Autonomus and Adaptive Systems project

Michele Armillotta

1 Introduction

This project aims to implement a Reinforcement Learning (RL) algorithm for the Overcooked benchmark, with the goal of achieving strong performance in a complex multi-agent environment. The selected algorithm for this task is PPO-Clip (Proximal Policy Optimization with Clipping), chosen for several reasons. Firstly, PPO-Clip is a widely adopted and well-documented algorithm, making it an excellent choice for learning and practically experimenting with state-of-the-art RL techniques. Moreover, it strikes a favorable balance between performance, training stability, and implementation simplicity. Since its introduction in 2017 (1), PPO has consistently outperformed more traditional algorithms (e.g., value-based methods, TRPO) in a variety of challenging environments.

2 Background

2.1 Overcooked

Overcooked is a benchmark designed for evaluating cooperative agents (2), inspired by the popular video game of the same name. The main objective is to prepare and deliver soups as efficiently as possible. Each soup requires the collection and insertion of three ingredients into a pot, waiting for it to cook, and finally delivering the prepared soup. This process yields rewards that agents can leverage to improve their performance.

Despite its playful nature, the Overcooked environment poses a non-trivial coordination challenge. Agents must synchronize their actions effectively to achieve high scores. The benchmark includes multiple environment layouts, which introduce different spatial constraints and strategic requirements. These layouts serve not only to increase the task's difficulty but also to test the agents' ability to generalize their learned policies to previously unseen scenarios.

2.2 PPO-Clip

PPO-Clip is an actor-critic method, meaning it concurrently learns both a policy function and a value function. It provides a more practical and computationally efficient alternative to Trust Region Policy Optimization (TRPO), while maintaining similar theoretical motivations such as achieving stable policy updates that do not deviate excessively from the current policy.

PPO-Clip accomplishes this by modifying the objective function through a clipping mechanism, which constrains the magnitude of policy updates. This prevents the new policy from diverging too far from the old one, thereby stabilizing training. The objective function is defined as follows:

$$\theta_{t+1} = \arg\max_{\theta} \mathbb{E}_{s,a\sim\pi_{\theta_t}} \left[L(s, a, \theta_t, \theta) \right]$$

$$L(s, a, \theta_t, \theta) = \min \left(\frac{\pi_\theta(a|s)}{\pi_{\theta_t}(a|s)} A^{\pi_{\theta_t}}(s, a),\ \text{clip} \left(\frac{\pi_\theta(a|s)}{\pi_{\theta_t}(a|s)}, 1 - \epsilon, 1 + \epsilon \right) A^{\pi_{\theta_t}}(s, a) \right)$$

Using policy approximation methods like PPO in Overcooked-like environments offers several theoretical and practical advantages. PPO's clipped objective function prevents excessively large policy updates that could destabilize training, which is particularly important in multi-agent settings where sudden policy changes can break established coordination patterns. The method maintains a balance between exploration and exploitation through its stochastic policy parameterization, allowing agents to discover diverse cooperative strategies while ensuring sample efficiency. Additionally, PPO's on-policy nature and trust region approach provide stable learning dynamics, which helps preserve beneficial collaborative behaviors discovered during training while still allowing for gradual policy improvement.

3 Implementation Choices and Experiments

To implement PPO-Clip, I explored several variants of the algorithm, which are discussed and compared in this section. While strong performance was already achieved on the basic layout ("Cramped Room") using a straightforward implementation, alternative design choices were also investigated for two main reasons.

First, an undetected bug in the initial implementation led to explore additional approaches, such as reward shaping, shared critics, and clipped critics, under the assumption that the original solution might have been too simplistic for the task. Second, I aimed to find an implementation that could generalize better across more complex and previously unseen layouts, where agents must coordinate effectively under more challenging conditions.

3.1 Baseline PPO Implementation

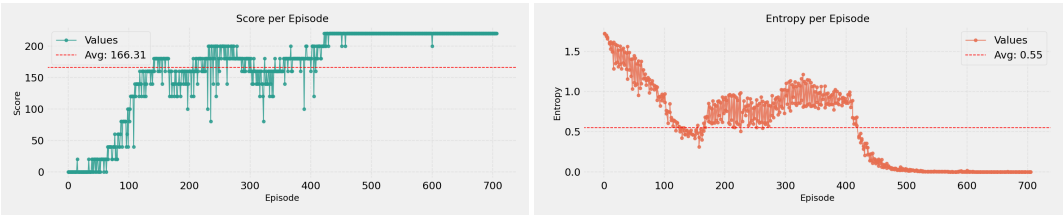
The first implementation uses two simple MLPs for the actor and the critic, a memory buffer to store past transitions and manage minibatches, and a training module that rigorously applies the PPO-Clip update rules. Two identical agents are instantiated, each acting independently and learning its own policy and value function.

However, this version yielded poor results: even after 500 episodes, the average score remained at zero. This failure is likely due to the extremely sparse reward structure of the environment, which provides little guidance for agents to discover useful behaviors through random exploration.

3.2 PPO with Reward Shaping

This experiment extends the baseline by introducing a custom reward shaping function. While the Overcooked environment already provides per-agent rewards, I further shaped them to encourage cooperation: agents receive small rewards for individual actions (e.g., picking up an onion or a plate), and larger ones for cooperative achievements (e.g., cooking a soup).

With this reward structure, agents learn effective policies in relatively few episodes on the "Cramped Room" layout. The policy quickly converges to a highly deterministic behavior that achieves an average score of 220.



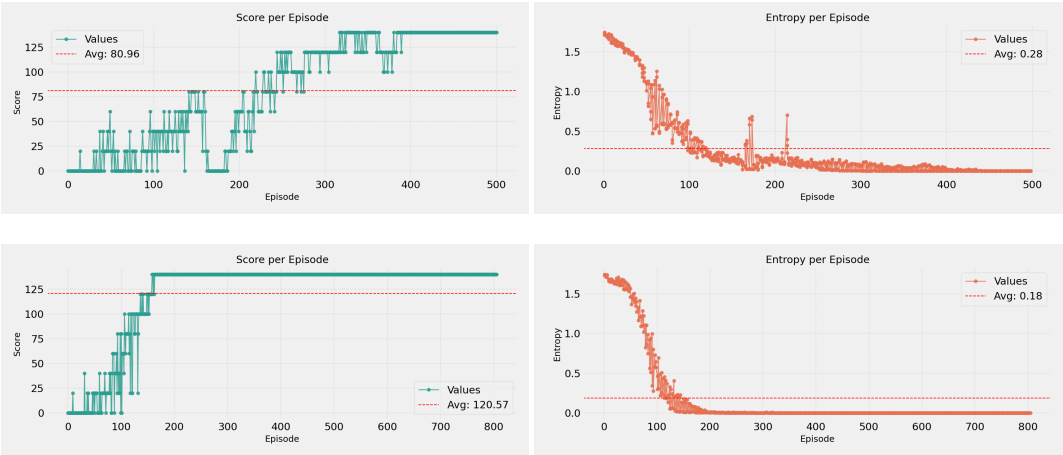
3.3 PPO with Shared and Clipped Critic

In these experiments, I modify the critic in two distinct ways.

The first variant introduces a shared critic between the two agents. The motivation is that a centralized value function, aware of both agents' observations and actions, could provide a more informed

evaluation of the state, especially in a cooperative environment. The shared critic architecture employed in this work draws inspiration from the methodologies presented in (3) and (4). However, my implementation represents a simplified adaptation of the techniques proposed by these authors. Specifically, I utilize a single centralized critic network that incorporates the global state information from both agents during training. This shared critic then provides value estimates that are used by both agents to update their respective policy networks. While maintaining the core principle of centralized training with decentralized execution, my approach streamlines the original frameworks by employing a unified critic architecture rather than the more complex multi-critic or agent-specific critic configurations described in the referenced works.

The second variant instead applies clipping to the critic's value updates, analogous to the clipping already used in the policy updates. This technique, also used in the PPO implementation of Stable Baselines3 (5), is expected to improve training stability. In my case, I clip the value function but do not normalize the advantages.



In both variants, the policies converge more quickly to deterministic behaviors, but achieve lower average scores compared to the reward-shaped baseline. I hypothesize that the shared critic may suffer from conflicting optimization objectives as a single value function may struggle to accurately represent the distinct optimal behaviors required for different agent positions. The clipped critic variant instead may impose overly conservative constraints that limit the exploration necessary for discovering effective coordination strategies, trading off the stability benefits of clipping for reduced adaptability in the multi-agent coordination task.

4 Generalization Attempts with Curriculum Learning

After achieving promising results in the "Cramped Room" layout, I explored the agents' ability to generalize across different environments. A naive approach, randomizing the layout at the beginning of each episode, resulted in immediate failure, likely due to the complexity introduced by sudden changes in task structure. As a natural next step, I adopted a curriculum learning strategy.

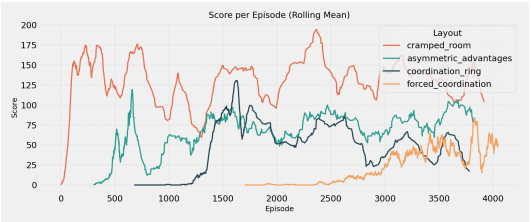
The agents were progressively exposed to four layouts: *Cramped Room*, *Asymmetric Advantages*, *Coordination Ring*, and *Forced Coordination*. A new layout was introduced only after agents demonstrated sufficiently strong performance on the previous ones. Additionally, the most recently added layout was sampled more frequently to allow for focused learning while preserving knowledge of earlier tasks.

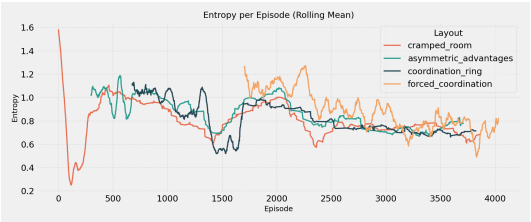
I evaluated curriculum learning using all PPO variants discussed earlier, and also introduced a new architecture based on convolutional neural networks (CNNs).

4.1 PPO Baseline

The first experiment employed the PPO baseline with custom reward shaping. This setup yielded the best generalization across multiple layouts. Agents were able to perform well simultaneously on three environments: *Cramped Room*, *Asymmetric Advantages*, and *Coordination Ring*. It is

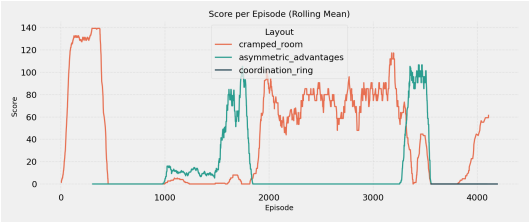
likely that more complex or larger networks could have further improved generalization to the final environment in the curriculum.

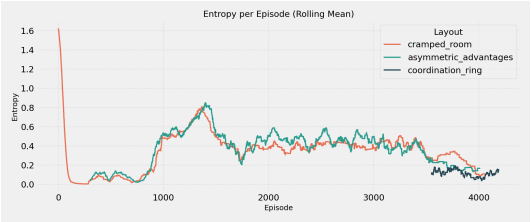




4.2 PPO with Clipped Critic

In this experiment, I used the same architecture and reward shaping as in the baseline PPO, with the only difference being the introduction of clipping in the critic updates. The goal was to stabilize training and accelerate convergence for each layout, thereby reducing overall curriculum training time. However, as shown in the plots, the algorithm struggles even to generalize to the second layout (*Asymmetric Advantages*). Although it eventually transitions to the third environment, it appears to forget how to perform well in the earlier one (*Cramped Room*).



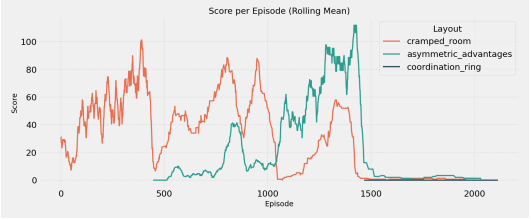


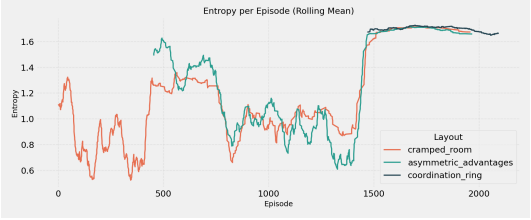
4.3 CNN-based PPO

This experiment explores a significantly different variant using two separate CNNs: one for the actor and one for the critic. The motivation is to leverage the spatial structure of the Overcooked environment. The grid-based layout may allow CNNs to capture reusable spatial patterns (e.g., "two onions in a pot" or "a soup near the serving window") that appear across different layouts.

However, Overcooked's raw observations are not designed to be used directly as input for CNNs. I therefore extracted the internal `OvercookedState` object after each joint action and passed it through a custom function that converts both state information and grid positions into a tensor of shape $15 \times 15 \times 40$. The final dimension aggregates four stacked frames to provide temporal context.

Despite extended training times, the results were disappointing. The agents were only able to generalize to the first two layouts (*Cramped Room* and *Asymmetric Advantages*), and failed to make progress when the third environment was introduced. Additionally, sporadic performance drops (e.g., zero scores even in familiar environments) were observed, suggesting unstable policy behavior. Moreover, the high computational cost of CNN inference and tensor preprocessing significantly slowed down training.





5 Hyperparameters

All experiments shared the same hyperparameter settings, which were chosen empirically through preliminary testing:

- $N = 1024$ (rollout length)
- Batch size: 128
- Number of PPO epochs: 6
- Learning rate $\alpha = 5 \times 10^{-4}$
- Discount factor $\gamma = 0.99$
- GAE $\lambda = 0.95$
- Clipping coefficient $\epsilon = 0.2$
- Entropy coefficient: 0.05
- Critic smoothing factor: 0.5

These values were kept fixed across all models to ensure fair comparison during evaluation.

6 Conclusions

In this work, I explored the application of Proximal Policy Optimization to the Overcooked multi-agent environment, with a focus on both performance and generalization. Through a series of progressively more complex experiments, I evaluated multiple PPO variants,including the standard PPO with reward shaping, a clipped critic version, and a shared-critic architecture, alongside an alternative approach based on convolutional neural networks (CNNs).

The results highlight that:

- The baseline PPO implementation with custom reward shaping achieves the best average performance and demonstrates the strongest generalization across multiple layouts.
- The clipped critic variant offers faster convergence but struggles to retain knowledge across different environments, showing signs of catastrophic forgetting.
- Sharing the critic between agents, although conceptually appealing for promoting global value estimation, did not lead to improved performance and may have limited the agents' ability to specialize.
- The CNN-based approach, while promising in theory due to its ability to extract spatial patterns, failed to outperform the MLP-based models and introduced significant computational overhead.

Additionally, I found that curriculum learning is essential for generalizing across diverse layouts: simply randomizing the environment per episode led to poor and unstable learning. A carefully constructed curriculum, combined with reward shaping, proved to be the most effective strategy for training agents capable of robust behavior in complex, cooperative tasks.

References

[1] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017.

[2] M. Carroll, R. Shah, M. K. Ho, T. L. Griffiths, S. A. Seshia, P. Abbeel, and A. Dragan, "On the utility of learning about humans for human-ai coordination," 2020.

[3] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," 2020.

[4] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," 2024.

[5] A. Hill, A. Raffin, M. Ernestus, A. Gleave, A. Kanervisto, R. Traore, P. Dhariwal, C. Hesse, O. Klimov, A. Nichol, M. Plappert, A. Radford, J. Schulman, S. Sidor, and Y. Wu, "Stable baselines." https://github.com/hill-a/stable-baselines, 2018.

[6] X. Wang, Y. Chen, and W. Zhu, "A survey on curriculum learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 4555–4576, 2022.

[7] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, (New York, NY, USA), pp. 41–48, Association for Computing Machinery, 2009.

[8] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," 2015.