Learning a Latent Representation of the Opponent in Automated Negotiation

Radu Gaghi
Supervisors: P.K. Murukannaiah, B.M. Renting
EEMCS, Delft University of Technology, The Netherlands

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

This paper introduces a strategy for learning opponent parameters in automated negotiation and using them for future negotiation sessions. The goal is to maximize the agent's utility while being consistent in its performance over various negotiation scenarios. While a number of reinforcement learning approaches in the field have used Q-learning, this paper uses the newer Proximal Policy Optimization algorithm. Machine learning has been used in opponent modeling, classifying opponents, and learning strategies, but there have been few attempts to store and re-use this information. In an experimental setup, it is shown that this approach outperforms a baseline in terms of individual utility.

1 Introduction

Negotiation is an inherent process in every human life. It can be defined broadly as a discussion aimed at reaching an agreement. Although negotiation can bring significant benefits, more than half of American employees do not negotiate their salaries [3]. What if it could be done automatically on your behalf? It is only natural that in the past few decades there has been a growing need to delegate the negotiation process to autonomous agents.

As technology plays an increasingly important role in our lives, the importance of reliable automated negotiation is also growing. Online dispute resolution, internet-of-things, e-commerce, or resource management [2], to name a few, could all benefit from developments in this field. Advances in automated negotiation promise to save time, reduce costs and lead to better outcomes [1].

In the same way that humans do, agents should be able to learn from their negotiating sessions to improve themselves continuously. Agents that do not adapt can be easily exploited by other agents given enough negotiating sessions [18] and adaptation usually results in improved performance.

The use of machine learning in this field has seen growing interest in recent years and techniques such as Bayesian learning, genetic algorithms [13] or Q-learning [14] have all been used. Some agents can classify the opponent's bidding strategy and adjust their own strategy accordingly [16], while other agents can estimate the opponents' preferences and exploit them [5]. A party (agent) that performs well against one opponent might perform poorly against another and the same can be said for domains. An agent that is able to learn could help face these issues by being able to generalize on both opponents and domains.

Although agents today can effectively learn and improve, there is still a long way to go until they can represent us in day-to-day life. It would require agents to be largely autonomous and have excellent knowledge regarding the domain and the preferences of the person it represents, as well as be able to adapt to each new opponent.

The research question this paper aims to answer is: *Can an agent learn a latent representation of an opponent and use that representation to improve its performance?*

An opponent model or "latent representation" is defined simply as knowledge about aspects of an agent, which are hidden or not available, meaning they must be inferred from the agent's actions. In the context of this research, a latent representation consists of the bidding strategy of the opponent, however other opponent modeling techniques exist [7]. An improvement is defined as an increase in the metrics used to evaluate the agents, such as average utility gained, percentage of agreements, and distance from the Pareto frontier.

Reinforcement learning (RL) is a machine learning approach in which agents try to maximize their reward while interacting with their environment. This way, agents can learn from their experiences in a bias-free way, compared to solutions proposed by humans. In this case, the environment is the negotiation taking place between the agent and its opponent, while the actions it can take are the offers sent and the decision to accept or reject, which are directly influenced by the policy. More specifically, Proximal Policy Optimization (PPO) is used, which is a state-of-the-art RL algorithm that is easy to implement, sample efficient, and easy to tune[15].

The goal of our work is to increase the agent's utility using machine learning in order to extract information about the opponent, which can be stored for later negotiation sessions. To the knowledge of the authors, PPO has not yet been used in this field, and storing information about past opponents, then reusing it is an approach yet unexplored. A new, simple solution is proposed for learning how the opponent concedes, building on the work of Yasumura et al. [19].

2 Related work

Opponent modeling is a topic frequently studied in the field of automated negotiation. There exists a large number of studies concerning the main approaches to learning about the opponent: modeling the opponent's strategy or the opponent's preferences. Modeling the opponent's strategy by learning certain parameters is the focus of this paper.

The aforementioned approach is also split into two main areas of focus, with some overlap: acceptance strategy modeling and bidding strategy modeling. Razeghi et al. [14] use a deep Q-network (DQN) to learn when to accept. They use the utility of the received bid and the estimated opponent utility, as well as the remaining time and target utility to determine the acceptance or rejection probability.

In contrast, Bakker et al. [10] have introduced a reinforcement learning framework using Q-learning for automated agents, focusing on the bidding strategy. They discretize the utility space into several bins and using the utility of previous bids, they estimate in which bin the utility of the next bid should be.

The main approaches mentioned earlier can also overlap, such as in the work of Bagga, Paoletti and Stathis [9]. They estimate parameters using statistical information derived from received bids, which are then used to select bidding and acceptance strategies that can change throughout the session. They pre-train their model using supervised learning (SL) using examples from previous negotiation sessions. The strategy is then improved using reinforcement learning.

Finally, Yasumura et al. [19] propose another Q-learning approach where the agent estimates how much it should concede, based on how much the opponent and itself have already conceded. The agent outputs the amount of concession for the next bid to be sent and the reward function is similar to the one used in this paper, specifically the utility at the end of the agreement.

3 Methodology

The agents used for training and testing, as well as the main agent in this paper, have been written using the GENIUS FRAMEWORK [12]. Negotiation sessions will be bilateral, and communication within a session will use the Stacked Alternating Offers Protocol (SAOP) [4]. Given a set of issues and values, a negotiation domain is a set containing the available bids, which are combinations of the issues and values. A preference profile specifies the utility of each value, and a utility function can compute the utility of any bid. This paper uses linear additive utility functions, meaning the preference for each issue is independent of what values other issues can take. The domains and preference profiles are generated automatically using a script and are split into training and test sets. The domain size is chosen at random, but must be between 200 and 10000 bids. The number of issues is also chosen at random and must be between 4 and 10 issues. The number of values per issue is partly random, but it is also dependent on the number of issues. Using a dictionary containing the set of issues and values, two profiles are created, also in a random fashion.

In order to answer the research question, the experiment is split into two phases, training and testing. Agent training is accomplished by online reinforcement learning using PPO. Data is collected throughout each negotiation session, and the policy updates its weights after a certain number of episodes. One episode consists of one negotiation session. Initially, the agent will be trained against multiple opponents, where the goal is to learn a particular feature of the opponent. Then, to test it, the agent will negotiate with another set of agents, and the results will be compared with 2 agents, one using reinforcement learning (referred to as the basic PPO agent) and one not using it (referred to as the baseline). They will negotiate against the same agents on the same domains. In this case, for every session during training, a random agent and a random domain are selected and this process is repeated until the allocated time budget expires. Agents are tested by negotiating against each opponent 50 times on a separate set of domains. The same random seed is used so that there is less variability and the results are more comparable. Average utility, percentage of agreements, and the average distance from the Nash point will be used in evaluating the agents.

$$\omega \in \Omega, p_{nash} \in argmax(U_{own}(\omega) * U_{opp}(\omega)) \quad (1)$$

Expression 1 shows how to compute the Nash product, where ω is a bid from the domain Ω, and p_{nash} is the Nash point, U_{own} is the agent's utility functions while U_{opp} is the utility function of the opponent.

4 Experimental Setup

4.1 Implementation

The most important implementation details will be covered in this section, while specific values for parameters or minor details are available on GitHub.

The most important method of the agent is $select\_action(observation)$. This receives an offer from the other party and decides if the agent should accept or reject and send a counter-offer. For the sake of simplicity, the acceptance strategy used is AC_{next}, which accepts an offer if the bid to be sent has lower utility than that received by the opponent. The bidding strategy is determined by the policy, using the state vector as input. The state is composed of 3 features: the opponent concession parameter, the concession balance, and the time until the deadline (the progress). The concession parameter is initialized to zero if no previous negotiations against the opposing party have taken place, or it is read from a file otherwise. The concession balance is the difference between how much the agent has conceded from the initial offer and how much the opponent has conceded from their initial offer, as shown in Equations 2, 3, and 4, where ω represents a bid, n is the number of the round, and m is the final round number.

$$Balance = \sum_{n=0}^{m}(Concession_{own}^{n} - Concession_{opp}^{n}) \quad (2)$$

$$Concession_{own} = U_{own}(\omega_{previous}) - U_{own}(\omega_{latest}) \quad (3)$$

$$Concession_{opp} = U_{opp}(\omega_{previous}) - U_{opp}(\omega_{latest}) \quad (4)$$

Using the state vector, the policy has 2 outputs: the updated opponent concession parameter and a utility goal. The latter is then used to find a random bid above that utility goal, from a sorted list containing all of the possible bids. At the end of the negotiation session, the opponent concession parameter is stored in a file named after the opponent it faced so that it can be used in future sessions against the same opponent. As the goal of the agent is to maximize individual utility, the reward given to the PPO policy at the end of an episode is the individual utility of the bid accepted at the end of the session or 0 in the case where the time runs out or no deal is made.

4.2 Experiment

The agent is trained for 1 hour (approximately 1500 episodes) against 17 agents on various domains. More details about how this is done can be found in Section 3. The deadline for one negotiation session is 10 seconds or until an agreement is reached (whichever comes first). After that, a new session begins, and so on.

There are 9 parties against which the agent is tested. Fifty negotiation sessions are done against each of them separately, and then metrics are computed. The different metrics are then compared to see if there is an improvement on average or specific opposing parties. The same is done for the other 2 agents: the baseline agent and the basic PPO agent. The former is a time-based conceder using AC_{next} for its acceptance

strategy. The latter (also trained for 1 hour), uses the utility of the past 3 received bids as input for the policy to determine an individual utility goal and one for the opponent. In order to find a bid, it samples 1000 random bids and chooses the one closest to both utility goals. It accepts when the received offer has higher utility compared to the individual utility goal.

5 Results

Figure 1 shows the average utility gained against each opponent by the agent (in blue), the baseline (in red), and the basic PPO agent using reinforcement learning (in yellow). In 6 out of 9 cases, the agent outperforms the baseline and it always outperforms the basic PPO agent. In the majority of the cases where it outperforms the baseline, it does so by around 10%, but there is some variance. In the 3 cases where the baseline performs better than the agent, the percentage of agreements is the main cause, being as low as 74% for the agent and 94% for the baseline when negotiating against Agent 67. As mentioned before, there is no reservation value for any of the agents, a session that does not result in an agreement awards 0 utility. Complete results can be seen in Tables 1 and 2.

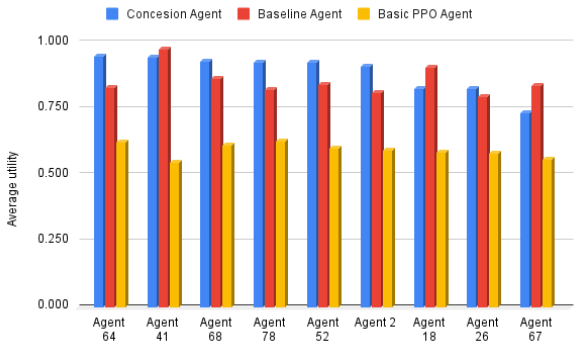


Figure 1: Comparison of average utility

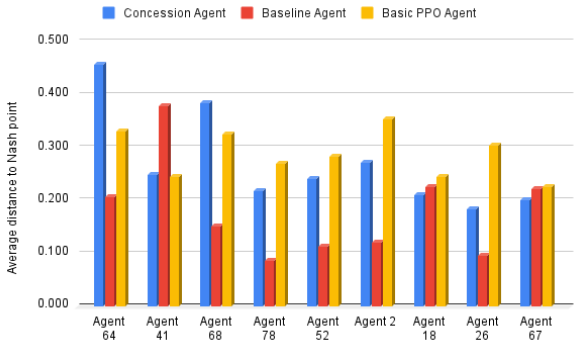


Figure 2: Comparison of average distance to the Nash point

6 Analysis

One important assumption made in this work is that opponents the agent has negotiated against will be faced again in the future. It would be beneficial if the agent's performance would increase after repeated negotiations against the same opponent. However, with the exception of a few outliers, the opponent concession parameter does not change significantly between sessions, as Figure 3 shows. In addition, the agent does not improve its performance after several sessions against the same opponent, meaning the value of the opponent concession parameter has converged within one session.

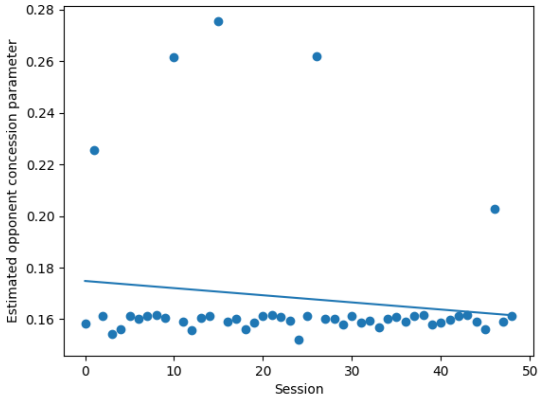


Figure 3: Evolution of the opponent concession factor at the end of the negotiation throughout 50 sessions

In Figure 4 the value of the concession parameter is plotted after each exchange of bids throughout 5 sessions, against different opponents from the test set. It can be seen that the value does not remain constant, but it decreases as the session progresses. This suggests that how the opponent concedes is not the only feature modeled by the neural network. Due to the black-box nature of the policy and the reward which only promotes high utility through any means, it is hard to say what exactly the policy is outputting in order to somehow classify the opponent. It is also dependent on the concession balance and the time until the deadline. The balance is itself dependent on the estimated utility of the opponent. The Smith frequency model is used, which has an accuracy of around 75% [6]. The performance of the agent with regard to different accuracy of opponent models has not been investigated, however, this could be the subject of future work.

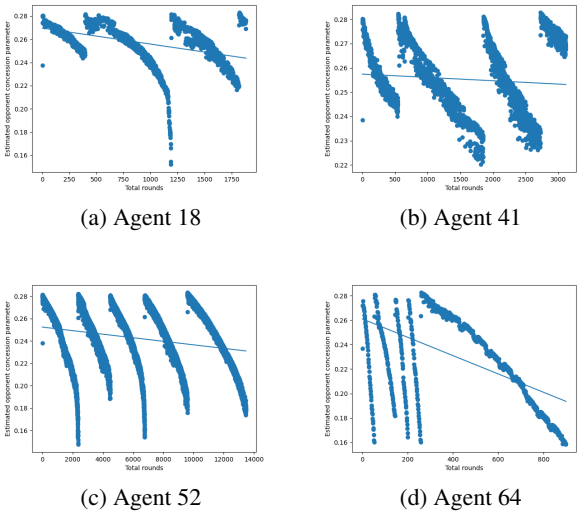
While the agent outperforms the baseline against most opponents, it results in more no-agreement outcomes. It is likely that it will perform worse than most agents when faced with hardliner-type agents that do not concede by the end of the session, as zero utility is worse than low utility. Usually, the agent does not concede by a large margin, so the baseline reaches outcomes that are closer to the Nash point against 6 out of 9 opponents. Even though the agent was not trained to reach win-win outcomes, it is interesting to see it has learned that being selfish, will usually yield a high individual utility. An unexpected and equally interesting finding was that the agent will begin to concede by a small margin (around 5%, but it depends on how much the opponent concedes as well)

Concession Agent						Baseline Agent				
Opponent	**Average utility**	**Variance**	**Average social welfare**	**% of agreements**	**Mean distance to Nash point**	**Average utility**	**Variance**	**Average social welfare**	**% of agreements**	**Mean distance to Nash point**
Agent 64	0.952	0.024	1.365	100.00%	0.457	0.832	0.007	1.510	100.00%	0.208
Agent 41	0.945	0.026	1.576	100.00%	0.249	0.976	0.001	1.478	100.00%	0.380
Agent 68	0.931	0.136	1.419	98.00%	0.385	0.864	0.005	1.583	100.00%	0.153
Agent 78	0.928	0.032	1.596	100.00%	0.219	0.824	0.004	1.631	100.00%	0.087
Agent 52	0.926	0.026	1.562	100.00%	0.242	0.843	0.004	1.617	100.00%	0.115
Agent 2	0.911	0.133	1.499	98.00%	0.273	0.813	0.005	1.601	100.00%	0.121
Agent 18	0.827	0.307	1.420	88.00%	0.211	0.906	0.021	1.536	98.00%	0.226
Agent 26	0.827	0.278	1.455	90.00%	0.184	0.799	0.018	1.593	98.00%	0.097
Agent 67	0.734	0.391	1.270	78.00%	0.201	0.838	0.051	1.454	94.00%	0.223

Table 1: Comparison of Concession Agent vs Baseline Agent

Concession Agent						Basic PPO Agent				
Opponent	**Average utility**	**Variance**	**Average social welfare**	**% of agreements**	**Mean distance to Nash point**	**Average utility**	**Variance**	**Average social welfare**	**% of agreements**	**Mean distance to Nash point**
Agent 64	0.952	0.024	1.365	100.00%	0.457	0.626	0.015	1.416	100.00%	0.332
Agent 41	0.945	0.026	1.576	100.00%	0.249	0.548	0.053	1.359	88.00%	0.245
Agent 68	0.931	0.136	1.419	98.00%	0.385	0.615	0.015	1.422	100.00%	0.325
Agent 78	0.928	0.032	1.596	100.00%	0.219	0.628	0.017	1.500	100.00%	0.271
Agent 52	0.926	0.026	1.562	100.00%	0.242	0.602	0.023	1.443	98.00%	0.283
Agent 2	0.911	0.133	1.499	98.00%	0.273	0.593	0.018	1.399	100.00%	0.355
Agent 18	0.827	0.307	1.420	88.00%	0.211	0.587	0.045	1.423	92.00%	0.245
Agent 26	0.827	0.278	1.455	90.00%	0.184	0.582	0.021	1.396	98.00%	0.305
Agent 67	0.734	0.391	1.270	78.00%	0.201	0.560	0.056	1.396	88.00%	0.227

Table 2: Comparison of Concession Agent vs Basic PPO Agent



(a) Agent 18 (b) Agent 41

(c) Agent 52 (d) Agent 64

Figure 4: Evolution of opponent concession factor against different opponents throughout 5 sessions

if the opponent begins to concede. However. when faced with an opponent that does not concede, such as a hardliner agent, it will not concede itself. This may point to the fact that the agent is sensitive to the concession balance feature of the state vector, and is reacting in a tit-for-tat manner.

From the results, one can see that the basic PPO agent performs worse than the agent and the baseline. One possible explanation could be the small number of random bids it samples (1000). Some domains have more than 10000 possible bids, so it can be that the utility of the bid it sends is much lower than the utility goal that the policy outputs. The decision to keep it as it is was made so that there is a common baseline with the authors' peers.

7 Responsible Research

7.1 Ethical aspects

Discussions about the ethical aspects of AI tend to revolve around the negative impact artificial intelligence can have on human lives. However, AI is a rapidly growing industry with billions of dollars invested annually by companies [17], promising to bring many ethical benefits to society in the coming years. More specifically, autonomous agents can represent humans in salary negotiations, helping bridge the wage gap between men and women and helping reduce income inequality by providing trustworthy and improved performance compared to the average human negotiation tactics. Negotiating agents could also improve the way we distribute resources and increase efficiency or reduce shortages in the global economy [2]. Problems that we are currently struggling to solve could be easily tackled by bias-free AI in the coming years, provided there is enough trust to adopt these new technologies. Ideally, trust is directly proportional to the performance

of these agents, which shows that first there needs to be further development in the field before a large-scale adoption for complex tasks such as economic planning is possible [8].

Nonetheless, there are negative ethical aspects to consider as well. Backdoor attacks on neural networks [11] can fool even well-trained agents. This type of attack consists of "poisoned" data being fed into the network, essentially leading to classification errors. This data has a special trigger, that can be later exploited by the attacker to influence the decision-making process of the policy. In the context of automated negotiation, this trigger could lead an agent to accept a bad outcome for itself, for the benefit of the opponent. Whether it is the fault of bad actors or errors in the system, automated negotiation could also lead to inequality. Say, for example, people could negotiate their electricity prices with the energy company using autonomous agents. Knowledgeable people, which could have had access to higher education could create better agents that get them advantageous prices, while less knowledgeable people would get worse prices. The same can be said for rich and poor people, the former being able to afford better agents, while the latter cannot, further exacerbating the gap between the upper and lower classes. Nonetheless, it is up to us to use technology for our own betterment, or for our own decline.

7.2 Reproducibility

The repository containing the implementation of the PPO algorithm and the agents will be made public. To reproduce the results from this paper, all one needs to do is download the repository and run the train and test files on the same domains and against the same agents as specified in the methodology and experiment sections.

8 Conclusions and Future Work

This paper has set out to answer the following question: *Can an agent learn a latent representation of an opponent and use that representation to improve its performance?* PPO was used to learn and store an opponent parameter in a bilateral negotiation. It uses the concession balance, the progress, and the previously learned opponent parameter to output a utility goal and update said parameter.

Trained and tested over multiple domains and against multiple opponents, this research has shown that by learning how the opponent concedes, the agent outperforms the baseline in terms of individual utility in the majority of cases. The value modeled by the agent is learned within one negotiation session, so there is no need for multiple sessions to reach its maximum performance against an opponent. The policy has become sensitive to how the opponent concedes and it will employ a tit-for-tat strategy, without conceding too much. This enables it to take advantage of the fact that opponents are more likely to accept worse offers in the end, while still receiving a high individual utility.

Future work can be done in creating a more sophisticated state vector in order to have noticeable improvement after more negotiation sessions. As mentioned before, the state vector depends on the accuracy of the opponent model, so research into how this affects the performance of the agent could prove useful. In addition, a recurrent neural network could be used, that might be better suited for this problem.

References

[1] Predicting opponent's moves in electronic negotiations using neural networks, February 2008. [Online; accessed 2. Jun. 2022].

[2] Automated negotiation in environmental resource management: Review and assessment, October 2015. [Online; accessed 7. Jun. 2022].

[3] Consumer Reports' Survey: Nine of 10 Americans Who Haggled Saved Money, June 2022. [Online; accessed 2. Jun. 2022].

[4] Reyhan Aydogan, David Festen, Koen V. Hindriks, and Catholijn M. Jonker. Alternating Offers Protocols for Multilateral Negotiation. In *Modern Approaches to Agent-based Complex Automated Negotiation*, pages 153–167. Springer, Berlin, Germany, 2017.

[5] Tim Baarslag, Katsuhide Fujita, Enrico H. Gerding, Koen Hindriks, and Colin R. Williams. Evaluating Practical Negotiating Agents: Results and Analysis of the 2011 International Competition. *Artificial Intelligence*, 198:73–103, May 2013.

[6] Tim Baarslag, Mark Hendrikx, Koen Hindriks, and Catholijn Jonker. Measuring the Performance of Online Opponent Models in Automated Bilateral Negotiation. In *AI 2012: Advances in Artificial Intelligence*, pages 1–14. Springer, Berlin, Germany, 2012.

[7] Tim Baarslag, Mark J. C. Hendrikx, Koen V. Hindriks, and Catholijn M. Jonker. Learning about the opponent in automated bilateral negotiation: a comprehensive survey of opponent modeling techniques. *Autonomous Agents and Multi-Agent Systems*, 30(5):849–898, September 2016.

[8] Tim Baarslag, Michael Kaisers, Enrico H. Gerding, Catholijn M. Jonker, and Jonathan Gratch. When Will Negotiation Agents Be Able to Represent Us? The Challenges and Opportunities for Autonomous Negotiators, 2017. [Online; accessed 8. Jun. 2022].

[9] Pallavi Bagga, Nicola Paoletti, and Kostas Stathis. Deep Learnable Strategy Templates for Multi-Issue Bilateral Negotiation. *ArXiv e-prints*, January 2022.

[10] J. Bakker, A. Hammond, D. Bloembergen, and T. Baarslag. RLBOA: A modular reinforcement learning framework for autonomous negotiating agents, May 2019. [Online; accessed 24. May 2022].

[11] Stefanos Koffas. Backdoor Attacks in Neural Networks, 2021. [Online; accessed 8. Jun. 2022].

[12] Raz Lin, Sarit Kraus, Tim Baarslag, Dmytro Tykhonov, Koen Hindriks, and Catholijn M. Jonker. GENIUS: AN INTEGRATED ENVIRONMENT FOR SUPPORTING THE DESIGN OF GENERIC AUTOMATED NEGOTIATORS. *Computational Intelligence*, 30(1):48–70, February 2014.

[13] Jim R. Oliver. A Machine-Learning Approach to Automated Negotiation and Prospects for Electronic Commerce. *Journal of Management Information Systems*, 13(3):83–112, December 1996.

[14] Yousef Razeghi, Ozan Yavuz, and R. Aydogan. Deep reinforcement learning for acceptance strategy in bilateral negotiations. *Turkish Journal of Electrical Engineering and Computer Sciences*, 28(4), 2020.

[15] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *ArXiv e-prints*, July 2017.

[16] Ayan Sengupta, Yasser Mohammad, and Shinji Nakadai. An Autonomous Negotiating Agent Framework with Reinforcement Learning Based Strategies and Adaptive Strategy Switching Mechanism. *ArXiv e-prints*, February 2021.

[17] Ubs. AI's coming of age. *evolution of artificial intelligence*, May 2022.

[18] D. D. B. van Bragt and J. A. La Poutré. Why Agents for Automated Negotiations Should Be Adaptive. *Netnomics*, 5(2):101–118, November 2003.

[19] Yoshiaki Yasumura, Takahiko Kamiryo, Shohei Yoshikawa, and Kuniaki Uehara. Acquisition of a concession strategy in multi-issue negotiation. *Web Intelligence and Agent Systems*, 7(2), January 2009.