Smart Team Play: Utility of Population-Based Training for Cooperative AI in Overcooked

Janaína Moreira-Kanaley
Supervisor(s): Robert Loftin, Frans Oliehoek
EEMCS, Delft University of Technology, The Netherlands

June 19, 2022

A Dissertation Submitted to EEMCS faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering

Abstract

In an ad-hoc teamwork environment, artificial intelligence agents have the potential to take on supportive roles and complete tasks in collaboration with human players. The following paper investigates the use of employing population-based training (PBT) for reinforcement learning agents in the multi-player game Overcooked. In addition to this, the research examines whether the incorporation of highly mutated agents, which serve to introduce noise into the initial population, could enhance the final performance of PBT. As the method used to answer the previous inquiries, the learning curve of a selected PBT agent is first evaluated and its final performance with a human proxy then examined within different layouts of the game. Following this method, it was concluded that PBT, and other self-play agents, have the tendency to drastically underperform against the human proxy and agents that are trained based on human data. Furthermore, while incorporating the mutated agents increased sample efficiency in layouts with low risk of collisions, it had negligible effect on the final performance of PBT with the human proxy.

1 Introduction

As advancements in technology grow, the adoption of reinforcement learning (RL) based artificial intelligence (AI) serves as an attractive solution for otherwise strenuous and repetitive human-operated tasks. From self-driving vehicles [1] to assisted decision-making in healthcare [2], RL agents have the ability to assume a multitude of roles in vastly different fields. In the case of operations that require human collaboration, the addition of a cooperative AI agent similarly has great potential to aid automation and efficiency. Within that context, multiplayer games provide an isolated and risk-free opportunity to investigate the uses and limitations of human-AI cooperation. Overcooked [3] is one such collaborative game and involves the cooperation of up to four players for the completion of cooking tasks. The aim of this paper is to assess the utility of RL-based agents in the multi-agent setting of Overcooked, trained with the framework known as population-based training (PBT).

Among the different methods of training collaborative RL agents, a strong case can be made for the use of population-based training. Compared to zero-shot coordination agents discussed in [4] and [5], PBT allows for ad-hoc teamwork in a multi-agent setting where it can adapt through long-term interaction to different agent types that may join the group dynamically. Additionally, PBT provides an advantage over self-play (SP) methods [6], [7] where agents train against a copy of themselves repeatedly, resulting in a learned policy that lacks generalization to carry over to other agent types. Population-based training ideally removes this issue considering it uses a population of independently-initialized policies to train against [8].

Besides the previously discussed advantages, the research into PBT also acts as a form of validation of results from other related works. In this field of research, some literature has already investigated the utility of using PBT for training RL agents in the same Overcooked environment [9]. By reproducing the experiments in the performed research [9], results and derived conclusions are able to confirm each other, increasing their credibility.

The aforementioned benefits associated with assessing PBT motivate the purpose of the research, which is to investigate whether the framework holds success when paired with a human player in Overcooked. This research can be fulfilled by answering the following questions:

- How does a PBT agent perform in a cooperative environment when paired with a

human player?

- What variations to PBT could improve an agent's performance with a human player?

As a response to the research questions, results show that PBT outperforms SP agents with a human proxy but underperforms against agents trained on human data. Based on related works [10], it was assumed that the performance of the PBT agent should improve if high levels of diversity are introduced to the initial training population. However, while the sample efficiency increased during training in some layouts, the final performance of PBT showed little to no signs of improvements. More research is therefore necessary to fully answer the second research question.

In order to provide a full overview of the research conducted, several steps are described within the paper. Related works that concern the research are accounted for in Chapter 2. The necessary background for comprehension of the research is presented in Chapter 3. The methodology used for executing the research is explained in Chapter 4. Results of the research are detailed in Chapter 5. The ethical concerns of the research are discussed in Chapter 6. Lastly, a reflection of the achieved results is provided in Chapter 7.

2 Related Work

Population-Based Training

Within this research area, an analysis of an existing implementation of PBT with Proximal Policy Optimization in the same Overcooked environment has already been conducted [9]. The performed research in PBT is based on this previous work. According to [9], agents trained to play with other AI agents perform drastically worse when paired with humans. Similarly, they claim that agents trained through a combination of self-play and human behavior show higher performance during teamwork with human players. Performing a secondary research of PBT allows for the comparison and confirmation of the results derived in the previously conducted research. In addition to this, changes to PBT not analysed within the work can be approached in order to determine possibilities for improving its performance.

Ad-Hoc Teamwork

One benefit PBT has to offer is its ability to provide ad-hoc teamwork in a multi-agent setting compared to the zero-shot coordination techniques seen in other collaborative RL works such as [4], [5]. Both ad-hoc teamwork and zero-shot coordination can be described as learning problems where the learned policy is compatible with different agent types in a collaborative environment [4]. Unlike ad-hoc teamwork however, pure zero-shot coordination focuses on short-term interaction and is not suitable for a scenario where an unknown agent joins a previously existing group of agents [4]. As a result, PBT serves as a great solution for long-term interaction in which agents need to adapt their policies to newly joined different agent types.

Self-Play (SP)

One critical issue for using multi-agent RL in collaborative tasks is the tendency for policies to overfit when paired with themselves and their failure to generalize to human players.

Several works make use of self-play (SP) for competitive gameplay [6], [7], which consists of training a single agent against a copy of itself as seen in Figure 1.

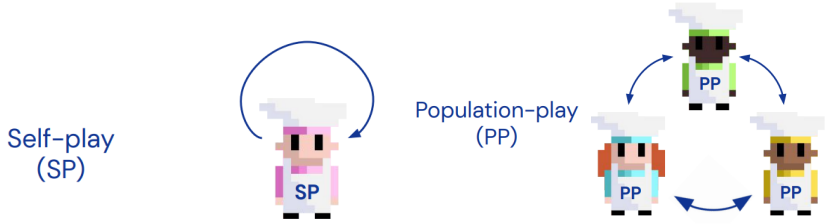


Figure 1: Depiction of self-play and population-based training methods. Both are frameworks for training RL agents in multi-agent settings. Illustrations belong to [10].

Due to the homogeneous nature of the training, frameworks such as SP especially suffer from vulnerability to human variations [9]. Population-based training possibly offers a solution to this issue by having a population of independently-initialized policies to train against, as depicted by Figure 1. Ideally, a population with multiple members will reduce overfitting and account for the differences in human behavior.

Population Diversity

Some of the related literature tackle the lack of generalization to human players by training against a diverse set of population. The work in [11] examines the effect of training agents sequentially against one another, while in [10] an agent is trained against other independently-trained agents in a method known as fictitious co-play. Although the first work has not yet been compared to PBT, the second states that fictitious co-play outperforms PBT in the same Overcooked environment. These results seem to suggest a positive effect may come from examining the influence of diversity during PBT, possibly by creating high variance between agents in the population before training.

3 Background

Overcooked Environment

In order to gauge the efficacy of RL algorithms in a cooperative setting, a simplified Overcooked environment (see Figure 2) was employed based on the original game Overcooked [3]. Similarly to the original game, players can control chefs inside a kitchen to prepare and deliver dishes. The objects within the simplified environment, however, consists of only onions, dishes, and soups. Each layout in the game has an unlimited supply of onions and dishes which can be accessed from the corresponding dispenser. A standard cooking itinerary within the game can be described as follows: players place three onions in a pot, wait 20 timesteps for the soup to cook, place the soup in a dish, and serve the dish which rewards all players with a value of 20 [9]. Players can choose from six possible actions: up, down, left, right, noop, and interact, whose results differ based on the faced tile. Within the Overcooked environment, agents should be able to navigate the layouts and collaborate together to serve dishes in a manner that maximizes the value of the received rewards.

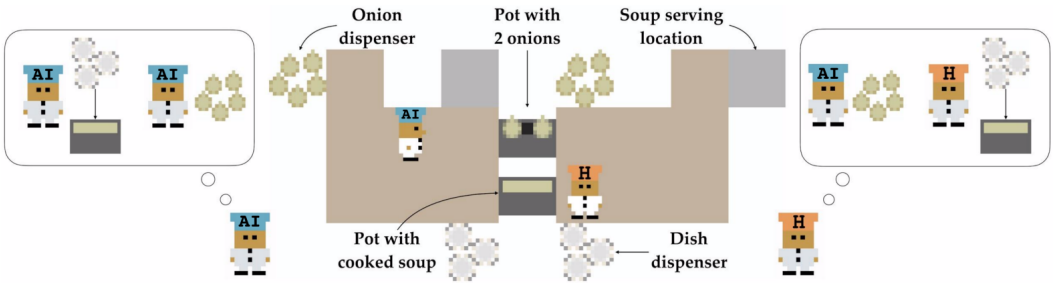


Figure 2: Overview of the Overcooked environment used for the research (retrieved from the previous Overcooked paper [9]). The observation space for this environment consists of a fully observable Boolean image with values that indicate whether a state is an onion, dispenser, etc.

Reinforcement Learning (RL)

Reinforcement learning trains policies based on states that are likely to maximize rewards. In this case, the reward function is defined in terms of sparse and dense rewards. Sparse rewards involve long-term actions and are awarded for only a few actions such as a dish being served. Dense rewards focus on short-term actions and are awarded for many actions such as avoiding collision or interacting with an item. Dense rewards are useful for feedback on instantaneous actions and help determine whether the current trajectory is appropriate.

Population-Based Training (PBT)

Population-based training is an evolutionary algorithm that optimizes neural networks [8]. It achieves this by randomly drawing pairs of agents from a population, training them against each other, and replacing the worst performing agents with mutated hyperparameters from the best performing policies. During training, the ranking of the agents is determined using dense rewards. In this paper, the RL algorithm that updates the policies is Proximal Policy Optimization.

Proximal Policy Optimization (PPO)

Proximal Policy Optimization is an on-policy reinforcement learning algorithm [12], used in this case to update the policies during PBT. The updates are performed in a way that prevents high variance from the original policy during training [12]. Its implementation consists of the actor-critic model which makes use of two deep neural networks that handle action and reward respectively [12]. The trained policy controls the actions of a singular agent during play.

Behavioral Cloning (BC)

Behavioral cloning relies on learning a policy from the demonstrated desired behavior of an expert. It learns a mapping from state to action through the use of supervised learning. The implementation of the BC-based agent models used in the research consists of: an encoding of the state as input, the probability distribution over actions as output, and the cross-entropy loss function for training [9].

Human Models

The human models employed during the experiments are behavioral cloning models based on collected human data from the previous research [9]. Human-human trajectories were gathered for each layout and split into two subsets of single-agent trajectories. For each layout and subset, a human model was learned through behavioral cloning. One model acts as the version that can be accessed for training agents (the BC agent), while the other is used to determine the final performance by simulating a human player (the H_{Proxy} agent).

4 Methodology

Performance Measurements

In order to determine how PBT performs in a cooperative setting, first the factors used to measure that performance must be defined. Two factors are employed for the evaluation of the agent: learning curves and the final performance when paired with a human proxy.

As the first performance measure, learning curves give a brief overview of the sample efficiency, in other words, how much data is necessary to train a optimal policy using PBT. When presenting the results, the learning curves are shown for each experiment layout as the mean episode reward over environment timesteps. The mean episode reward is the average of sparse rewards of a chosen agent paired to all other agents in the population including itself. An episode consists of horizon timesteps that indicate the amount of steps agents play a game for. Ultimately, the learning curves help indicate the required amount of environment timesteps for the agent to reach peak performance.

As the second performance measure, the mean episode reward during gameplay with the human proxy is compared against the performance of different agents also paired with the proxy. This provides a relative evaluation of the final performance of the PBT agent when matched with a human proxy. Critical is that the final performance evaluation against the human proxy serves as an estimate for how well the PBT can generalize to other unknown agents, in this case a human player.

Experiment Variations

In order to answer the second research question, changes that may enhance PBT must be investigated. As elaborated in Chapter 2, related works such as [10] seem to indicate that increasing noise in the PBT population may lead to agents with improved generalization to human players. Following this idea, the experiment variations entail mutating selected members of the initial population for some number of iterations in an attempt of diversification. The mutation iterations take place once at the start of training, which makes this variation fairly simple and cost effective.

Experiment Layouts

Due to time constraints concerning computational costs, only two different layouts will be analyzed. For each layout, agents are trained over three different seeds in order to minimize noise. The layouts chosen can be seen in Figure 3. The selection of these layouts was based on their ability to display the agents' capability to maximize their strengths (Asymmetric Advantages) as well as their ability to coordinate among layouts that have a high chance of collision (Coordination Ring).

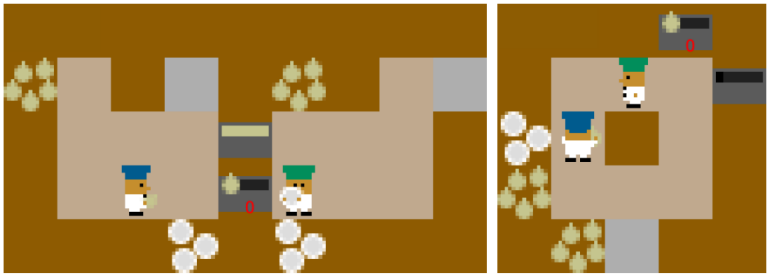


Figure 3: Chosen game layouts for the conducted experiments. From left to right: Asymmetric Advantages, Coordination Ring. Image is taken from [9].

Baseline Agents

As a way to contextualize the efficacy of PBT relative to other implementations, PBT's final performance with the human proxy is evaluated against the performance of other baseline agents paired with the proxy. The different agents types that are used in the comparison can be seen listed below. Parameters employed for training the agents are the same as in the previous paper [9].

- H_{Proxy}: Human model trained through behavioral cloning that simulates the human player.
- BC: Imitation agent based on a separate instance of the data used to train the human proxy.
- SP: PPO agent trained using the purely self-play method.
- PPO_{BC}: PPO agent trained with the human model BC.
- PPO_{HProxy}: PPO agent trained with the human proxy. It gives an overview of the peak performance that could be achieved by PPO given full access to the human proxy.

5 Results

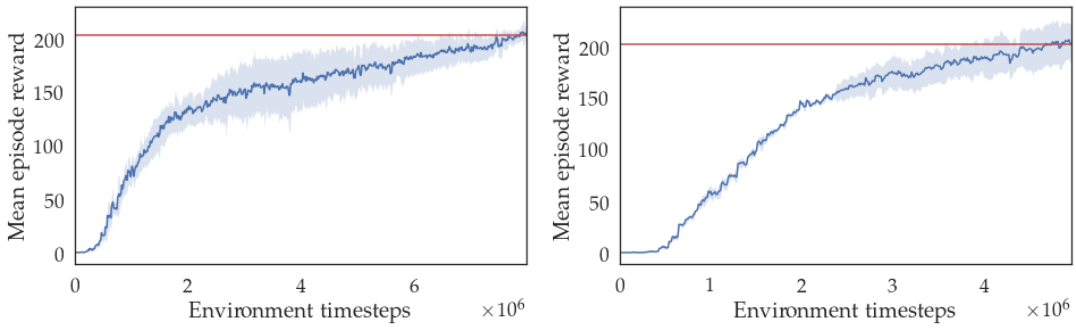
5.1 Experimental Setup

Described results are products from conducted experiments that have been divided into two phases, each targeting one research question respectively. The first phase focuses on evaluating a PBT agent trained based on experiments of previous literature [9] and attempts to reproduce the results derived in that work as a form of confirmation. The second phase handles the analysis of applying the experimental mutation iterations to PBT based on the same parameters of [9]. The selection of the mutation iterations was based on increasing and decreasing the number of iterations until some interesting behavior was shown in the learning curve (e. g. enhanced sample efficiency). Details of the iterations can be found in Appendix A.

5.2 Results After Reproduction of Experiments

Learning Curves

The learning curve of the PBT agent over both experiment layouts, Asymmetric Advantages (AA) and Coordination Ring (CR), is graphed (see Figure 4) and evaluated in order to examine the sample efficiency associated with training the agent.



(a) Asymmetric Advantages (AA)

(b) Coordination Ring (CR)

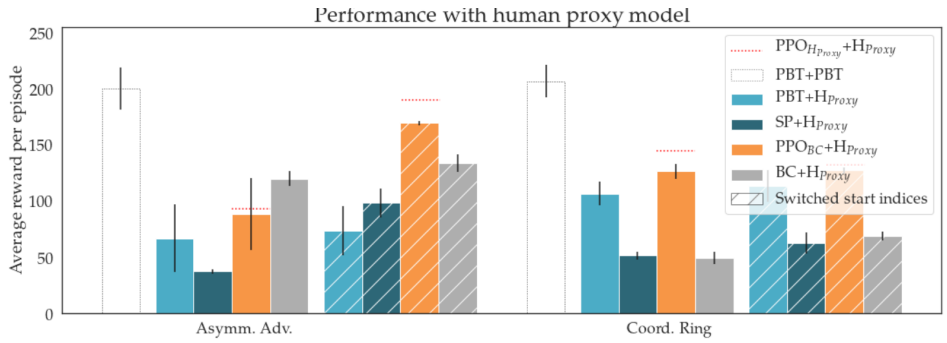
Figure 4: Learning curves of a PBT agent in the Asymmetric Advantages and Coordination Ring layouts, displaying the average sparse reward per episode (mean of 100 episodes) during training over 400 horizon timesteps. The shaded area indicates the standard error over three different seeds. The red line shows the mean reward in the last timestep. As the end population during PBT is fairly similar in terms of policy performance, the graphed agent was chosen randomly (first member in this case). The agent is evaluated against all population members including itself.

A quantitative analysis of Figure 4 is presented. In the first layout, a higher frequency of deviations between the used seeds can been seen. In both layouts, the trained agent sports very similar final performances, with AA and CR ending with a mean sparse reward of 203 and 202 respectively. On the other hand, the agent in AA requires 60% more environment timesteps ($8x10^6$) than in CR ($5x10^6$) to reach a maxima in performance. This means that the agent trained in the AA layout has a lower sample efficiency and requires more data for policy optimization. In total, for a complete run over all the timesteps using only one seed it took $\sim$7 and $\sim$5 hours for AA and CR respectively.

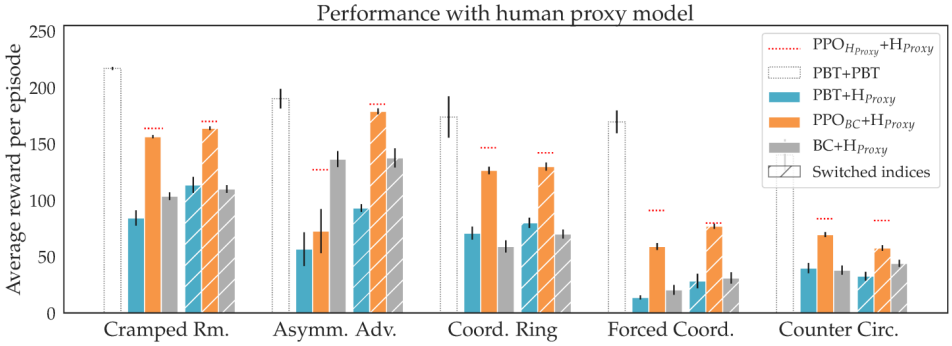
In theory, the differences between the number of timesteps during training could be tied to the amount of coordination necessary and the chances of collision within a layout. Unlike Coordination Ring, Asymmetric Advantages has no possibility for agent collision and allows both agents to complete tasks fully by themselves. This means that agents in AA may reach higher rewards early during training as they can serve the dishes completely independently without facing issues posed by colliding with other agents. However, these achieved rewards may in actuality be mediocre performances which are not optimized by fully coordinating with the other agent. On the other hand in CR, the agent must coordinate from the beginning with its teammate to avoid collision and maximize rewards. As a result of this forced coordination due to collision, the agent reaches its peak performance in a lower number of timesteps than in non-mandatory coordination layouts such as AA. In order to confirm this theory, however, more research is required.

Final Performance

In order to evaluate the final performance of PBT, all agents types are paired with a human proxy and each pair's average reward per episode is graphed as a bar chart in Figure 5a. The results of the experiments in Figure 5a are then compared and validated to the results presented by [9] in Figure 5b.



(a) Performance results produced from self-conducted experiments.



(b) Performance results produced from previous work [9].

Figure 5: Performance results of the PBT agent when paired with the human proxy in comparison to performances of other agents matched with the proxy. For each layout, performance is given by the average sparse reward per episode (mean of 100 episodes) over 400 horizon timesteps. Hashed bars indicate results from switching the starting positions of the agents. The graph from [9] depicts three additional layouts and uses five instead of three seeds for PBT during their experiments.

The performance of PBT is firstly examined in relation to its optimum performance. In Figure 5a, PBT performs drastically worse when paired with a human player (mean reward per layout over the indices: 70, 110) than with itself (mean reward: 200, 207). The same conclusion can be arrived in Figure 5b. However as all other models perform worse than their optimum, the dip could be at least partly attributed to H_{Proxy} performing suboptimally itself.

Next, PBT is analyzed in respect to the baseline agents. As supported by both research [9], PBT does outperform self-play which sports a mean reward of 68 and 57 per layout respectively in Figure 5a. In contrast, the PPO_{BC} agent trained with the human BC model has a higher performance with the human proxy (mean reward: 129, 127) than PBT. Furthermore, similarly to the results presented in Figure 5b, the performance of the

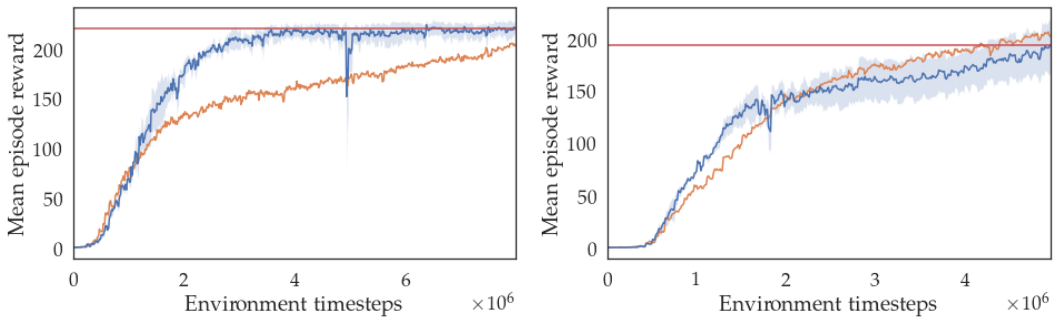
BC agent (mean reward: 127, 59) varies between that of the PBT and PPO_{BC} agents as it is a relatively simple model that only mimics a separate subset of the gathered human data. The gap between the optimum performance of PPO_{HProxy} and PPO_{BC} serves to show the differences between the individual human models H_{Proxy} and BC.

In summary, PBT paired with the human proxy does serve as an improvement to the self-play method, however, underperforms against agents trained based on human data. This means that, in part, the population of independently-initialized policies has increased the generalization capabilities towards a human player compared to SP. It understandably fails to other baseline agents as it does not learn its policy from human data.

5.3 Results After Variation of Experiments

Learning Curves

After applying the iterations of mutation (see Appendix A) to the population of PBT, an examination of the learning curves in the chosen layouts can determine the variation's influence on sample efficiency. These learning curves are plotted in Figure 6.



(a) Asymmetric Advantages (AA)

(b) Coordination Ring (CR)

Figure 6: Multiple learning curves of a PBT agent in the Asymmetric Advantages and Coordination Ring layouts, displaying the average sparse reward per episode (mean of 100 episodes) during training over 400 horizon timesteps. The blue learning curve is the result from applying the mutation iterations to PBT, with the shaded area indicating the standard error over three different seeds. The red line shows the blue curve's final performance reached. The orange learning curve is the original curve (mean of curves over three seeds) from before variations were applied to PBT. Similar to the original curves, the graphed agent is the first member of the population, evaluated against all members.

An analysis of the learning curves displayed in the Asymmetric Advantages layout is performed. As shown in Figure 6a, the variations have accelerated the rate in which the agent's performance arrives at a plateau. Within $4\text{x}10^6$ environment timesteps the blue learning curve has already reached its peak performance (220), as indicated by the red line. This is a full 50% decrease from $8\text{x}10^6$ in terms of environment timesteps necessary for policy optimization. On the other hand, the standard PBT agent without any variations requires all $8\text{x}10^6$ environment timesteps to reach an optimum of 203 which is below that of the mutated agent. This means that after the variations, PBT in the AA layout has a higher

sample efficiency. In addition to this, the blue learning curve displays fewer deviations over the seeds than that of the standard PBT agent seen in Figure 4a.

The learning curves of the Coordination Ring layout depicted in Figure 6b are examined. In this case, both learning curves are fairly similar. Firstly, the shape of the curves and the rate at which they reach peak performance are quite close to each other. Furthermore, both agents require the full $8x10^6$ environment timesteps to reach a maxima in performance. The performance deviations across the seeds appear to be similar as well when comparing Figure 6b and Figure 4b together, with the mutated agent displaying deviations a bit more frequently early. At the last timestep, the agent without variations finishes PBT with a higher performance (202) than the mutated agent (193), as indicated by the red line of the blue learning curve.

As described by the analysis of the figures, the variation significantly improves the sample efficiency of the agent in the AA layout, however, does not seem to have much of an effect on the learning curve in CR and even underperforms in terms of optimum performance reached. A theory behind the boost in layout AA could be that the variations in the population created policies which force agents to coordinate with each other to maximize rewards. A previously supposed issue of AA is that agents can perform independently to achieve decent but subpar rewards. The mutation in the population might have introduced a scenario in which agents can no longer perform decently without coordinating with each other. As a result, this could be the reason as to why the peak performance is reached in a much lower amount of timesteps than previously. Additionally, this theory would explain why the variation had very little effect in the second layout, as agents were already forced to coordinate in it from the start of training due to the high chances of collision.

Final Performance

In order to assess the effect the variation has on the final performance of PBT, all agents types are paired with a human proxy and each pair's average reward per episode is graphed as a bar chart in Figure 7. The figure is then compared to the results of the original unvaried experiments in Figure 5a.

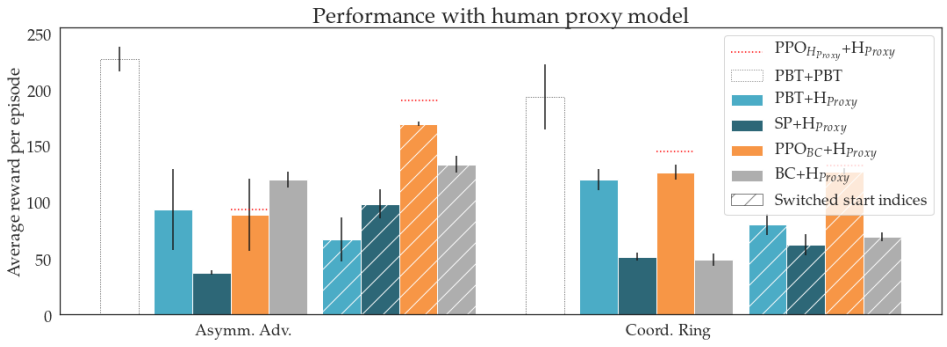


Figure 7: Performance results of the mutated PBT agent when paired with the human proxy in comparison to performances of other agents matched with the proxy. For each layout, performance is given by the average sparse reward per episode (mean of 100 episodes) over 400 horizon timesteps. Hashed bars indicate results from switching the starting positions of the agents.

The performance of PBT displayed in Figure 7 is analysed. In the Asymmetric Advantages layout, the agent has a mean average reward of 80 over the switched indices, outperforming unvaried PBT in Figure 5a (mean of 70). On the other hand, the varied agent achieves a mean average reward of 100 in the Coordination Ring layout, underperfoming the original PBT which yields 110. The performance in respect to the other baseline agents stays mostly the same as Figure 5a, with the exception of PBT slightly outperforming PPO_{BC} at one starting position in layout AA. In the unvaried experiment, the mean reward of PBT was always strictly below PPO_{BC}. Overall, the final performance of the PBT agent after both experiments is very similar.

As described in the analysis of Figure 7, the application of the mutation iterations to PBT seems to not evoke any negligible change to the average final performance of the agent with the human proxy. As a result, while adding high variance agents to the initial population seems to improve sample efficiency for some layouts, it is unclear whether the performance of PBT can be improved in this manner.

6 Responsible Research

In order to adhere to ethical standards concerning the presentation of honest and reliable findings, it is crucial to consider the ethical aspects of a research.

For the conducted research, results could have been misrepresented in various ways. Firstly, the evaluation of the PBT agent heavily relies on using the human models which are based on previously gathered human data. During the data collection process, this data could have in some form been omitted or inaccurately recorded which would make the conclusions regarding the performance of the PBT agent invalid. However, this seems unlikely considering that the original paper, which used the same human models, conducted additional experiments with actual human players that support their findings.

Another manner in which the findings could be misrepresented, would be if the data concerning the reported agents' learning curve or final performance was fabricated or falsified. Environment parameters, such as the seeds or number of episodes carried out, could be inflated or inaccurately reported which may affect the validity of the results. However, this would defeat the purpose of the research as it tries to accurately assess the utility of using PBT in a cooperative setting and does not profit from skewing the conclusion into one direction rather than the other.

In regards to the reproduction of the experiments, the results described in the paper can be reproduced as long as there is computational power available. The seeds and specifications for the experiments are fully detailed in a public repository[1]. A Jupyter Notebook used to generate the graphs that confirm the findings is additionally provided. The largest bottleneck when it comes to reproduction is training the agents which may take a few days to complete. However, if time is available then the experiments are fully reproducible and can be verified.

Ultimately, results arrived during the research could have potentially been victim to data falsification or fabrication, however this is unlikely as previous researchers have confirmed

[1] `https://hub.docker.com/repository/docker/jmoreirakanaley/pbt`

the results and it would defeat the purpose of the investigation. In the case of doubt, derived findings can be verified as the experiments are reproducible.

7 Conclusions and Future Work

Although agents that use self-play-like methods during training can perform well when paired with themselves, this performance dives drastically when having to collaborate with a human player. Within the collaborative game Overcooked, it was shown that population-based training underperforms against agents which are trained with a human model. However, experiments also confirm that population-based training serves as an improvement compared to purely self-play techniques. From this it can be concluded that the incorporation of a population of independent policies has been in part successful in enhancing performance with a simulated human player. When it comes to sample efficiency, PBT seems to require lower amounts of data to optimize policies in layouts that have high chances of collision and require high levels of coordination.

In an attempt to improve generalization in gameplay with a human player, mutation iterations were introduced to the initial PBT population. As a result, while this has improved sample efficiency in layouts with low levels of collision, the addition of noise seems to have little effect on the final performance of PBT itself. In order to determine whether high variance population members may improve PBT's final performance, more time and research is required.

Limitations

The main limitations during the research were constraints caused by the lack of computational power and limited time to conduct the research. In any instance where the performance of PBT had to be assessed, a population of agents were trained multiple times over several seeds that took many hours each to complete. Coupled with the short period of the research, this added a lot of overhead to the amount of parameters (e. g. seeds, population size, etc.) and investigations that could be performed.

Future Work

As future work, continued research into the effects of incorporating high variance agents into the population of PBT could yield a definite answer to whether final performance can be improved. Some investigation into possibly more sophisticated methods of increasing noise besides mutating the population would also prove useful here.

Another possibility for future work would be to examine the effects of adding a human based agent such as the BC agent into the population of PBT. It is assumed that the main culprit behind PBT's poor performance against agents trained based on a human model is the lack of access to human data during training. The addition of a human model to PBT would allow research into if that is in fact the case and confirm whether PBT would also outperform agents such as PPO_{BC} since it is trained based on a population of policies. If the latter can be proven it would also support the idea that diversity in the population improves generalization against a human player.

References

[1] B. R. Kiran, I. Sobh, V. Talpaert, *et al.*, "Deep reinforcement learning for autonomous driving: A survey," *CoRR*, vol. abs/2002.00444, 2020. arXiv: 2002.00444. [Online]. Available: https://arxiv.org/abs/2002.00444.

[2] C. Yu, J. Liu, and S. Nemati, "Reinforcement learning in healthcare: A survey," *CoRR*, vol. abs/1908.08796, 2019. arXiv: 1908.08796. [Online]. Available: http://arxiv.org/abs/1908.08796.

[3] Ghost Town Games, *Overcooked!* 2016.

[4] H. Hu, A. Lerer, A. Peysakhovich, and J. N. Foerster, ""other-play" for zero-shot coordination," *CoRR*, vol. abs/2003.02979, 2020. arXiv: 2003.02979. [Online]. Available: https://arxiv.org/abs/2003.02979.

[5] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *CoRR*, vol. abs/1706.02275, 2017. arXiv: 1706.02275. [Online]. Available: http://arxiv.org/abs/1706.02275.

[6] D. Silver, T. Hubert, J. Schrittwieser, *et al.*, "Mastering chess and shogi by self-play with a general reinforcement learning algorithm," *CoRR*, vol. abs/1712.01815, 2017. arXiv: 1712.01815. [Online]. Available: http://arxiv.org/abs/1712.01815.

[7] OpenAI, *Openai five defeats dota 2 world champions*, Sep. 2020. [Online]. Available: https://openai.com/blog/openai-five-defeats-dota-2-world-champions/.

[8] M. Jaderberg, V. Dalibard, S. Osindero, *et al.*, "Population based training of neural networks," *CoRR*, vol. abs/1711.09846, 2017. arXiv: 1711.09846. [Online]. Available: http://arxiv.org/abs/1711.09846.

[9] M. Carroll, R. Shah, M. K. Ho, *et al.*, "On the utility of learning about humans for human-ai coordination," *CoRR*, vol. abs/1910.05789, 2019. arXiv: 1910.05789. [Online]. Available: http://arxiv.org/abs/1910.05789.

[10] D. Strouse, K. R. McKee, M. M. Botvinick, E. Hughes, and R. Everett, "Collaborating with humans without human data," *CoRR*, vol. abs/2110.08176, 2021. arXiv: 2110.08176. [Online]. Available: https://arxiv.org/abs/2110.08176.

[11] B. Cui, H. Hu, L. Pineda, and J. Foerster, "K-level reasoning for zero-shot coordination in hanabi," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34, Curran Associates, Inc., 2021, pp. 8215–8228. [Online]. Available: https://proceedings.neurips.cc/paper/2021/file/4547dff5fd7604f18c8ee32cf3da41d7-Paper.pdf.

[12] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017. arXiv: 1707.06347. [Online]. Available: http://arxiv.org/abs/1707.06347.

A Experiment Specifications of Mutation Iterations

Mutation iterations are conducted once at the start of population-based training (PBT). For the conducted experiments, the listed iterations were applied for each member of the initial population.

Table 1: Specifications of experiment variations. For the experiments, a population of three was used during population-based training. All three members of the population are listed along with the applied number of iterations. In this case, chosen iterations are based on intervals of 40.

Population Member	Mutation Iterations
1	0
2	40
3	80