Training a Negotiating Agent through Self-Play

Renāts Jurševskis
Supervisor(s): Bram Renting, Pradeep Murukannaiah
EEMCS, Delft University of Technology, The Netherlands

23-6-2022

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

Training a Negotiating Agent through Self-Play

Renāts Jurševskis
Supervisor(s): Bram Renting , Pradeep Murukannaiah
EEMCS, Delft University of Technology, The Netherlands

Abstract

Recent developments in applying reinforcement learning to cooperative environments, like negotiation, have brought forward an important question: how well can a negotiating agent be trained through self-play? Previous research has seen successful application of self-play to other settings, like the games of chess and Go. This paper explores the usage of self-play within the training of a negotiating agent and determines if it is possible to successfully train an agent purely through self-play. The results of the experimentation show that a training stage using self-play can match or even exceed an approach using a set of training opponents. By using multiple self-play opponents, the average utility can be further improved by introducing more variance during training. In addition, using a combination of both self-play and training opponents leads to a hybrid approach that performs better than either of the two techniques separately.

1 Introduction

Negotiation is an interactive process through which different parties aim to achieve their preferred result. Since the preferences of the negotiating parties are likely to differ, to reach an agreement, one or multiple sides must concede. However, unlike purely competitive environments like chess or Go, negotiation rewards the cooperation between parties. Instead of choosing to purely attempt to maximize their own utility, agents can collaborate to receive a satisfactory result for all parties involved.

To allow for large-scale practical applications of negotiation, the process can be automated by using negotiating agents, which cooperate with the opponent to come to a mutual agreement. Negotiation in settings with incomplete information is particularly important for real-world applications, where the opponent preferences are unknown [3]. The usage of such a negotiating environment allows for an efficient simulation of bilateral negotiations, which can be applied to real-world problems like e-commerce and energy markets [7; 9]. Because of the high potential of using automated negotiation within various commercial applications, the demand for accurate negotiating agents is high and numerous approaches and techniques have been thoroughly researched [2].

When training a negotiating agent, it is usually trained against a set of default opponents. However, training can also be done by putting the agent against itself. By training through self-play, it is not required to collect an initial set of opponents. Because of this advantage, self-play can be applied to problems without requiring extensive exploration of various opponent strategies, gathering data, and evaluating different approaches. However, similar to traditional methods with a set of default opponents, there is a challenge when transitioning to a testing environment: although an agent developed using self-play will perform well against agents that follow similar principles, it might struggle against others that employ completely different strategies that were not thoroughly explored within training. For example, when playing an opponent that is not willing to concede, the agents might not be able to reach an agreement before the deadline of the negotiation session.

There has been previous research on the application of self-play in various competitive settings like chess, Go, and shogi [11; 12]. Traditionally, most of the focus has been on zero-sum domains, where multiple agents are competing for a fixed reward. However, more recently, self-play has been successfully applied to more cooperative settings [8]. Because of this, the usage of self-play in negotiating agents has potential for further research.

This paper aims to answer the research question: *How well can we train a negotiating agent without a default set of opponents through self-play?*

To answer this question, the research explores various properties and applications of self-play, investigates the effectiveness of training only using self-play, and compares various conditions under which self-play yields the best results. In particular, a new self-play opponent is added to the negotiating environment, which is then used to train the agent through self-play. The performance of this agent is tested and compared to the baseline, continuing with the exploration of multiple self-play opponents during training to introduce additional variance. Finally, a hybrid approach utilizing both self-play and a default set of opponents is tested to evaluate the potential of using self-play to further improve existing agents.

2 Related Works

Traditionally, most research on machine learning and artificial intelligence has been based on interpreting and analyzing baseline data provided to the model. However, the acquisition of effective data can be very resource-intensive, unreliable, or sometimes even impossible [12]. When looking at applications of machine learning within negotiating agents, the agent is usually trained by performing numerous negotiations against a set of training opponents. Nevertheless, extensive training does not necessarily lead to good results, as for a negotiating agent to perform well, it must learn to play against various agents that apply different strategies to achieve a good performance. In particular, while a negotiating agent might perform well during training, it does not necessarily translate to a testing environment, with previously unknown opponents [3]. The generalization ability of an agent depends on the training conditions, and can also differ based on the underlying techniques used within the agent. Reinforcement learning in particular has been shown to require a large training data set in other settings [5].

However, there are ways to avoid the need for extensive baseline data. One such option is to train the machine learning model by putting an agent against itself. This concept is commonly known as self-play and has been extensively applied in various competitive games with perfect information. Excellent results were first achieved in the game of Go using an *AlphaGo Zero* algorithm, trained exclusively through self-play [12]. Furthermore, the usage of self-play within competitive games has further been explored with the introduction of *AlphaZero*: a more generic version of *AlphaGo Zero*, which can be applied to various domains, requiring no domain-specific knowledge except the game rules [11]. In particular, *AlphaZero* has been successfully applied to the games of chess and shogi, resulting in outstanding performance [11].

In addition to perfect-information games, self-play has shown to be effective in settings with imperfect information. In particular, research has found the usage of self-play to approach state-of-the-art methods in Limit Texas Hold'em poker [6], where players do not know the cards of their opponents. More cooperative settings have also seen the usage of self-play. Liu et al. [8] introduce training through self-play in a cooperative version of pong, where both players receive equal punishment for either of them missing the ball. This research shows that training through self-play can lead to an improvement in performance [8], therefore, it also has the potential to be used in the training stage of bilateral negotiation agents.

3 Methodology

To answer the research question, we divide it into multiple smaller sub-questions, starting with investigating the definition and previous applications of self-play. Then, the benefits and potential drawbacks of self-play are explored, before researching the specific ways of using self-play within the training stage and implementing a self-play opponent. Afterward, an evaluation is performed to compare the usage of self-play during training against a conventional training approach utilizing a default set of opponents. Finally, the paper explores the various training conditions, for which self-play provides the best results.

In addition, the practical elements of the research are based on a negotiating agent, which uses Proximal Policy Optimization (PPO) to determine the next bid. The negotiations are performed in a custom negotiation environment, which utilizes the Stacked Alternating Offers Protocol (SAOP) [1] and ranges over a variety of different domains. Additionally, the default implementation trains the agent using a set of training opponents, which is followed by an evaluation of the agent to set a baseline result. To answer the research question, this agent needs to be modified to support training using self-play, so it can evaluate the effectiveness of self-play.

The modification of the baseline framework begins with the implementation of a self-play opponent. This agent shares many similarities with the original PPO agent, with slight differences in the process of updating the policy. In addition, the self-play opponent uses a separate policy, such that both of the agents are not necessarily following a similar strategy. Once the opponent is fully implemented and functional, a full self-play approach simulation follows, where no training agents are used. Then, this model will be evaluated in various settings and compared to the result of the baseline. For the comparison, a set of 10 negotiation agents from the TU Delft course CSE3210 (Collaborative Artificial Intelligence) will be used. Finally, a hybrid training approach and training against multiple self-play opponents will be explored, to find the conditions where self-play results in the best performance.

4 Experiment Setup

As self-play is a general approach that can be used in many multiplayer environments and has previously been successfully applied to various settings like chess or go, it can be easily transferred to new domains and problems. To determine the viability of exclusively using self-play during the training of a negotiating agent, it is crucial to not only understand previous usages and benefits of self-play but also work on implementing it within the existing environment as a self-play opponent that can be used during training. In addition, the agent is further modified to be able to train against multiple self-play opponents at the same time to analyze the exact conditions, under which training using self-play leads to the best utility.

4.1 Implementation of Self-Play

The initial agent is based on a reinforcement learning approach using Proximal Policy Optimization (PPO) to train the bidding strategy. Similar to other reinforcement learning methods, the agent utilizes a policy to determine, which action to take to achieve the highest reward. Then, this reward is fed back into the policy, which is updated based on the results of the action. However, where PPO differs from traditional policy gradient methods is the technique used for updating the policy: instead of performing a single gradient update, PPO utilizes mini-batches to perform multiple epochs of updates [10]. These changes allow PPO to be applied to more general settings while matching or exceeding the performance of other, more complex methods [10].

The PPO model is not the only factor that affects the agent's offers. As all of the domains used within the environment can differ in size, the reinforcement learning approach cannot be used directly to select each bid, as PPO requires fixed input and output dimensions that do not change between negotiating sessions. Therefore, instead of choosing a bid directly, the model uses the utilities of recent offers and the progress within the negotiation session to estimate the optimal target utility of the next bid for both the agent itself and its opponent. Finally, as its bidding strategy, the agent randomly samples 1000 bids to select the one closest to the target in terms of not only its own utility but also the opponent utility estimated from opponent modeling.

To evaluate the performance of using self-play within the training stage, the environment requires the addition of a new self-play opponent. Then, the training is performed without the use of any other training opponents by exclusively performing self-play. The self-play opponent is designed in a similar way to the original agent – it uses PPO to estimate the utility goals of both parties and chooses the closest of 1000 random bids to determine the best offer to make. The agent follows the same architecture as the main agent, leading to each agent interacting with a structurally identical agent, as can be seen in Figure 1.

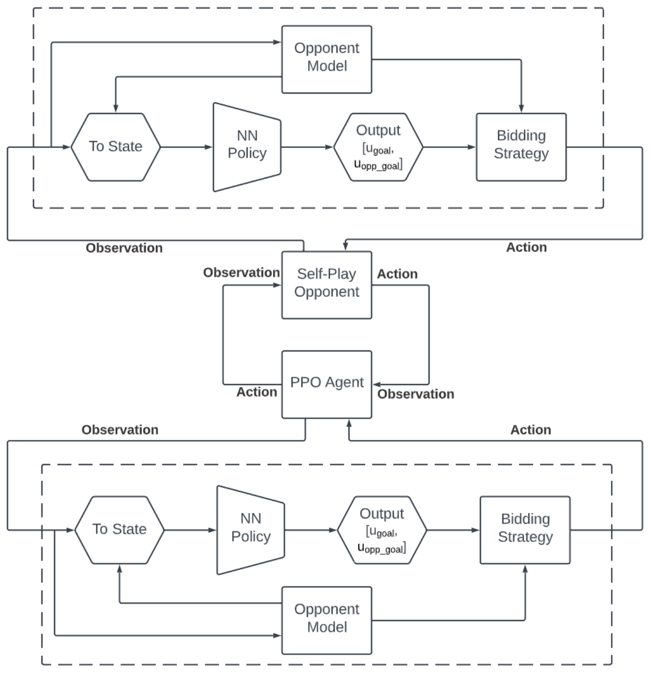

Figure 1: Schema showing the negotiation model for self-play.

The PPO model and other data used for the two agents is completely separate—the changes to one have no direct way of affecting the other. For example, as one agent learns a strategy during training, it is not necessarily the case that the opponent chooses its actions similarly. Therefore, one agent may develop an optimal strategy significantly before the other, leading to one party completely dominating the training procedure. In such cases, the training results might not transfer well to the testing environment.

4.2 Exploring optimal conditions

Another aspect that this research explores is the conditions under which the usage of self-play results in the highest utility. First off, this is explored by using multiple self-play opponents during training to lower the possibility of converging to a suboptimal solution. In addition, the usage of self-play together with a training set of opponents is explored and compared to using each technique separately.

When performing training using self-play, there is a possibility of convergence to a suboptimal solution, where one of the agents learns to always outperform the other. However, in such cases, the specific solution that performs extremely well against a self-play opponent does not necessarily transfer well to real-world opponents that apply a variety of techniques, which differ from the training environment. Therefore, it might be beneficial to perform training against more than one opponent, as competing against different opponents can help mitigate this problem by introducing additional variance during training. Because of this, various amounts of self-play opponents are compared to determine if the performance can be increased by adding additional opponents, and if so, what is the optimal amount of opponents.

Additionally, a hybrid approach is explored, where training uses a combination of self-play and a set of training opponents. In particular, this approach is compared to both the baseline and pure self-play to determine, if it is possible to achieve a better result by combining the two techniques. During training, each negotiation is randomly performed either against one of the training opponents or against the self-play opponent. In addition, the two types of opponents are selected with an equal probability, as the goal of this test is to simply determine the viability of combining the two approaches. Therefore, it is possible that the two methods can be combined more optimally, leading to further improvements in performance.

5 Results

During the testing process, each agent is directly evaluated against a set of test agents by simulating negotiation sessions. Each negotiation ends with either an agreement, where each agent receives some utility from the bid, or both parties fail to come to a consensus, leading to both receiving a utility of 0. The performance of an agent can be measured using various metrics such as its own utility, social welfare, and the difference between the two utilities. However, since the main focus of this research is to develop competitive agents, the best metric for the evaluation of an agent is its utility. Therefore, to evaluate two different agents that use reinforcement learning, the most useful metric is the opponents own utility. In addition, the social welfare metric is also briefly considered.

To have a fair and non-biased evaluation, the testing stage uses a fixed seed to choose the same predefined order of opponents and domains that help eliminate randomness between testing sessions. While testing, 50 negotiation sessions are performed. To ensure an accurate comparison, all agents are compared to a set of 10 different opponents created by students during the TU Delft course CSE3210 – Collaborative Artificial Intelligence.

In addition, each time the training is performed for a fixed amount of 2500 iterations while updating the PPO model and performing testing every 100 iterations. These parameters are fixed for all tests and have been chosen in a way, as to promote longer training sessions that converge to the optimal results for a wide variety of agents. In addition, all testing sessions are repeated 10 times to further reduce the effect of randomness. Finally, the results are analyzed by calculating the average utility and standard deviation over multiple repetitions of each evaluation.

5.1 Comparison against baseline

To evaluate the performance of self-play, it is compared to the baseline agent, which uses a set of default opponents. For self-play, a basic agent using only one self-play opponent is used.

Comparing the average utilities over 10 repetitions, self-play achieves a higher final utility of 0.704, while the baseline approach only achieves a value of 0.671. This results in self-play outperforming the conventional baseline. In addition, when using self-play, the opponent utility is significantly lower, averaging as low as 0.242, while the baseline exceeds this significantly with an opponent utility of 0.764 (Table 1).

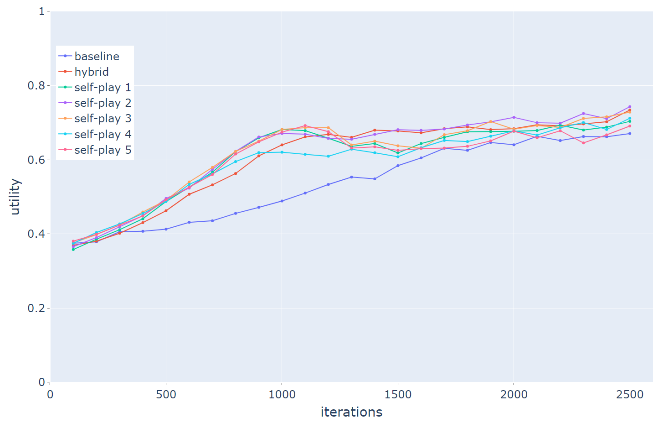

Figure 2: Comparison of the utility evolution per training type.

5.2 Multiple self-play opponents

An agent's ability to generalize to different opponents is extremely important to achieve a good result when testing against unknown opponents. However, an agent can sometimes struggle with learning a general strategy [4]. Because of this, it can be beneficial to train against a broader set of opponents to increase the variance during training.

A similar approach can also be applied to training through self-play: instead of playing against one opponent, the agent can train against multiple self-play opponents. Because of this, training using self-play has been performed with a varied amount of self-play opponents, starting with the baseline of only one opponent, and up to 5 simultaneous self-play opponents. To accommodate these changes, the frequency of policy updates and the decay of *action_std* for each self-play opponent has been applied inverse linear scaling depending on the number of opponents. Therefore, if for one opponent the policy was updated every 100 iterations, for 5 opponents this update frequency increases to once every 20 iterations. This has been done to make sure that both the agent and its self-play opponents are always at a similar stage of training.

By evaluating the performance depending on the number of self-play opponents, the average utility with multiple self-play opponents outperforms a basic self-play approach. In particular, the peak utility can be observed when using two self-play opponents. An even greater number of opponents does not seem to improve the results, and can even result in a slight decrease in performance compared to using just one opponent (Figure 3).

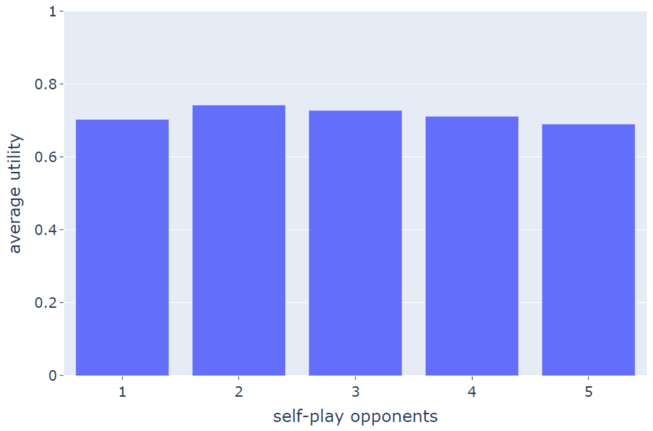

Figure 3: Average utility for various number of self-play opponents.

Training type	Average utility	Average opponent utility
Baseline	0.671 ± 0.078	0.764 ± 0.161
Self-play, 1 opponent	0.704 ± 0.047	0.242 ± 0.053
Self-play, 2 opponents	0.743 ± 0.057	0.425 ± 0.167
Self-play, 3 opponents	0.729 ± 0.046	0.363 ± 0.189
Self-play, 4 opponents	0.712 ± 0.055	0.332 ± 0.208
Self-play, 5 opponents	0.691 ± 0.042	0.273 ± 0.181
Hybrid	0.734 ± 0.037	0.445 ± 0.221

Table 1: Average utility and opponent utility for each training type.

5.3 Hybrid approach

Combining the baseline approach of utilizing a set of training opponents together with using self-play leads to an improved agent that outperforms both of the approaches. The testing finds the hybrid approach to have an average utility of 0.734, while the baseline results in a utility of only 0.671, and self-play offers a slight improvement at 0.704. As can be seen in Figure 4, the hybrid approach takes slightly longer to converge than training purely through self-play. However, once it starts to outperform self-play at around 1200 iterations, the hybrid approach continues to perform the best throughout the remaining iterations.

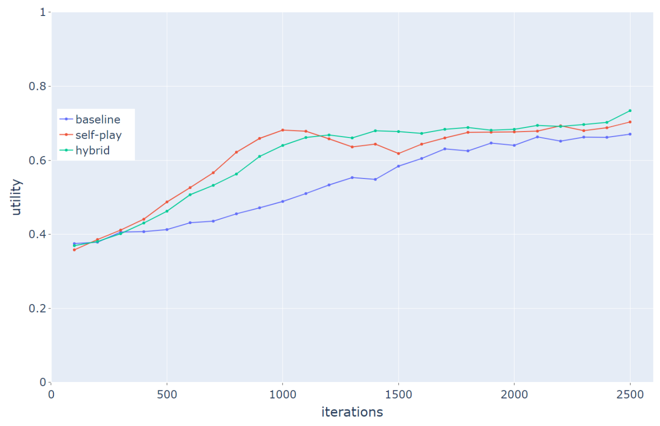

Figure 4: Utility evolution for the baseline, self-play, and hybrid training types.

6 Analysis

The evaluation of exclusively using self-play within the training of an agent has shown a slight increase in the overall utility compared to the baseline agent. The utility increase is not insignificant, reaching 0.704 with self-play while averaging at only 0.671 for the baseline agent. In addition, the variance in utility is smaller when training the agent through self-play. Because of the higher variance, the performance of the baseline model during testing cannot be predicted as consistently, leading to significant differences between multiple repetitions.

As self-play has previously been successfully used not only in settings like chess, Go, and shogi, but also in an imperfect information game like poker, it has proven to often provide performance that approaches or even exceeds state-of-the-art approaches [11; 12; 6]. The results of this research also show similar results, with self-play outperforming the baseline agent. Therefore, self-play has a high potential to be used during the training of a negotiating agent.

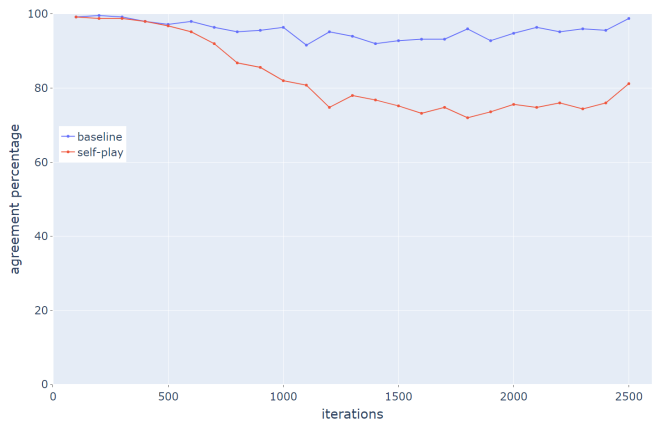

Figure 5: Percentage of agreements for the baseline and self-play agent.

As can be seen from the results, training through self-play leads to a significantly lower opponent utility. However, the increase in own utility compared to the baseline is not as pronounced. This can be explained by self-play adopting an approach similar to a hardliner agent, which is often unwilling to concede, leading to a significant amount of negotiations ending without an agreement. While the baseline agent resulted in an agreement 98.8% of the time, the agent trained through self-play reached an agreement in only 81.2% of the testing negotiations. This result shows that the self-play agent can successfully exploit its opponents but might not reach its target utility before the deadline, resulting in no agreement. However, as can be seen in Figure 5, the percentage of agreements looks to slightly increase over the last 700 iterations. Therefore, it is possible that with more training the agent could reach more agreements and further improve its utility.

The use of multiple self-play opponents during training shows an improvement in average utility – while the best overall performance is found with two self-play opponents, values from two to four lead to an increase when compared to just one opponent. However, after reaching the peak utility at two opponents, selecting any additional opponents results in a slight decrease in utility. Therefore, it can be concluded, that further increasing the self-play opponent amount does not lead to better performance. This observation can be explained by the inverse linear scaling of the policy update frequency, which is decreased for a higher amount of opponents. Therefore, the policy is updated more frequently, leading to a higher vulnerability to the effects of random factors.

The combination of using a training set of opponents together with self-play to refine the agent results in an average utility of 0.734, which is higher than using either of the approaches separately. These results show that self-play can be a great addition to the training stage of existing agents, in order to improve their performance. By combining a traditional approach with self-play, the baseline agent can be further refined by introducing additional variance in strategies encountered during training.

7 Responsible Research

In order to uphold high standards in academic integrity, an effort has been made to perform the research in a responsible manner. Responsible research dictates that all research data must be valid, reproducible, and must not be manipulated or trimmed. To adhere to these principles, all results collected as part of the research have been made public as part of a repository[1]. In addition, all raw results also include notes regarding the parameters and conditions under which they were collected and how they are used. All collected results have been utilized, and no manipulations have been performed in order to avoid bias within the research. Reproducibility is further improved by performing all testing against a predefined order of opponents and domains to eliminate most of the randomness. The training stage has also been modified in such a way, where a significant amount of randomness can be eliminated by providing a set seed, which is constant for all collected data. Together, all of these measures ensure that the results have been obtained responsibly and are reproducible.

[1]The raw results obtained during the research can be accessed at https://github.com/brenting/negotiation_PPO/tree/testing-self-play

In addition, the research practices must adhere to the ethos of science and must be wary of plagiarism and conflicts of interest. To follow best scientific practices, all sources viewed during the research have been documented and all citations or quotes are clearly and accurately accompanied by a reference to the original source. In addition, all of the research has been completed by the author with assistance from the supervisors. None of the parties involved in the completion of the research have any conflicts of interest and have not been influenced by outside parties.

Additionally, the training and testing of the agents use opponents created by students of the TU Delft course CSE3210 (Collaborative Artificial Intelligence). Although these agents were originally created as part of an assignment, the students were informed that their submissions would be made publicly available. Because of this, the research is permitted to make use of the agents for training and evaluation, and does not make any claims regarding their ownership.

8 Conclusions and Future Work

Traditionally, the training of a negotiating agent requires a set of default opponents. However, it can be difficult to find a representative set of opponents that offers good performance and a high variance in strategy types. Since an agent must learn to perform well against a variety of opponents to achieve good results, a subpar training set can significantly limit the performance of an agent. However, it is also possible to perform training without requiring any opponents – the agent can be put against itself to perform self-play.

This paper has investigated the feasibility of using self-play in the training of a negotiating agent by answering the research question: *How well can we train a negotiating agent without a default set of opponents through self-play?* The results have shown the exclusive usage of self-play during training to be viable and lead to comparable, or sometimes even better results than the baseline agent, which utilizes a set of training opponents. These results share similarities with previous research on the usage of self-play in other settings, like chess, Go, and shogi [11; 12].

Furthermore, self-play results in a slightly higher utility when introducing additional variance in training by playing against multiple self-play opponents. The usage of multiple self-play opponents during training shows a slight improvement over an agent that trains against only one self-play opponent. In particular, the best average utility is achieved by having two self-play opponents. Any further increase in the opponent count leads to a slight decrease in utility, as more opponents result in more frequent updates to the underlying policy.

As the usage of self-play has shown good results, it can further be explored in more refined agents that utilize advanced techniques for selecting bids and modeling opponents. In addition, the results of using self-play in combination to a more conventional approach, which utilizes a set of training opponents, show that the addition of a self-play stage together with an existing training setup can further improve the agent to achieve a higher utility. Because of this, research can further explore the usage of self-play to refine existing agents that only train against a set of training opponents. In particular, this technique could be explored in negotiation competitions, where a very extensive and diverse training using self-play could be simulated using a large amount of computational power. Additionally, the usage of self-play together with a traditional approach in a hybrid training session could be further explored in more detail to investigate the different ways of combining the two approaches.

References

[1] Reyhan Aydoğan, David Festen, Koen V. Hindriks, and Catholijn M. Jonker. Alternating offers protocols for multilateral negotiation. *Modern Approaches to Agent-based Complex Automated Negotiation*, page 153–167, 2017.

[2] Tim Baarslag, Reyhan Aydoğan, Koen V. Hindriks, Katsuhide Fujita, Takayuki Ito, and Catholijn M. Jonker. The automated negotiating agents competition, 2010–2015. *AI Magazine*, 36(4):115–118, 2015.

[3] Tim Baarslag, Katsuhide Fujita, Enrico H. Gerding, Koen Hindriks, Takayuki Ito, Nicholas R. Jennings, Catholijn Jonker, Sarit Kraus, Raz Lin, Valentin Robu, and et al. Evaluating practical negotiating agents: Results and analysis of the 2011 international competition. *Artificial Intelligence*, 198:73–103, 2013.

[4] Jasper Bakker, Aron Hammond, Daan Bloembergen, and Tim Baarslag. Rlboa: A modular reinforcement learning framework for autonomous negotiating agents. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '19, page 260–268, Richland, SC, 2019. International Foundation for Autonomous Agents and Multiagent Systems.

[5] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, pages 1282–1289. PMLR, 2019.

[6] Johannes Heinrich and David Silver. Deep reinforcement learning from self-play in imperfect-information games. *CoRR*, abs/1603.01121, 2016.

[7] Chun-Che Huang, Wen-Yau Liang, Yu-Hsin Lai, and Yin-Chen Lin. The agent-based negotiation process for b2c e-commerce. *Expert Systems with Applications*, 37(1):348–359, 2010.

[8] Shanqi Liu, Junjie Cao, Yujie Wang, Wenzhou Chen, and Yong Liu. Self-play reinforcement learning with comprehensive critic in computer games. *Neurocomputing*, 449:207–213, 2021.

[9] Fernando Lopes, Tiago Rodrigues, and Jorge Sousa. Negotiating bilateral contracts in a multi-agent electricity market: A case study. *2012 23rd International Workshop on Database and Expert Systems Applications*, 2012.

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

[11] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm, 2017.

[12] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, and et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359, 2017.