Understanding the Evolution of Cooperation in Societies of Artificial Agents

UCL 2021-22 COMP0031 Group 10: Jeremy Lo Ying Ping, Anirudh Lakra, Zihan Zhu, Zeyu Zhou, Ruochen Sun and Tianang Chen. Supervised by Prof Mirco Musolesi.

***Abstract*—Cooperation lies at the heart of human lives and society. Understanding its evolution could unlock new insights and possibilities, potentially leading to exciting discoveries across the social and biological sciences. At the same time, the field of artificial intelligence has been rapidly making progress, finding novel and interesting applications for machine learning across a wide variety of domains.**

In this paper, we review how studying societies of artificial agents has contributed towards our comprehension of the dynamics of social dilemmas and explore how Q-learning – a type of reinforcement learning – can discover cooperative strategies for playing the two-player iterated prisoner's dilemma game.

We investigate the impact of different hyperparameters and training opponents on the performance of Q-learning agents in the competitive setting of a round-robin tournament against other rule-based agents found in the literature and show that tabular Q-learning agents are capable of achieving similar scores and tournament ranks to well-performing rule-based agents. Finally, we discuss a few limitations of our approach and point to interesting areas for future research.

***Index Terms*—evolution of cooperation, iterated prisoner's dilemma, round-robin tournaments, Q-learning, reinforcement learning, societies of artificial agents**

I. Introduction

THE fundamental challenge behind understanding the evolution of cooperation is the determination of the circumstances under which self-interested competitors become incentivised to cooperate with each other so as to maximise their mutual payoff [1]. While the pursuit of one's interests can benefit society more widely as a whole, oftentimes, the interests of individuals and the collective conflict, thereby posing a social dilemma for everyone involved.

With cooperation necessary for the development of complex organisational hierarchies, the ubiquity and relevance of cooperation problems cannot be understated. In the natural world, they are present at all scales from genes and chromosomes to complex multi-cellular organisms and entire ecosystems [2]. In economics and international relations, social dilemmas manifest in situations involving oligopolistic competition in markets [3], the free-riding of public goods, international trade disputes, nuclear non-proliferation and so on [4]. Finally, they can occur elsewhere in the context of self-driving vehicles and the response to major societal issues, such as the Covid-19 pandemic and climate change [5]. As a result, gaining insights into mechanisms that spur the emergence of cooperation, such as reciprocal altruism [6] and kinship [2], is of particular interest in the social and biological sciences.

The study of societies of artificial agents – algorithms that act in response to their environment [7] – provides one avenue for further investigating the nature of cooperation. In particular, the iterated prisoner's dilemma (IPD) game neatly encapsulates the notion of a social dilemma: while the game-theoretic solution would be for the players not to cooperate (i.e. to defect) given the risk of being exploited if the cooperation is unreciprocated, players can work together – especially after repeated encounters – in order to maximise their return over the course of the game [4].

Furthermore, tournaments, in which these artificial agents play the IPD game against each other, create a controlled, emulated environment where there is scope for cooperation to develop over repeated interactions while also serving as a useful tool for ranking the effectiveness of different strategies at maximising their return.

Since Axelrod catalysed the field by inviting leading game-theoreticians to submit rule-based agents to compete in his eponymous tournaments [3], [8] – afterwards publishing his seminal paper [1] and subsequent book [4] of the same name, *The Evolution of Cooperation* – a lot of work has gone into studying the cooperative behaviour of rule-based and more recently learning-based agents as they relate to the field of cooperative artificial intelligence [5], which we outline in Section II.

Continuing in this tradition, in this paper, we explore the application of Q-learning [9] – a form of reinforcement machine learning (RL) – in training agents to play the IPD game and present our findings from running a series of our own round-robin IPD tournaments pitting these learning-based agents up against rule-based agents found in the literature.

In particular, we investigate the impact of various hyperparameters, such as the rate of exploration during training, as well as the selection of rule-based opponents trained against, on a learning-based agent's performance in the tournament setting, both in terms of mean score and rank according to the methodology set out in Section III.

While we primarily focus on tabular Q-learning agents, which perform the best in our experience, we also discuss a few experiments training and evaluating deep Q-learning agents [10] and deep recurrent Q-learning agents using a gated recurrent unit (GRU) as part of their Q-network on which we elaborate in Section III-B.

In Section IV-C, we demonstrate the feasibility of tabular Q-learning agents discovering interesting, cooperative policies in order to perform reasonably well in tournaments, often scoring comparably with top-performing rule-based agents. We then comment on a few limitations of our approach in Section V.

Finally, we conclude this report in Section VI, summarising

our findings and suggesting interesting directions for future research to explore this domain further.

II. Related Work

A. Early Study of Cooperation

The emergence of cooperation in the natural world has perplexed evolutionary biologists for generations due to the challenge it poses to Charles Darwin's Theory of Natural Selection. With the advent of John von Neumann's landmark paper "On the Theory of Games of Strategy," [11], the problem of cooperation became embedded into Game Theory and has attracted scholars from a diverse range of disciplines. Since then, research into the emergence of cooperation has exploded, generally leveraging social dilemmas, such as the Prisoner's Dilemma (PD), to study cooperation between competing agents. Inspired by the study of this example, many theories about cooperation have been proposed: W.D. Hamilton revealed true altruism between interacting individuals with close relatedness [12], [13]; and Robert Trivers demonstrated how reciprocal altruism evolves between unrelated individuals, even between individuals of different species [14]. The key is that in the IPD, both parties can benefit from the exchange of many seemingly altruistic acts.

B. Axelrod's Tournament in 1979

In 1979, to figure out what strategies thrive in a sophisticated multi-agent environment, Axelrod held a round-robin tournament known as Axelrod's first tournament [3]. To Axelrod's surprise, the winning strategy was `TitForTat` (`TFT`), a rather simple strategy relative to others. The mechanism underlying `TFT` is simple: `TFT` cooperates on the first move, then reciprocates whatever its adversary performed on its previous move.

In Axelrod's paper published in 1981 [1], he analysed the success of `TFT` via three standards: robustness, stability and initial viability. He proposed two theories on how pairwise collaborative interactions can occur between agents, which are genetic kinship theory and reciprocation theory, respectively. Moreover, Nowak expanded upon the work done by Axelrod to produce a comprehensive classification in terms of cooperation mechanisms [2]: 1. Kin Selection, 2. Direct Reciprocity, 3. Indirection Reciprocity, 4. Network Reciprocity, 5. Group Selection.

C. The Emergence of Other Rule-based Agents

After Axelrod's first tournament, researchers from various disciplines devoted themselves to finding strategies that outperform `TFT`. One well-known attempt is the strategy proposed by Nowak and Sigmund [15], `Pavlov`, based on win-stay and lose-shift. Pavlov optimised for two shortcomings of `TFT`:

1) Mistake Correction: it corrects occasional mistakes that lead to an indefinite chain of mutual defections between `TFT` players.
2) Exploiting the suckers: `TFT` will always cooperate with suckers, such as `AllC`, which is not the best strategy; rather, `AllD` is optimal versus `AllC`.

Most of the other successful strategies are the mutants of `TFT` and `Pavlov` (e.g. `OmegaTitForTat` and `AdaptivePavlov`). Interestingly, [3] also mentions one class of agents known as *kingmakers*, which do not perform very well on their own but drastically impact the rankings of the agents depending on how well they synergise with the kingmaker.

D. MARL agents

Recently, experts have been leveraging reinforcement learning (RL) methods to train agents and study their behaviour in multi-agent environments. The mainstream algorithms are either value-based (e.g. Q-learning [16]), policy-based (e.g. policy-gradient [17]) or a combination of both (e.g. actor-critic [18] and A3C [19]). These complex algorithms are extremely convenient in aiding researchers in gaining insights into the evolution of cooperation. For example, Harper et al. [20] showed that trained strategies can potentially perform better than a variety of human-made ones in the IPD. Moreover, Anastastacos et al. [21] show that agents trained by using deep Q-learning will develop cooperative behaviour in an IPD and promote pro-social behaviour in a multi-agent society if they can select their partners. Leibo et al. [22] reveal that the policies deep Q-network (DQN) agents learnt in complex sequential social dilemmas are influenced by the environment, the intrinsic properties of agents and the model design, and motivate agent behaviour using social psychology. Lastly, Bard et al. [23] investigated how RL techniques can be applied to train agents in a purely cooperative game, Hanabi, with multiple players and imperfect information.

III. Methodology

In order to explore the dynamics of different rule-based and learning-based agents in the two-player iterated prisoner's dilemma (IPD) game, we reimplemented a significant proportion of the rule-based agents submitted to Axelrod's tournaments and a selection from the surveyed literature, in addition to training our own tabular, deep Q-network, deep recurrent Q-network agents. Having done so, we then ran a series of round-robin tournaments to explore the relationship between the properties of an agent and its performance, as well as the emergent behaviour of our learning-based agents in a way that is comparable to his eponymous tournaments and related work.

A. Rule-based Agents

In broad terms, we implemented several different categories of rule-based agent: agents submitted to Axelrod's first and second tournaments [3], [8]; constant-action agents; probabilistically cooperating agents; `TitForTat`-style agents; and other interesting agents found in the literature. Table I contains a full list of the rule-based agents we implemented.

When it came to reimplementing rule-based agents submitted to Axelrod's tournaments, we tried to remain faithful to the original agent strategy descriptions set out by Axelrod in his tournament analyses. All of the first tournament agents were recreated, except the agent whose author withheld their

TABLE I
RULE-BASED AGENTS IMPLEMENTED

Category	Agent

Axelrod's first tournament [3]	`TitForTat` (submitted by Anatol Rapoport)
	`Random`
	`Davis`
	`Downing`
	`Feld`
	`Friedman`
	`Graaskamp`
	`Grofman`
	`Joss`
	`Nydegger`
	`Shubik`
	`SteinAndRapoport`
	`TidemanAndChieruzzi`
	`Tullock`

Axelrod's second tournament [8]	`Adams`
	`Black`
	`Börufsen`
	`Cave`
	`Champion`
	`GraaskampAndKatzen`
	`Harrington`
	`Leyvraz`
	`TidemanAndChieruzzi2`
	`Weiner`
	`White`

Constant	`AllC` `AllD`
Stochastic [25]	`ProbabilisticLB1` `ProbabilisticLB2`
`TitForTat`-like strategies	`GenerousTitForTat` [26] `GradualTitForTat` [27] `OmegaTitForTat` [28] `TitForTwoTats` [3] `TwoTitsForTat` [29]
Miscellaneous	`Grudger` [29] `Pavlov` (win–stay–lose–shift) [15]

name, with some minor adjustments to account for our tournament continuing probabilistically. For example, our version of `TidemanAndChieruzzi` does not defect on the last two moves, since they are not known. We then added 11 more well-performing agents from Axelrod's second tournament.

We would like to profusely thank the maintainers of the Python Axelrod library [24], whose own strategy implementations have been incredibly helpful to us in running our own tournaments, particularly where agent descriptions have not been completely clear.

Nowak and Sigmund investigated agents that had separate probabilities of cooperating each move depending on whether the opponent cooperated or defected in the previous move [25]. We thought it would be interesting to briefly explore this further, implementing it as `ProbabilisticLB1` and extending it to cover the opponent's previous two moves as `ProbabilisticLB2`.

B. Learning-based Agents

Our primary focus when it came to learning-based agents was centred on Q-learning: a type of model-free reinforcement learning [9]. For some action A (`COOPERATE` or `DEFECT`), state S (encoding a fixed number of the last few moves played by the agent and its opponent), new state S' (after A is played), reward R, learning rate α and discount factor γ, its action–value function updates in the following way [10]:

$$Q(S,A) \leftarrow Q(S,A) + \alpha\left[R + \gamma \cdot \max_a Q(S',a) - Q(S,A)\right] \tag{1}$$

We examined two Q-learning variants: the more traditional tabular approach, wherein $Q : S \times A \to \mathbb{R}$ is represented by a Q-table; and deep Q-learning, where a neural network – in this context known as a deep Q-network – is used to parameterise Q instead. Our tabular and deep Q-learning implementations made particular use of the NumPy [30] and PyTorch [31] libraries, respectively.

For our deep Q-learning agents, we chiefly experimented with fully connected neural networks with one and two hidden layers of sizes between 4 and 32 neurons. We used the rectified linear unit (ReLU) activation function throughout, as well as the Adam optimiser and Huber loss, having experimented with using mean square error (MSE) as well.

Originally, we were initialising the weights of our linear layers using the default Xavier initialisation strategy used in PyTorch [32], which picks values uniformly at random from the range $-\frac{1}{\sqrt{n}}$ to $\frac{1}{\sqrt{n}}$, where n is the number of neurons in the layer; however, on occasion, our Q-networks would refuse to train properly. Fortunately, this was remedied by instead initialising layer weights with the procedure set out by He et al. [33] using a uniform distribution (`torch.nn.init.kaiming_uniform_` in PyTorch):

$$\mathcal{U}\left(-\sqrt{\frac{6}{n}}, \sqrt{\frac{6}{n}}\right)$$

Out of interest and curiosity, we also tried using a gated recurrent unit (GRU) recurrent neural network (RNN) [34] followed by a linear layer as our Q-network, although the slower training times and similar performance to our DQN agents limited the extent to which we explored its use.

In training, the ϵ-greedy strategy [10] was employed to select the next action at each time step t:

$$A_t = \begin{cases} \operatorname{argmax}_a Q(S_t, a_t), & \text{with probability } 1-\epsilon \\ \text{a random action} & \text{otherwise} \end{cases} \tag{2}$$

In other words, with probability ϵ (the exploration rate), an action is selected uniformly at random from the action space $\mathcal{A} = \{\texttt{COOPERATE}, \texttt{DEFECT}\}$; otherwise, the action with the highest associated Q-value for the state S_t is selected greedily.

The learning-based agents were trained in an environment where they would repeatedly play 200-move games against a certain subset of rule-based agents sequentially with the agent order shuffled every epoch. For example, a tabular Q-learning agent could be trained against `TitForTat`, `AllC` and `AllD` for 2000 epochs, where every epoch the agent would play against those three agents in a random order.

This approach was taken, as opposed to playing a certain number of matches against each individual agent type in order

(e.g. playing 2000 matches against `TitForTat` followed by 2000 matches against `AllC` and so on) as we found that this would cause agents to readapt whenever the agent being trained against changed and therefore ultimately overfit to the last opponent.

In evaluation, we switched to a purely greedy strategy so that at any time-step t, the action taken would be $A_t = \text{argmax}_a Q(S_t, a_t)$. We considered instead using the ϵ-greedy strategy with a small ϵ value, such as 0.05, as done by Mnih et al. [35] to reduce the risk of overfitting in evaluation; however, we found it to be disruptive in the tournament context. For example, it would introduce the risk of a random defection breaking up a chain of cooperation once established from which many agents would be unable to recover to the detriment of the agent's performance in the tournament.

The strategy taken to evaluate these learning-based agents for different available history lengths (called the `lookback` in our code), hidden layer sizes, exploration rates, learning rates and discount factors is elaborated in Section III-D.

A \ B	**Cooperate**	**Defect**
Cooperate	3, 3	0, 5
Defect	5, 0	1, 1

TABLE II
THIS IS THE PAYOFF MATRIX USED TO ASSIGN REWARDS IN THE TOURNAMENTS WE RAN FOR TWO PLAYERS, A AND B. THIS IS IDENTICAL TO THE ONE USED IN AXELROD'S TOURNAMENTS [3], [8].

C. Tournament Design

In order to evaluate the performance of the implemented rule-based and learning-based agents, it was necessary to agree on a specific tournament structure among the many tournament types, e.g. knockout tournaments.

In order for our results to be more comparable with the existing body of research, we elected to conduct a series of probabilistically continuing round-robin tournaments – in which each agent plays itself and every other agent a fixed number of times – so as to mirror the setup in Axelrod's second tournament [8]. After each move in a game, there was a constant probability of the game continuing for another iteration of 0.99654 as in the second tournament.

The iterated prisoner's dilemma games we ran also rewarded players in accordance with the same payoff matrix used in Axelrod's tournaments, shown in table II.

With the number of matches growing in proportion to $\mathcal{O}(n^2)$ for n agents, our cross-platform Python tournament code runs parallelisable agent matches across multiple processes in order to achieve a speed-up over running each match sequentially.

We have made our tournament code available at the following link: https://github.com/UCL-COMP0031-2022-Project-10/tournament/

D. Data Collection

One of our first goals was to run two tournaments as described in Section III-C containing the agents we had implemented from Axelrod's first and second tournaments (listed in table I), respectively, along with `TitForTat` and `Random`, so that we could verify our own agent implementations aligned with Axelrod's original results, conscious that Axelrod's first tournament had a fixed move cap of 200. In doing so, each game between two agents was repeated 5000 times and the results averaged for robustness.

Shifting our focus to training and evaluating learning-based agents, we performed a series of grid searches to more comprehensively cover the large space of hyper-parameters and opponent subsets to be trained against. To do so, for each combination we were seeking to investigate, an agent would be trained in the training environment described in Section III-B and then evaluated by running it against other agents in a tournament with its resulting score and rank logged.

The capacity for a tabular agent to deliberately 'overfit' to a particular opponent or small subset of opponents was briefly examined by conducting grid searches in which an agent would train and then compete against the same sample of opponents, e.g. `TitForTat` or other individual rule-based agents listed in Table I.

Nevertheless, our primary objective when it came to learning-based agents was to train agents that could generalise more widely and therefore perform well in a much larger evaluation tournament.

The agents used as part of the evaluation tournament in our grid searches were the following: `AllC`, `AllD`, `Black`, `Böufsen`, `Champion`, `Davis`, `Downing`, `Feld`, `GenerousTitForTat`, `GraaskampAndKatzen`, `GradualTitForTat`, `Grofman`, `Grudger`, `Harrington`, `Joss`, `Leyvraz`, `Nydegger`, `OmegaTitForTat`, `Pavlov`, `Random`, `Shubik`, `SteinAndRapoport`, `TidemanAndChieruzzi`, `TidemanAndChieruzzi2`, `TitForTat`, `TitForTwoTats`, `Tullock`, `TwoTitsForTat`, `Weiner` and `White`. These agents were selected for being reasonably performant, as well as encompassing a large subset of the rule-based agents we had implemented.

E. Analysis Approach

Due to the stochastic nature of many of the rule-based agents, interpreting the tournament data produced by our grid searches was a challenge; the same agent instance would often fluctuate in the rankings across tournament runs. Nevertheless, we set out to identify broad trends between hyperparameters, such as the exploration rate and discount factor, and the

TABLE III
TOURNAMENT RESULTS INVOLVING AGENTS FROM AXELROD'S FIRST TOURNAMENT (5000 GAME REPETITIONS PER PAIR) ALONGSIDE AXELROD'S ORIGINAL RESULTS

#	Agent	Mean Score	Agent from Axelrod's first tournament	Mean Score
1	SteinAndRapoport	704.11	TitForTat	504
2	TidemanAndChieruzzi	700.14	TidemanAndChieruzzi	500
3	TitForTat	694.16	Nydegger	486
4	Grofman	677.77	Grofman	482
5	Davis	664.39	Shubik	481
6	Shubik	663.84	SteinAndRapoport	478
7	Friedman	661.32	Friedman	473
8	Nydegger	659.89	Davis	472
9	Graaskamp	551.9	Graaskamp	401
10	Tullock	465.96	Downing	391
11	Downing	441.97	Feld	328
12	Feld	441.09	Joss	304
13	Joss	410.11	Tullock	301
14	Random	399.95	Random	276

While the original scores from Axelrod's first tournament have been included for completeness, it is nevertheless important to note that Axelrod's first tournament had fixed game lengths of 200, whereas our tournament run continues probabilistically as elsewhere in this paper, so the mean scores are going to be of different magnitudes across both runs.

TABLE IV
TOURNAMENT RESULTS INVOLVING AGENTS FROM AXELROD'S SECOND TOURNAMENT (5000 GAME REPETITIONS PER PAIR)

#	Agent	Mean Score
1	Cave	865.91
2	Weiner	864.73
3	GraaskampAndKatzen	863.27
4	Borufsen	858.93
5	White	850.63
6	Champion	850.33
7	Leyvraz	848.91
8	TitForTat	848.73
9	Black	842.41
10	Adams	834.39
11	TidemanAndChieruzzi2	828.05
12	Harrington	806.5
13	Random	532.57

performance of an agent as measured by their tournament rank and score from our many tournament runs.

IV. RESULTS & DISCUSSION

A. Axelrod-style Tournaments

Tables III and IV show results from running two tournaments involving agents from Axelrod's first and second tournaments, respectively. The original results from Axelrod's first tournament have been included alongside our own in Table III to facilitate comparisons of agents' rankings.

It should be noted, however, that while games in Axelrod's first tournament always ran for 200 moves, games in our tournament continued probabilistically as in his second tournament; moreover, we were unable to include the agent associated with the submitter who withheld their name. As a result, it would not be appropriate to compare the magnitudes of the scores directly.

As can be seen in Table III, most of the first tournament agents nevertheless ranked similarly across our run and Axelrod's. Indeed, TidemanAndChieruzzi, Grofman, Friedman, Graaskamp and Random maintained the same rank across both tournaments.

On the other hand, TitForTat lost its crown, dropping down to third place after SteinAndRapoport and TidemanAndChieruzz, although it still ranked amongst some of the best-performing agents coming in only 9.95 below the best agent (SteinAndRapoport) and 294.21 above the very worst agent (Random).

The most dramatic change was SteinAndRapoport soaring from 6th place in the original tournament to 1st place in our one. This matches up more closely with a 2015 reanalysis of Axelrod's first tournament by Rapoport et al. [36] in which SteinAndRapoport performs similarly well.

We conjecture that after accounting for the slightly different tournament setups, there are two further main causes that contribute towards the rankings being different:

1) Benefiting from advances in computing power since the original tournaments were run, we were able to repeat each match 5000 times – as compared to the 5 repetitions used in Axelrod's first tournament – thereby dampening some of the effects arising from the stochastic nature of some agents; and
2) While the excluded agent with the anonymous author came second-to-last in Axelrod's first tournament, it could still have impacted the overall rankings by helping certain agents while hindering others in the style of a so-called kingmaker agent.

Turning to our tournament run involving 13 agents from Axelrod's second tournament shown in Table IV, we observed very different results. TitForTat fell to 8th place, although its score was merely 17.18 below the top score, which is not a large gap.

Interestingly, there was only a relatively small difference of 59.41 between the scores of the 1st-place and 12th-place agents; however, this was not unexpected considering that we chose to implement 12 of some of the best performing agents out of the 62 entries in Axelrod's second tournament.

B. Learning-based Agents Evaluated Against a Single Agent

Before evaluating our tabular Q-learning agents in the full evaluation tournament, we wanted to briefly test their capacity to overfit to a particular opponent before continuing by evaluating them in a tournament alongside the opponent they were trained against. All agents were trained for 2500 epochs with $\epsilon = 0.2$, $\alpha = 0.01$ and $\gamma = 0.95$. Results from the top 10 runs all ranked first and are shown in Table V.

These exploratory results were interesting, as they showed that our tabular Q-learning agents could learn both more cooperative strategies, such as against TidemanAndChieruzzi2, as well as more exploitative strategies, such as against Adams and Nydegger.

C. Learning-based Agents Evaluated Against Multiple Agents

From Figures 1 and 2, we notice that our tabular agents consistently outperform our DQN agents, with tabular agents

TABLE V
TOP 10 GRID SEARCH RUNS ATTEMPTING TO OVERFIT AGENTS TO A PARTICULAR OPPONENT

#	Training and Evaluation Opponent	LB	COOP%	Mean Reward (Training)	Mean Score
1	`Adams`	2	16%	4.58	1225.93
2	`Nydegger`	2	12%	4.71	1218.83
3	`TidemanAndChieruzzi2`	4	55%	2.47	1019.565
4	`Adams`	4	24%	4.43	958.59
5	`TidemanAndChieruzzi`	2	24%	1.23	950.325
6	`Friedman`	2	14%	0.99	946.635
7	`GraaskampAndKatzen`	4	22%	1.17	932.67
8	`Graaskamp`	4	29%	1.52	928.06
9	`Davis`	4	16%	1.05	927.18
10	`GraaskampAndKatzen`	2	20%	1.14	871.155

LB stands for the lookback, i.e. the number of most recent moves upon which agents would base their decisions. COOP% stands for the percentage of cooperative moves throughout the training process.

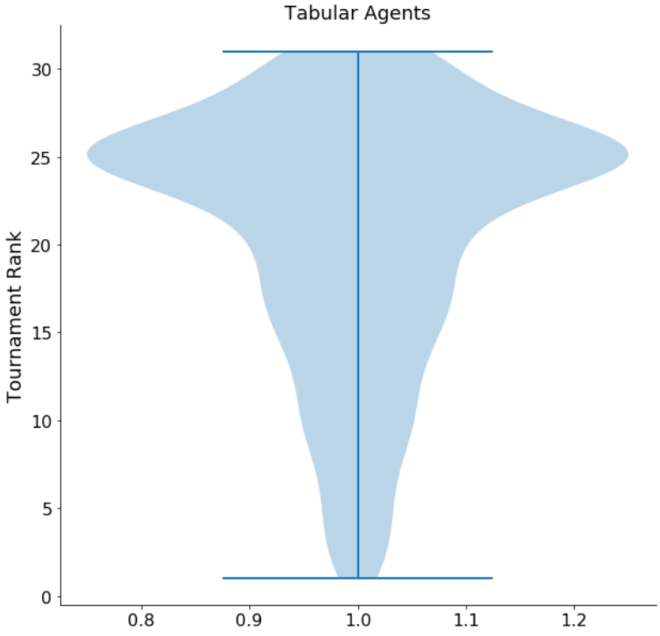

Fig. 1. Distribution of Tournament Ranks for Tabular Agents.

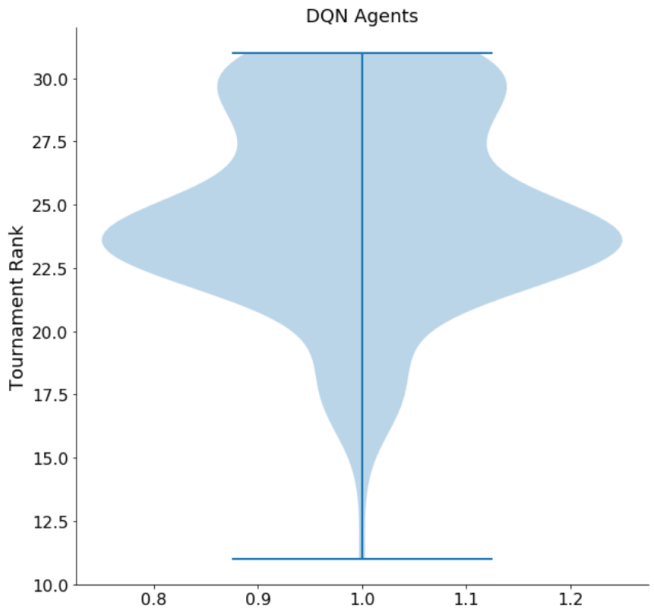

Fig. 2. Distribution of Tournament Ranks for DQN Agents.

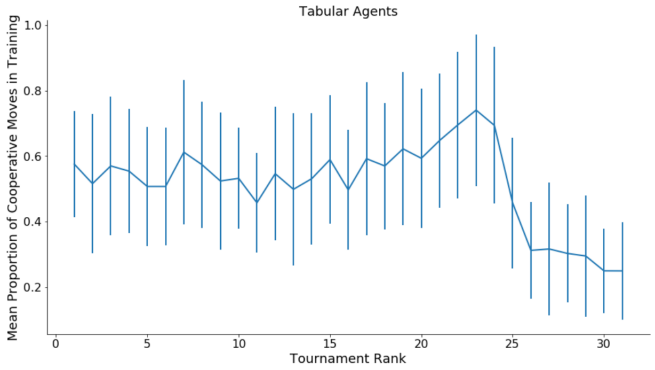

Fig. 3. The Relationship between the Rank of a Tabular Agent and its Mean Proportion of Cooperative Moves in Training

averaging a rank of 20.86 relative to 24.88 for DQN agents. A deeper analysis of the best and worst-performing agents revealed that the best-performing tabular agents were able to learn more sophisticated strategies that were often variants of the better rule-based agents; on the other hand, the best DQN agents would adopt the `AllC` strategy while the worst DQN agents learnt to utilise `AllD`. This could be indicative of a failure of our DQN agents to properly train with our setup.

Although the best tabular agents displayed a diverse range of strategies, we shall focus our analysis on two tabular agents we found particularly interesting that both ranked first in their respective evaluation tournaments: we have affectionately named them `GradualFriedman` and `Forgiver`. Both were trained with the discount factor $\gamma = 0.99$ and the learning rate $\alpha = 0.05$. `GradualFriedman` has a lookback of 2, whereas `Forgiver` has a much longer lookback of 8, basing its moves on a longer history of actions.

Both of these agents are nice, i.e. they are never the first to defect. Against agents, such as `TitForTat`, they are able to continue mutual cooperation indefinitely, never being the first to defect. Additionally, both of these strategies are provokable and will eventually punish defections by switching to an `AllD`-like strategy.

`Forgiver`, as the name suggests, is extremely forgiving; it will tolerate multiple defections before defecting, and if its adversary starts cooperating again, `Forgiver` will try to restart a streak of mutual cooperation. Forgiveness is a useful property while playing against rule-based agents of a stochastic nature, as they may sometimes defect randomly. In order to not jeopardise their streak of mutual cooperation, `Forgiver` allows them to get away with the defection.

This trait comes with its weaknesses as well; you can get away with defecting every four turns without `Forgiver` retaliating, allowing for a degree of exploitation. In contrast, `GradualFriedman` is completely intolerant of defections.

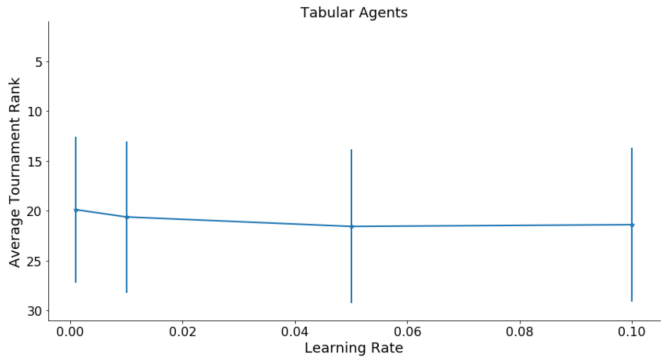

Fig. 4. The Impact of Learning Rate on Tabular Agent Performance.

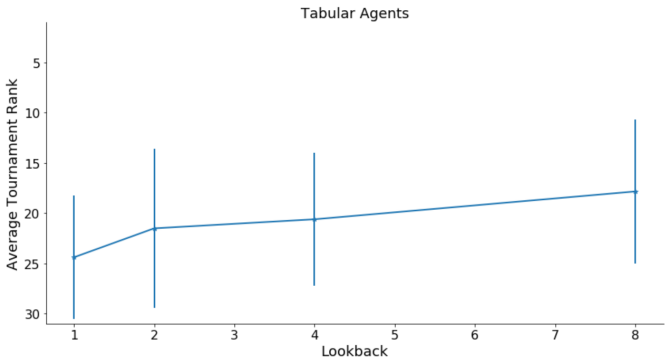

Fig. 5. The Impact of Lookback on Tabular Agent Performance.

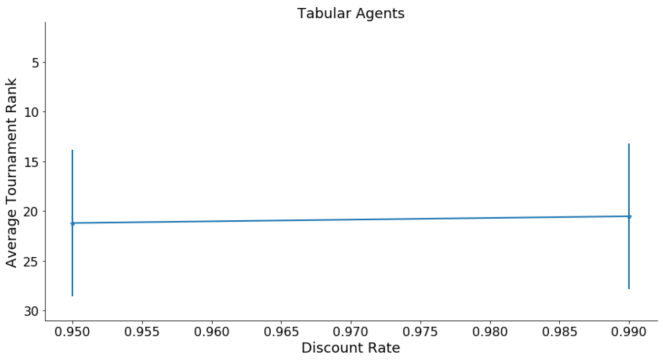

Fig. 6. The Impact of Discount Rate on Tabular Agent Performance.

After saturating its history with three defections, `Gradual-Friedman` becomes a lot more exploitative, choosing not to resume cooperation. This strategy is reminiscent of policy employed by the agent `Friedman`, which immediately switches to `AllD` upon encountering a defection. `Friedman`'s main flaw is that it gives up on cooperating too early on, leading to chains of mutual defections, which results in a worse performance in the tournament. `GradualFriedman`, on the other hand, is more tolerant of random defections, resulting in a higher tournament ranking. The optimal strategy against `GradualFriedman` is to cooperate and to defect once it starts defecting.

From Figure 3, we observe that highest-ranking agents leverage cooperation more frequently in contrast to the worst-performing agents that defect more frequently during the training process. Indeed, analysis of the strategies adopted by DQN agents makes it abundantly clear how polarised the strategies learnt by agents are: the top-ranking DQN agents cooperate 97% of the time while the worst performing ones cooperate 17% of the time in training.

Unfortunately, it seems that it was difficult for our DQN agents to learn a more refined strategy with our model architectures and training approach. An analysis of how we were training our DQN agents highlighted that DQN agents with a higher learning rate or increased number of epochs seemed to fare better in the tournament. Conversely, our tabular agents performed better with a low learning rate with the optimal value being 0.001. Interestingly, cooperation also decreases with the learning rate from 66% to 38% in our data.

We observe a linear improvement in performance with lookback in Figure 5, showcasing that tabular agents can sometimes make use of the information encoded in a longer history to learn more sophisticated strategies. Furthermore, we posit that a longer lookback could allow a tabular agent to learn to distinguish between agents, such as `TitForTat` and `AllD`, allowing it to switch strategies based on its opponent. This would not be possible with a lookback of 1, for example. A longer lookback also correlates with cooperativeness; an agent with a lookback of 1 cooperates 40% of the time on average, while an agent with a lookback of 8 cooperates 62% of the time, helping to improve its performance in a tournament of predominantly cooperative agents.

Our DQN agents do not display the same relationship. The peak DQN agent performance occurs with a lookback of 4 and decreases with each increment of lookback. Each addition to lookback adds an extra dimension to the state space, increasing the potential for our agents to get stuck in suboptimal local minima.

The preferred discount rate for both tabular and DQN agents is 0.99, which can be seen from Figure 6. One interpretation of the discount rate is the degree to which an agent values potential future rewards; therefore, an agent with a discount rate of 0.99 values future interactions more highly than one with $\gamma = 0.95$. An agent that considers future interactions more will realise that defections tend to lead to more defections, whereas cooperation tend to be rewarded with more cooperation; thus, the future value of cooperation is higher than defection. Agents with a discount rate of 0.99 are, on average, 18% more cooperative than agents with a discount rate of 0.95. Despite discount rate improving performance of agents, it is does not have as much of an impact as tuning the other hyperparameters does.

From Figure 7, we observe tabular agents like to have a low exploration rate, displaying an almost linear decrease in performance with exploration rate; thus, we can conclude that tabular agents do not necessarily need to explore too much beyond the initially cooperative behaviour caused by having a zeroed-out Q-table. This partially stems from an implementation detail: in our code, our tabular agents cooperate when the Q-values for cooperating and defecting are equal, so agents cooperate when an unseen state is encountered. Furthermore, a higher exploration rate leads to an increased

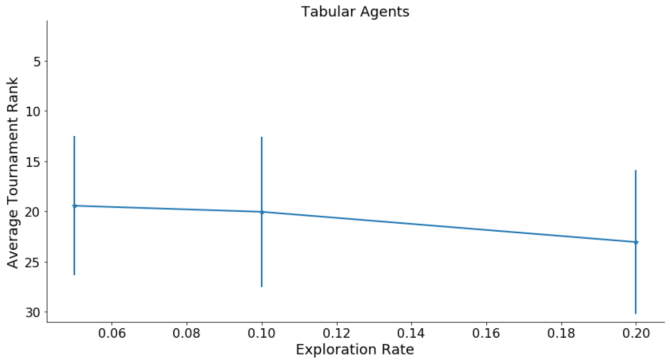

Fig. 7. The Imapct of Exploration Rate on Tabular Agent Performance.

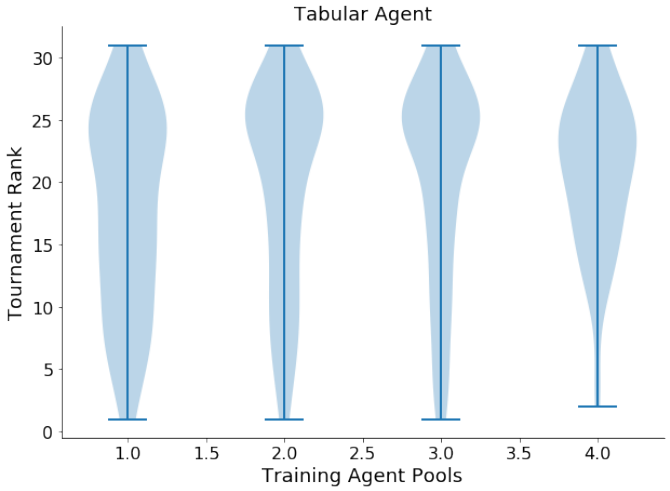

Fig. 8. The Impact of Different Training Opponent Pools on Agent Performance. Pool 1 is `TFT`, `TFTT`, `TTFT`, `GenerousTFT`, `OmegaTitForTat`, `Davis`, `Grofman`, `Leyvraz`; Pool 2 is `TFT`, `Joss`, `Downing`; Pool 3 is `TFT`, `Champion`, `Böufsen`, `SecondByCave`, `SecondByWmAdams`, `SecondByGraaskampKatzen`; Pool 4 is `TFT`, `GradualTFT`, `GenerousTFT`, `TTFT`, `TFTT`, `OmegaTFT`

number of random moves, which leads to more defections, potentially explaining the substantial decrease in cooperative moves during the training process.

Conversely, an exploration rate of 0.1 appears to be the goldilocks value for DQN agents, increasing or decreasing it reduces performance. This is attributed to the fact that DQN agents like to explore their environment to get out of sub-optimal minima, but too large an exploration rate hinders the agent from refining their strategy.

We found that training our agents against particular training opponents significantly boosted their performance, as can be observed from Figure 3. One such pool was the combination of `Tit-For-Tat`, `Tit-For-Two-Tats`, `Two-Tits-For-Tat`, `GenerousTFT`, `OmegaTitForTat`, `Davis`, `Grofman`, and `Leyvraz`. Playing against all of the `TFT`-esque agents in our tournament taught our agents to be nice, as the optimal strategy against all of the `TFT` agents is mutual cooperation. Both `Leyvraz` and `Grofman` have stochastic properties, which are beneficial in training as it allows our agents to reach states that they would never otherwise reach if the agents had been deterministic.

TABLE VI
TOP 15 AGENTS FROM THE `PROBABILISTICLB1` EVALUATION TOURNAMENT

#	Agent	Mean Score
1	`Grudger`	855.2069231
2	`GraaskampAndKatzen`	836.6753846
3	$p_C = 0.00195031, p_D = 0.027560873$	831.3684615
4	OmegaTitForTat	828.9353846
5	GradualTitForTat	818.22
6	AllD	811.2969231
7	Shubik	810.2030769
8	Davis	807.2069231
9	$p_C = 0.026136146, p_D = 0.16342874$	791.3538462
10	`Böufsen`	762.9353846
11	$p_C = 0.013708539, p_D = 0.349198779$	758.0376923
12	$p_C = 0.066677213, p_D = 0.253919453$	755.9353846
13	$p_C = 0.004784118, p_D = 0.036253353$	755.2330769
14	$p_C = 0.158565961, p_D = 0.092609743$	754.4038462
15	`Downing`	750.2638462

Finally, `Davis` is similar to `GradualFriedman` – it will cooperate for the first ten moves and continue to cooperate until the first defection, after which it will play as `AllD`. If we did not have `Davis` in our pool, our agents would not know how to appropriately handle repeated defections as the Q-values for that state would not be well trained. We conclude that it is useful for agent pools to include `TFT` agents to teach our agents to be nice, stochastic agents so that we can train on a broad range of different states, and an unforgiving agent to teach our agents to defect in response to defections.

In summary, the best performing Q-learning agent we found for the IPD is a tabular Q-learner with a discount rate of 0.99, epsilon of 0.01, lookback of 8, learning rate of 0.001 trained against `TitForTat`, `TitForTwoTats`, `TwoTitsForTat`, `GenerousTitForTat`, `OmegaTitForTat`, `Davis`, `Grofman`, and `Leyvraz`.

D. Probabilistically Cooperating Agents

In order to investigate the probabilistically cooperating agents we had implemented, we generated 100 and 500 instances of `ProbabilisticLB1` and `ProbabilisticLB2`, respectively, each with their cooperation probabilities chosen uniformly at random from the interval $[0, 1)$. We then ran two tournaments: one containing the 100 `ProbabilisticLB1` instances alongside the other rule-based evaluation agents; and another but with the 500 `ProbabilisticLB2` instances. The top 15 agents from these tournaments are shown in Tables VI and VII.

In contrast with the other tournaments we ran, these tournaments were overwhelming defective environments, leading to more exploitative agents, such as `AllD`, ranking more highly. This shows the importance of the selection of agents used for evaluation in influencing agents' tournament rankings.

V. LIMITATIONS

There are a few main limitations of our approach. Firstly, while we endeavoured to include a wide selection of rule-based agents, we did not reimplement all the agents from Axelrod's

TABLE VII
TOP 15 AGENTS FROM THE PROBABILISTICLB2 EVALUATION TOURNAMENT

#	Agent	Mean Score
1	GraaskampAndKatzen	829.6992453
2	OmegaTitForTat	826.8977358
3	AllD	825.7692453
4	Downing	824.9439623
5	Grudger	821.0443396
6	Tullock	814.164717
7	Shubik	811.2573585
8	GradualTitForTat	810.4581132
9	Davis	797.9886792
10	$p_{CC} = 0.03$, $p_{CD} = 0.40$, $p_{DC} = 0.04$, $p_{DD} = 0.56$	779.3311321
11	$p_{CC} = 0.08$, $p_{CD} = 0.10$, $p_{DC} = 0.44$, $p_{DD} = 0.05$	775.3160377
12	$p_{CC} = 0.02$, $p_{CD} = 0.40$, $p_{DC} = 0.03$, $p_{DD} = 0.59$	773.1560377
13	Feld	764.3362264
14	$p_{CC} = 0.07$, $p_{CD} = 0.12$, $p_{DC} = 0.41$, $p_{DD} = 0.32$	763.795283
15	$p_{CC} = 0.09$, $p_{CD} = 0.60$, $p_{DC} = 0.01$, $p_{DD} = 0.15$	762.5388679

first and second tournaments; moreover, our sample of agents from Axelrod's second tournament was drawn from amongst the top performers, which tended to have similar properties, such as niceness, potentially biasing the composition of the evaluation tournaments we ran against exploitative strategies.

Given the performance of an agent depends on its opponents in a tournament, a greater presence of kingmaker agents – which themselves perform poorly but serve to discriminate between top-performing agents in the rankings [3] – could affect the rankings of our Q-learning agents positively or negatively. It would therefore be interesting to try training and evaluating Q-learning agents against a more diverse sample of rule-based strategies. Unfortunately, there does not seem to be a clear definition as to what constitutes a kingmaker agent, so more research is potentially needed into fairer tournament designs.

Furthermore, due to the limited compute resources available to us, we were unable to explore as many training opponent subsets and hyperparameter combinations as we would have liked, e.g. exploration rate ϵ (including with a schedule), discount factor γ and learning rate α values. This is acutely apparent in our deep Q-learning results with respect to hidden layer sizes. As a result, we had to judiciously select a few opponent samples in which we were interested and distribute the grid search computations across the team. This could have knock-on effects on the strength of our conclusions. In the future, it would be interesting to analyse the impact of a much larger variety of training opponent selections.

Finally, while tabular Q-learning agents sometimes demonstrated a propensity to readily overfit to particular opponents – in one run achieving a phenomenal score of 1226 in a match against Adams, the agent it was trained against – their performance would drop in the much larger tournaments indicating a failure to generalise completely.

VI. CONCLUSION & FUTURE WORK

In conclusion, in this report, we review related work into how studying societies, and in particular tournaments, of artificial agents has contributed towards understanding the evolution of cooperation and then discuss our work exploring round-robin tournaments and chiefly applying Q-learning to discover new insights about the nature of cooperation in the two-player iterated prisoner's dilemma game.

We have recreated subsets of Axelrod's tournaments, while implementing other interesting agents from the literature. We were inspired by Nowak and Sigmund's probabilistically cooperating agents [25] to extend them to have a lookback of 2. Finally, we have seen how tabular Q-learning agents in particular are capable not only of overfitting to particular opponents and performing well in that context, but also of learning sophisticated cooperative strategies and ranking highly in a much larger tournament against cooperative rule-based agents.

While there have been limitations to our approach, we are optimistic about what our findings suggest in relation to the potential for applying Q-learning and reinforcement learning more broadly to cooperative problems and social dilemma games. We therefore end by suggesting a few potential avenues for future research.

A. Future Work

1) Investigating the Addition of Noise: Noise in the form of random errors in implementing a choice is a common problem in real-world interactions [37]. It would be interesting to investigate the impact of introducing varying levels of noise in the tournaments on agents' performance to explore which strategies would be robust, generous and forgiving in the presence of noise. It would also be fascinating to explore the impact of noise in the training environment on the strategies to which learning-based agents converge.

2) Multi-player Iterated Prisoner's Dilemma: While our research has focused on the two-player IPD, in many real-world scenarios, cooperation usually occurs in groups of individuals. It is therefore worth investigating how cooperative relationships can be forged in a suitable multi-agent structure or networked population, where the number of individuals and the amount of resource can come into play, although this would require a re-evaluation of the tournament approach.

3) Exploring Other Evaluation Strategies: While our research has centred on round-robin tournaments, it could be interesting to analyse the impact of other tournament structures, e.g. knockout tournaments. Furthermore, Axelrod investigated a more 'ecological' approach where the distribution of agents in a population could change over time [8]. This is another interesting lens to examine the performance of agents through, particularly if agents can select their partners in each round, as investigated by Anastassacos et al. [21].

4) Investigating Other Forms of Reinforcement Learning: While we focused on the more traditional implementation of Q-learning, it would be interesting to investigate the application of double Q-learning [10] and more complex actor-critic methods [38] to the two-player IPD.

REFERENCES

[1] R. Axelrod and W. D. Hamilton, "The evolution of cooperation," *science*, vol. 211, no. 4489, pp. 1390–1396, 1981.

[2] M. A. Nowak, "Five rules for the evolution of cooperation," *science*, vol. 314, no. 5805, pp. 1560–1563, 2006.

[3] R. Axelrod, "Effective choice in the prisoner's dilemma," *Journal of conflict resolution*, vol. 24, no. 1, pp. 3–25, 1980.

[4] ——, *The Evolution of Cooperation*, ser. Penguin Press Science Series. Penguin, 1990. [Online]. Available: https://books.google.co.uk/books?id=tZdfQgAACAAJ

[5] A. Dafoe, E. Hughes, Y. Bachrach, T. Collins, K. R. McKee, J. Z. Leibo, K. Larson, and T. Graepel, "Open problems in cooperative AI," *CoRR*, vol. abs/2012.08630, 2020. [Online]. Available: https://arxiv.org/abs/2012.08630

[6] R. L. Trivers, "The evolution of reciprocal altruism," *The Quarterly Review of Biology*, vol. 46, no. 1, pp. 35–57, 1971. [Online]. Available: http://www.jstor.org/stable/2822435

[7] L. Panait and S. Luke, "Cooperative multi-agent learning: The state of the art," *Autonomous agents and multi-agent systems*, vol. 11, no. 3, pp. 387–434, 2005.

[8] R. Axelrod, "More effective choice in the prisoner's dilemma," *Journal of conflict resolution*, vol. 24, no. 3, pp. 379–403, 1980.

[9] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3, pp. 279–292, 1992.

[10] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[11] J. von Neumann, *1. On the Theory of Games of Strategy*. Princeton University Press, 2016, pp. 13–42. [Online]. Available: https://doi.org/10.1515/9781400882168-003

[12] W. Hamilton, "The Genetical Evolution of Social Behaviour. I," *Journal of Theoretical Biology*, vol. 7, no. 1, pp. 1–16, 1964.

[13] ——, "The Genetical Evolution of Social Behaviour. II," *Journal of Theoretical Biology*, vol. 7, no. 1, pp. 17–52, 1964.

[14] R. L. Trivers, "The Evolution of Reciprocal Altruism," *The Quarterly Review of Biology*, vol. 46, no. 1, p. 35–57, 1971.

[15] M. Nowak and K. Sigmund, "A strategy of win-stay, lose-shift that outperforms tit-for-tat in the prisoner's dilemma game," *Nature*, vol. 364, no. 6432, pp. 56–58, 1993.

[16] C. J. Watkins and P. Dayan, *Machine Learning*, vol. 8, no. 3/4, p. 279–292, 1992.

[17] B. Banerjee and J. Peng, "Adaptive policy gradient in multiagent learning," *Proceedings of the second international joint conference on Autonomous agents and multiagent systems - AAMAS '03*, 2003.

[18] J. V. Barbosa, A. H. R. Costa, F. S. Melo, J. S. Sichman, and F. C. Santos, "Emergence of cooperation in n-person dilemmas through actor-critic reinforcement learning," 2020.

[19] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," 2016. [Online]. Available: https://arxiv.org/abs/1602.01783

[20] M. Harper, V. Knight, M. Jones, G. Koutsovoulos, N. E. Glynatsi, and O. Campbell, "Reinforcement learning produces dominant strategies for the iterated prisoner's dilemma," *PloS one*, vol. 12, no. 12, p. e0188046, 2017.

[21] N. Anastassacos, S. Hailes, and M. Musolesi, "Partner selection for the emergence of cooperation in multi-agent systems using reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 7047–7054.

[22] J. Z. Leibo, V. F. Zambaldi, M. Lanctot, J. Marecki, and T. Graepel, "Multi-agent reinforcement learning in sequential social dilemmas," *CoRR*, vol. abs/1702.03037, 2017. [Online]. Available: http://arxiv.org/abs/1702.03037

[23] N. Bard, J. N. Foerster, S. Chandar, N. Burch, M. Lanctot, H. F. Song, E. Parisotto, V. Dumoulin, S. Moitra, E. Hughes, and et al., "The hanabi challenge: A new frontier for ai research," *Artificial Intelligence*, vol. 280, p. 103216, 2020.

[24] V. Knight, O. Campbell, Marc, T. Gaffney, E. Shaw, V. R. Janga, N. Glynatsi, J. Campbell, K. M. Langner, S. Singh, J. Rymer, T. Campbell, J. Young, MHakem, G. Palmer, K. Glass, D. Mancia, edouardArgenson, M. Jones, kjurgielajtis, Y. Murase, S. Parvatikar, M. Beck, C. Davidson-Pilon, M. Zoulias, A. Pohl, P. Slavin, T. Standen, A. Kratz, and A. Ahmed, "Axelrod-python/axelrod: v4.12.0," Oct. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5616793

[25] M. A. Nowak and K. Sigmund, "Tit for tat in heterogeneous populations," *Nature*, vol. 355, no. 6357, pp. 250–253, 1992.

[26] C. Wedekind and M. Milinski, "Human cooperation in the simultaneous and the alternating prisoner's dilemma: Pavlov versus generous tit-for-tat." *Proceedings of the National Academy of Sciences*, vol. 93, no. 7, pp. 2686–2689, 1996. [Online]. Available: https://www.pnas.org/doi/abs/10.1073/pnas.93.7.2686

[27] B. Beaufils, J.-P. Delahaye, P. Mathieu *et al.*, "Our meeting with gradual, a good strategy for the iterated prisoner's dilemma," in *Artificial Life V: Proceedings of the Fifth International Workshop on the Synthesis and Simulation of Living Systems*. Citeseer, 1996, pp. 202–209.

[28] W. Slany, W. Kienreich *et al.*, "On some winning strategies for the iterated prisoner's dilemma, or, mr. nice guy and the cosa nostra," *The iterated prisoners' dilemma*, vol. 20, p. 171, 2007.

[29] J. Li, P. Hingston, and G. Kendall, "Engineering design of strategies for winning iterated prisoner's dilemma competitions," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, no. 4, pp. 348–360, 2011.

[30] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2

[31] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[32] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[34] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[35] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[36] A. Rapoport, D. A. Seale, and A. M. Colman, "Is tit-for-tat the answer? on the conclusions drawn from axelrod's tournaments," *PloS one*, vol. 10, no. 7, p. e0134128, 2015.

[37] J. Wu and R. Axelrod, "How to cope with noise in the iterated prisoner's dilemma," *Journal of Conflict Resolution*, vol. 39, no. 1, p. 183–189, 1995.

[38] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in neural information processing systems*, vol. 30, 2017.