an innovative approach by striving to maximize not only the expected return but also the policy's entropy. This implies that the algorithm aims to accomplish the task while maintaining as much randomness in its actions as possible.

Multi-agent Deep Reinforcement Learning

Given the complexities and challenges of multiplayer games, multi-agent reinforcement learning has emerged as a promising approach. Lowe, Ryan, et al. (2017) proposed an adaptation of the actor-critic method that takes into account the action strategies of other agents and it is able to successfully learn strategies that require complex multi-agent coordination. By defeating the Dota 2 world champions (Team OG) , OpenAI Five proved that self-playing reinforcement learning can achieve superhuman performance on difficult tasks (BERNERC et al. 2019) . Then Brown et al. (2019) introduced an artificial intelligence system named 'Pluribus', which is capable of outperforming top human players in six-player no-limit Texas Hold'em poker. In the same year, Vinyals et al. (2019) developed AlphaStar, a multi-agent reinforcement learning algorithm that leverages data from human and agent games, achieving a Grandmaster level of play for all three StarCraft races and surpassing over 99.8% of officially ranked human players.

Our Work

Model

First, the state of each step of the game is an 84*84 RGB image. After a series of preprocessing, an 84*84 grayscale

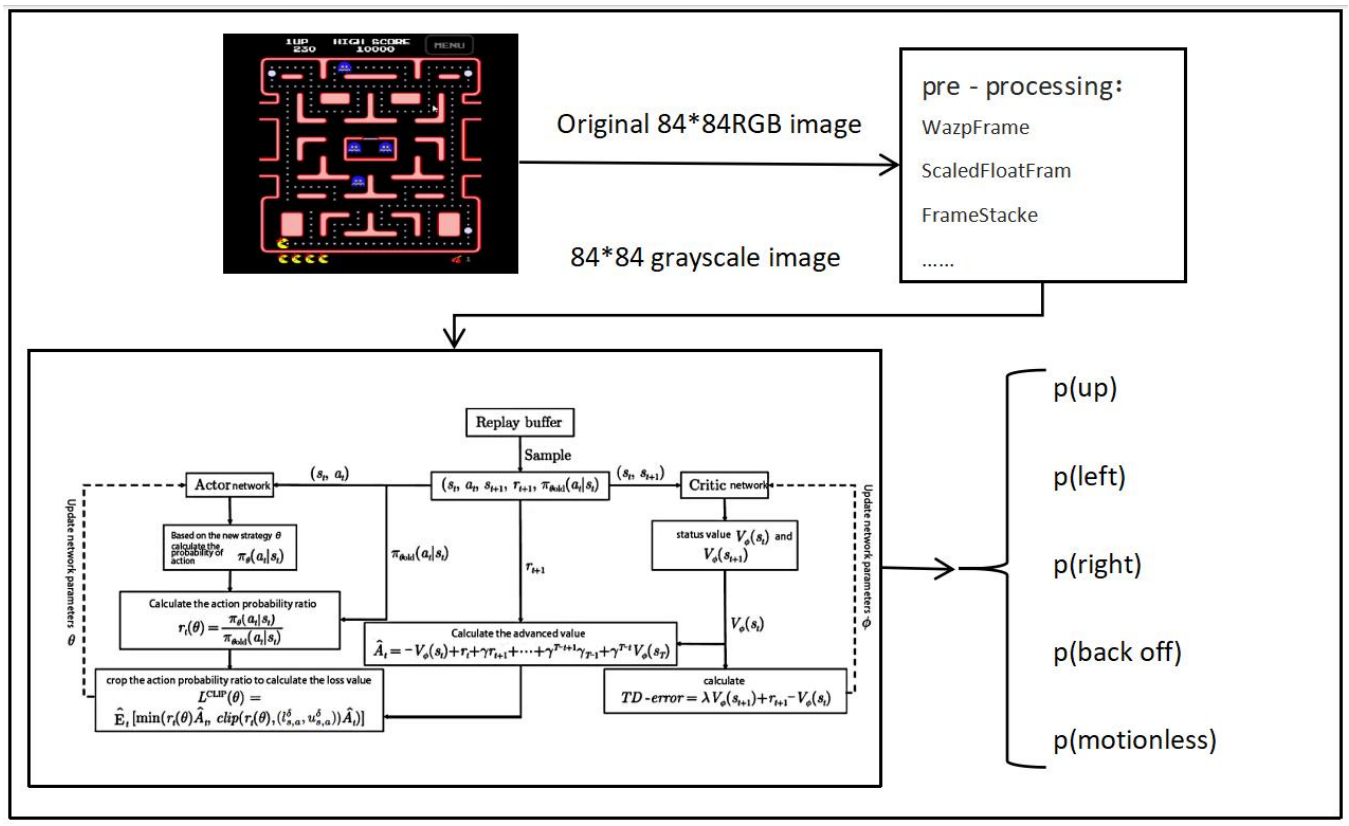

Figure 1: After each frame of image is pre-processed and input into the decision-making network, the probability distribution of the next action in a certain state is obtained.

image is obtained and input into the network. The advantage function of the state and the action is calculated through critic and calculated through actor. The probabilities of each action of the agent are obtained, and random sampling is performed based on these probabilities to obtain the next action. The actor parameters are updated based on the ratio of the probabilities of the new and old strategies and the calculated advantage function, and the probability ratio is adaptively tailored according to our algorithm. Then the parameters are updated according to gradient ascent.

Previous Algorithm

PG The policy gradient algorithm is an on-policy algorithm. The limited trajectory obtained by interacting with the environment through the policy function uses gradient ascent to update the policy function. The most commonly used gradient estimator has the following form:

$$\hat{g} = \hat{E}_t \left[\nabla_\theta \log \pi_\theta \left(a_t \mid s_t \right) \hat{A}_t \right] \tag{1}$$

Among them, $\hat{A}_t$ represents the advantage function, and the specific form is as follows. It is expected that E_t represents the average value of limited batch sample experience.

$$A^\theta \left(s_t, a_t \right) = \sum_{t'>t} \gamma^{t'-t} r_{t'} - V_\phi \left(s_t \right) \tag{2}$$

Among them, $V_\phi(st)$ is calculated by critic. It consists of a neural network with the same structure as the policy network but different parameters. It is mainly used to fit the discount reward from s_t to the end game. The first half of A^θ is the actual sampling discount reward, and the second half is the fitted discount reward. A^θ represents the advantage of the actual discount reward obtained by taking action a_t under s_t compared to the simulated discount reward. This advantage function is given by a critic (evaluator). The final corresponding objective function is:

$$L^{PG}(\theta) = \hat{E}_t \left[\log \pi_\theta \left(a_t \mid s_t \right) \hat{A}_t \right] \tag{3}$$

A big disadvantage of the PG method is that the parameter update is slow, because every time we update the parameters, we need to resample, and our sample can only be used once. This is actually an on-policy strategy, that is, the agent we want to train. It is the same agent as the agent that interacts with the environment; corresponding to it is the off-policy strategy, that is, after we use the sample to update it once, we no longer discard it, but reuse the old data to update it multiple times.

PPO-Penalty The biggest improvement of TRPO over PG is the implementation of off-policy, but it is obviously unreasonable to just repeat learning on old data, which will lead to destructive updates of policy parameters. For this purpose,

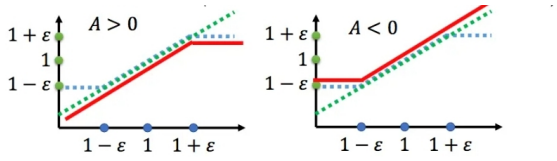

Figure 2: The green line represents the first item in min, that is, no processing is performed, and the blue line represents the second item, which is the clipping item. If the gap between the two distributions is too large, a certain degree of clipping will be performed. The final red line is to take min from these two items.

the objective function of importance sampling is introduced as follows:

$$\underset{\theta}{\text{maximize}} \hat{E}_t \left[\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)} \hat{A}_t \right] \tag{4}$$

$$\text{subject to } \hat{E}_t \left[\mathrm{KL}\left[\pi_{\theta_{\text{old}}}\left(\cdot \mid s_t\right), \pi_\theta\left(\cdot \mid s_t\right) \right] \right] \leq \delta. \tag{5}$$

$\theta_(old)$ refers to the parameters before updating the strategy. $\theta_(old)$ and θ cannot be too far apart, because too far a difference will introduce fallacies, so we need to use KL divergence to constrain the difference between the two. Distribution bias. The theory that proves TRPO actually suggests using penalties rather than constraints to solve unconstrained optimization problems, which is what follows

$$\max_\theta \hat{E}_t \left[\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)} \hat{A}_t - \beta \mathrm{KL}\left[\pi_{\theta_{\text{old}}}\left(\cdot \mid s_t\right), \pi_\theta\left(\cdot \mid s_t\right) \right] \right] \tag{6}$$

TRPO uses hard constraints instead of penalties because it is difficult to choose a value β that performs well in different problems, or even within a problem when the learned features vary. Experiments show that simply selecting a fixed penalty coefficient β and using SGD to optimize the penalty objective equation is not enough; β needs to be continuously adjusted during the update process.

PPO-clip The definition is as follows:

$$r_t(\theta) = \frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_{\text{old}}}\left(a_t \mid s_t\right)} \tag{7}$$

$$L^{CLIP}(\theta) = \hat{E}_t \Big[\min(r_t(\theta)\hat{A}_t, \mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t) \Big] \tag{8}$$

We now consider modifying the target to penalize changes that move the ratio away from 1. The first term within the minimum is the original value. The second term limits rt to (1-ϵ) and (1+ϵ) through clipping, thus eliminating the possibility of r_t running out of this interval. Finally, the minimum value of the clipped target and the uncut target is taken.

Adaptive shearing mechanism

We give an in-depth analysis of the exploration behavior of PPO and show that PPO is susceptible to the risk of lack of exploration especially in the case of poor initialization, which can lead to failure of training or getting stuck in poor local optima. To address these issues, we propose a new policy optimization method that adaptively adjusts the clipping range within the trust region. If current policy does not favor that action, then PPO's ratio-based measure will tend to continually weaken the likelihood that that action will be chosen in the future. Therefore, PPO is prone to the risk of lack of exploration, especially in the case of poor initialization, which may lead to training failure or falling into a bad local optimum. To address these issues, we propose an enhanced PPO method, whose improved exploration capabilities and better performance constraints are theoretically justified compared to the original PPO. This approach builds a connection between ratio-based metrics and trust-region-based metrics such that the resulting ratio clipping mechanism allows relaxing the constraints imposed on less preferred operations. This effectively encourages policy to explore more potentially valuable actions, whether or not they were favored by previous policies. At the same time, the range of the new ratio-based constraints remains within the trust region; therefore, it does not harm the stability of learning. Extensive results on several benchmark tasks show that this method significantly improves policy performance and sampling efficiency. We define a new clipping range ($l^\delta_{s,a}$,$u^\delta_{s,a}$), a is a hyperparameter

$$\begin{aligned} l^\delta_{s,a} &= \min_\pi \left\{ \frac{\pi(a|s)}{\pi_{\text{old}}(a|s)} : D^s_{\text{KL}}(\pi_{\text{old}}, \pi) \leq \delta \right\} \\ u^\delta_{s,a} &= \max_\pi \left\{ \frac{\pi(a|s)}{\pi_{\text{old}}(a|s)} : D^s_{\text{KL}}(\pi_{\text{old}}, \pi) \leq \delta \right\} \end{aligned} \tag{9}$$

A PPO metric with a constant shear range can cause exploration problems because it imposes relatively tight constraints on actions that older strategies dislike. Therefore, we hope to be able to relax this constraint by increasing the upper clipping range and decreasing the lower clipping range when the probability of the old policy decreases. Figure 2 shows the shear range of PPO2 and PPO. For PPO2 (blue curve), as π_{old} gets smaller, the upper shear range increases, while the lower shear range decreases, which means that as π_{old} gets smaller, The restrictions are relatively loose. This mechanism can encourage agents to explore more potentially valuable actions that were not favored by the old policy.

We now propose how to efficiently compute adaptive clipping ranges. For the discrete action space, using the KKT condition, it is transformed into solving the following equation

$$\begin{aligned} g(\pi_{\text{old}}(a|s), X)(1 - \pi_{\text{old}}(a|s)) \log \frac{1 - \pi_{\text{old}}(a|s)}{1 - \pi_{\text{old}}(a|s)X} \\ -\pi_{\text{old}}(a|s) \log X = \delta \end{aligned} \tag{10}$$

By solving the above equation, we can get ($l^\delta_{s,a}$,$u^\delta_{s,a}$)

Experiments

In this section, we present the experimental evaluation of several reinforcement learning algorithms, including Soft Actor-Critic (SAC), Proximal Policy Optimization (PPO), Deep Q-Network (DQN), and our proposed algorithm. The

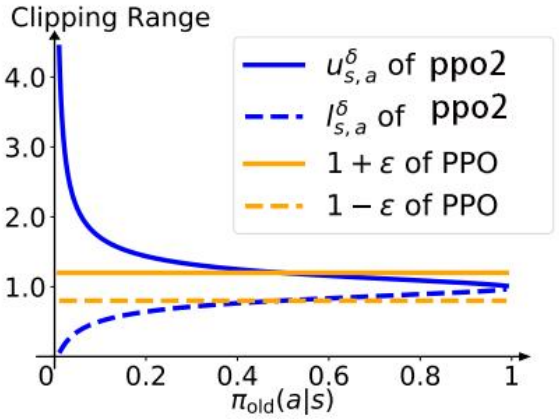

Figure 3: Comparison of shear ranges of ppo and ppo2 as π_{old} changes

experiments were conducted in the Atari game environment, aiming to assess the performance and effectiveness of these algorithms in challenging and dynamic environments.

Experiments Settings

Hardware Configuration Our experimental hardware environment consists of a Linux workstation equipped with an RTX 4060 graphics card boasting 16GB of dedicated memory. This configuration provides us with the necessary computational power and resources for training and evaluating reinforcement learning algorithms.

Experimental Platform For our research, we have chosen Tianshou as the experimental platform. Tianshou is a reinforcement learning platform that is built upon pure PyTorch. It offers a modularized framework and a Pythonic API, enabling the development of deep reinforcement learning agents with minimal lines of code. Tianshou has been widely recognized as a highly suitable platform for algorithm comparison and evaluation in the field of reinforcement learning.

Experimental parameter setting To ensure a fair comparison, we have carefully selected and fine-tuned the parameters of each algorithm based on existing literature and empirical observations. After training each algorithm for ten million steps in our custom game environment, we conducted a comprehensive evaluation to compare their performance. The key parameters for each algorithm are as follows:

For all the algorithms, we maintained a buffer size of 100,000, a discount factor (gamma) of 0.99, and conducted 100 epochs with each epoch consisting of 100,000 steps. Additionally, we tailored specific parameters for each algorithm.

- For the Proximal Policy Optimization (PPO) algorithm and our proposed algorithm, we set the learning rate to 2.5e-4 and the batch size to 256.
- Regarding the Soft Actor-Critic (SAC) algorithm, we assigned a learning rate of 1e-5 for both the actor and critic networks, and utilized a batch size of 64.
- For the Deep Q-Network (DQN) algorithm, we employed a learning rate of 1e-4 and a batch size of 32.

Experiments results

Our experimental evaluation reveals significant insights into the performance of the studied reinforcement learning algorithms. The experiments were conducted over 10,000,000 training steps, and the algorithms were evaluated based on their training and test rewards, as well as loss trajectories.

Training and Test Rewards The train reward curve shown as Fig.4 indicates that SAC and our proposed algorithm consistently outperformed DQN and PPO in terms of cumulative rewards throughout the training process. Notably, SAC demonstrated a rapid ascent in rewards, suggesting a swift policy improvement over initial steps. Our algorithm exhibited competitive performance, closely following SAC and surpassing PPO and DQN. In the test environment, as can be seen in Fig.5 the algorithms' rankings remained consistent with the training phase, with SAC achieving the highest rewards, followed by our algorithm, PPO, and DQN, respectively.

Figure 4: Training Reward Progression: Reward progression during training, showing SAC and our algorithm outperforming PPO and DQN.

Loss Trajectory The loss curve shown as Fig.6 presents an interesting perspective, with our algorithm manifesting a more stable decline in loss compared to PPO. Initially, both algorithms experienced fluctuations; however, as training progressed, our method showed a steadier and more pronounced decrease in loss values, indicating a more stable learning process.

Computational Efficiency In terms of computational efficiency, our proposed PPO variant demonstrated a significant reduction in training time, completing the 10,000,000 steps in just 2.579 hours, while the standard PPO completed it in 2.355 hours. SAC and DQN exhibited comparable training

Figure 5: Test Reward Performance: Test phase rewards, with SAC leading, followed by our algorithm, PPO, and DQN.

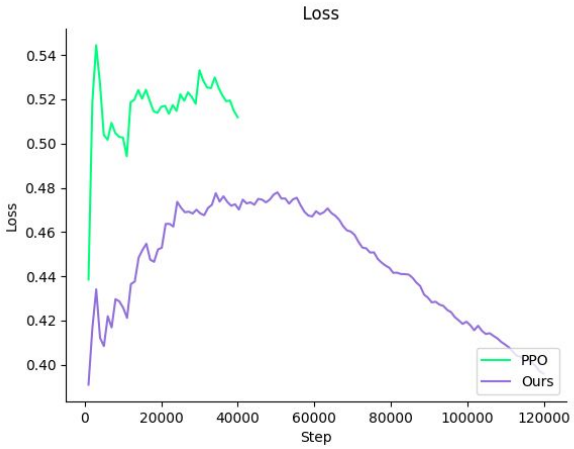

Figure 6: Loss Trajectories:Loss trajectories for PPO and our algorithm, indicating more stable learning for our method.

durations of 6.715 and 6.761 hours, respectively, highlighting the efficiency of our approach.

Summary In summary, SAC led the performance charts in terms of reward acquisition, with our proposed algorithm not far behind, showcasing robust learning capabilities. Our algorithm's efficiency in training time offers a considerable advantage, particularly in resource-constrained scenarios. The lower loss values and faster convergence rate of our algorithm suggest an improved exploration strategy and policy optimization over standard PPO. These results underscore the potential of our enhanced PPO method in challenging and dynamic environments.

Conclusion

This paper uses the adaptive regional trust criterion shearing mechanism to improve the original PPO. This method can adaptively adjust the clipping range within the trust area. We formally show that this approach not only improves the exploration capability within the trust region but also has better performance constraints than the original PPO. Our PPO2 method improves the performance of the original PPO through more exploration and better sample efficiency, and is competitive with several state-of-the-art methods, while maintaining the stable learning performance and simplicity of PPO. We start our work from the impact of the measurement of policy constraints on the exploratory behavior of policy learning. In this sense, our adaptive pruning mechanism is a new alternative that can incorporate prior knowledge to achieve fast and stable policy learning. We hope it will inspire future work to examine more explicit policy indicators to guide effective learning behaviours.

References

Amodei, D.; Ananthanarayanan, S.; Anubhai, R.; Bai, J.; Battenberg, E.; Case, C.; Casper, J.; Catanzaro, B.; Cheng, Q.; Chen, G.; et al. 2016. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, 173–182. PMLR.

Andrade, G.; Ramalho, G.; Santana, H.; and Corruble, V. 2005. Automatic computer game balancing: a reinforcement learning approach. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, 1111–1112.

Berner, C.; Brockman, G.; Chan, B.; Cheung, V.; Debiak, P.; Dennison, C.; Farhi, D.; Fischer, Q.; Hashme, S.; Hesse, C.; et al. 2019. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*.

Brown, N.; and Sandholm, T. 2019. Superhuman AI for multiplayer poker. *Science*, 365(6456): 885–890.

Guo, X.; Singh, S.; Lee, H.; Lewis, R. L.; and Wang, X. 2014. Deep learning for real-time Atari game play using offline Monte-Carlo tree search planning. *Advances in neural information processing systems*, 27.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Heinrich, J.; and Silver, D. 2016. Deep reinforcement learning from self-play in imperfect-information games. *arXiv preprint arXiv:1603.01121*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

Lample, G.; and Chaplot, D. S. 2017. Playing FPS games with deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Lowe, R.; Wu, Y. I.; Tamar, A.; Harb, J.; Pieter Abbeel, O.; and Mordatch, I. 2017. Multi-agent actor-critic for mixed

cooperative-competitive environments. *Advances in neural information processing systems*, 30.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533.

Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; et al. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*.

Smith, M.; Lee-Urban, S.; and Muñoz-Avila, H. 2007. RETALIATE: Learning winning policies in first-person shooter games. In *AAAI*, 1801–1806.

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1–9.

Van Hasselt, H.; Guez, A.; and Silver, D. 2016. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354.

Wender, S.; and Watson, I. 2012. Applying reinforcement learning to small scale combat in the real-time strategy game StarCraft: Broodwar. In *2012 ieee conference on computational intelligence and games (cig)*, 402–408. IEEE.