Exploring Cooperation in Multi-Agent Systems: A Q-Learning Approach to the Iterated Prisoner's Dilemma

Student Name: Kaiji Zhou
Supervisor Name: Jennifer Badham

Abstract — tIn the realm of multi-agent systems, decision-making processes have gained prominence, simulating complex real-world scenarios. This research explores the dynamics of decision-making within the Iterated Prisoner's Dilemma (IPD), employing Q-learning as a model-free reinforcement learning algorithm. Through a systematic approach, the study investigates the behaviour of agents under varying weights of individual and team rewards, uncovering patterns of cooperation, trust, and reciprocity. The research reveals that agents are likely to opt for mutually cooperative strategies, even when team benefits are not considered. Concurrently, we have calculated and verified a critical weight point at which a pivotal shift in agent behavior is likely to occur. The results provide insights into the alignment and evolution beyond the Nash Equilibrium, contributing to the broader understanding of cooperation in game theory.

Index Terms — Artificial Intelligence, Game Theory, Multi-Agent Learning, Prisoner's Dilemma, Q-Learning, Reinforcement Learning

1 Introduction

In the contemporary world of artificial intelligence and computational modeling, the dynamics of decision-making processes within multi-agent environments have become a focal point of research. Multi-agent systems are attractive for their ability to model complex interactions among diverse entities such as robots, software agents, and humans across various sectors like transportation, finance, and healthcare.

Historically, game theory has provided a systematic framework to understand decision-making in scenarios characterized by conflict, cooperation, and strategic interactions. The Prisoner's Dilemma, conceived in the 1950s by Merrill Flood and Melvin Dresher[10], stands as a paragon in this domain, illustrating the complexities inherent in decision-making processes.

The crux of the Prisoner's Dilemma lies in its inherent conflict between individual and collective rationality. While each player's optimal strategy (from a purely selfish standpoint) is to defect, this mutual defection leads to a suboptimal outcome for both players. On the other hand, mutual cooperation, though collectively rational, is unstable because each player has a temptation to defect to maximize their individual gain. This tension between personal and collective interest, coupled with the dependency of one's payoff on the other player's choice, encapsulates the essence of the "dilemma".[11]

The Iterated Prisoner's Dilemma (IPD) extends this concept, introducing repetitive interactions that mimic long-term relationships, be they between businesses, nations, or individuals. In the IPD, strategies evolve over multiple interactions, offering a lens to study the emergence of trust, reciprocity, and long-term cooperation.

Yet, the allure of the IPD and multi-agent systems remains not just in their history but in the challenges they continue to present. Deciphering how varying reward structures influence agent behaviors, particularly when rewards span both individual and collective realms, is a central challenge. Another intricate aspect is navigating the balance between exploration and exploitation, ensuring agents can adapt to dynamic environments while building upon acquired knowledge.

Building upon the foundational concepts of the IPD, a plethora of strategies have been devised to navigate its intricate landscape. The classical "Tit-for-Tat" (TFT) strategy, introduced by Robert Axelrod in his seminal tournaments in the early 1980s, epitomizes a cooperative yet retaliatory approach, where a player mirrors the opponent's last move [12]. Similarly, strategies like "Win-Stay, Lose-Shift" (WSLS) reflect adaptive behavior, responding to the outcomes of prior interactions [13]. The "Grim Trigger" strategy is another noteworthy approach that emphasizes the long-term consequences of defection [14]. More recent advancements have given rise to adaptive strategies leveraging machine learning and computational intelligence, such as Q-learning in the IPD [15] and Genetic Algorithms [16]. These modern approaches facilitate a deeper exploration of dynamics, such as trust, retaliation, forgiveness, and cooperation, enabling agents not only to learn from past interactions but also to anticipate and adapt to evolving scenarios [17]. The study of these strategies offers a multifaceted perspective on the IPD, shedding light on both theoretical constructs and practical applications in various disciplines. This rich tapestry of strategies forms the bedrock of our investigation into the behavior of agents under varying weights of individual and collective rewards, as outlined in the subsequent sections of this paper.

Q-learning, a pivotal algorithm in Reinforcement Learning, is particularly suited for our research as it excels in guiding agents to discern the potential rewards of their actions in dynamic and complex environments like the IPD. Its iterative refinement process allows agents to adapt and optimize decision-making strategies effectively. This research endeavors to address these challenges by marrying the principles of the IPD with Q-learning, a renowned reinforcement learning algorithm.

This research has broad implications. It promises to impact fields ranging from economics and social sciences to global diplomacy by offering new insights into cooperative behaviors. Importantly, it also addresses ethical dimensions of AI. The study's focus on how agents balance individual and collective rewards can inform ethical frameworks, particularly in critical applications like autonomous driving and healthcare.

Guided by the complex nature of the IPD and armed with the robust capabilities of Q-learning, we set forth with two pivotal objectives:

1. To discern agent behaviors influenced by a spectrum of individual and collective rewards.

2. To pinpoint the weightage of team rewards that acts as a turning point in agent decision-making.

To achieve the stated objectives, this research employs the PettingZoo platform, a cutting-edge environment tailored for the development and analysis of multi-agent systems. Utilizing an agent-based modeling approach allows us to scrutinize systems at the micro-level, focusing on individual agents and their interactions. In the IPD environment constructed within PettingZoo, agents are trained using a Q-learning algorithm. By varying the weights assigned to individual and team rewards, we are able to quantitatively compare the emergent behaviors and derive conclusive insights.

The remainder of this thesis is organized to provide a comprehensive understanding of this complex landscape. The upcoming "Methodology" section will further elaborate on the research design and the algorithms employed. This will be followed by the "Results" section, where the empirical data and observations are presented, particularly focusing on agent behavior under different reward weightings. Subsequently, the "Discussion" section will interpret these results in the context of existing literature and theoretical frameworks.

Additionally, we have computed a theoretical reward weight value, serving as a critical inflection point for agent decision-making. This calculated value is then empirically validated, providing robust evidence for the pivot in agent strategies. Through this comprehensive methodology, which will be detailed in the aforementioned sections, we aim to unravel the complexities of agent behavior in cooperative settings, thereby bridging the divide between individual choices and collective outcomes. Finally, the "Conclusion" section will synthesize the key findings and their implications, offering insights into potential avenues for future research.

2 Related Work

The iterated prisoner's dilemma (IPD) has been a cornerstone in the study of cooperative behaviors and strategic decision-making. With the infusion of computational techniques, particularly reinforcement learning, researchers have augmented our understanding of the intricate dynamics and strategies within this game.

In the realm of multi-agent learning and the Iterated Prisoner's Dilemma (IPD), the work by Kies (2020) stands as a significant contribution to the field [1]. This research delves into the complex challenge of gauging the exploitability of strategies within the IPD, considering factors such as imperfect public monitoring and observation errors. Through an improved Q-Learning algorithm, Kies presents a novel approach to constructing a best response to any given strategy, allowing for an accessible numerical measure of exploitability. The study's rigorous theoretical foundation, combined with insights into robust strategies and optimal exploration, makes it a valuable reference for understanding the dynamics of repeated games, especially in the presence of noise.

In the exploration of cooperative strategies within Multi-Agent Reinforcement Learning (MARL), the research by Gupta (2020) introduces a novel obedience-based learning mechanism[2]. This intuitive approach rewards agents for obeying commands, facilitating enhanced coordination and communication in sequential social dilemmas. Demonstrated to be flexible in handling population heterogeneity and adaptable to varying vocabulary sizes, obedience-based learning stands out for its minimal complexity and overhead. Gupta's work offers a promising direction in the field of cooperative MARL, with significant implications for scenarios requiring robust and adaptive agent interactions. The insights from this research contribute to the understanding of agent behavior in complex environments and provide a unique perspective on fostering cooperation through intrinsic rewards.

In the study of cooperation and competition within the Iterated Prisoner's Dilemma (IPD), the work by Stewart and Plotkin (2013) offers a groundbreaking perspective on the role of "zero-determinant" (ZD) strategies [7]. By exploring the evolutionary prospects of these strategies, the authors introduce a subset of generous strategies that not only forgive defecting opponents but also dominate in evolving populations. Their findings demonstrate that generous strategies can be as successful as or even more successful than traditional IPD strategies, providing a novel angle on cooperative behavior that aligns with Darwinian principles. Through meticulous analysis and modeling, the paper reveals a disproportionate favoring of generous strategies in evolving populations, underscoring their significance in understanding the dynamics of cooperation. Stewart and Plotkin's insights into ZD strategies and the success of generosity in the IPD contribute to a deeper comprehension of the evolutionary mechanisms governing agent interactions, enriching the field of multi-agent learning and game theory .

In the examination of strategies within the Iterated Prisoner's Dilemma (IPD), the work by Hilbe, Traulsen, and Sigmund (2015) presents a comprehensive analysis of memory-one strategies, including partner strategies, competitive strategies, and zero-determinant strategies [8]. By exploring these distinct approaches, the authors shed light on the dynamics of cooperation, reciprocity, and extortion within repeated games. Partner strategies emphasize fairness and penalization for unfair behavior, competitive strategies ensure a player never obtains less than a co-player, and ZD strategies unilaterally enforce a linear relationship between players' payoffs. This research contributes to a nuanced understanding of social interactions, enriching the field of game theory and providing valuable insights for the development of cooperative and competitive mechanisms in multi-agent learning.

The work by Park and Kim (2016) offers a novel approach to active player modeling using active learning techniques [3]. By conducting an experiment with twelve representative player types, the authors demonstrated the effectiveness of an active modeling algorithm in predicting opponent behavior. Strategies such as Tit-for-Tat (TFT) were analyzed, with insights into optimal play, cooperation, revenge, and forgiveness. The paper's emphasis on active learning and the potential application in video games contributes to a broader understanding of strategic behavior within the IPD, reflecting the complexity of cooperation and competition. This research aligns with the ongoing exploration of player modeling, cooperative strategies, and the development of predictive algorithms, enriching the field of game theory and multi-agent systems.

In the study of Multiagent Reinforcement Learning (MARL) within the Iterated Prisoner's Dilemma (IPD), the work of Vassilis Vassiliades and Chris Christodoulou (2010) presents a groundbreaking approach to fast cooperation through evolved payoffs [9]. By employing an evolutionary algorithm, the authors successfully evolved valid payoff structures that quickly encourage mutual cooperation. The results of the study reveal that these structures should contain a mixture of positive and negative values, with the positive values' magnitude being much smaller. This innovative approach to payoff design has significant implications for fostering cooperation in multi-agent environments, aligning closely with our exploration of reward strategies in the IPD.

In the exploration of cooperative behavior within the context of the alternating Prisoner's Dilemma, the work of Benjamin M. Zagorsky et al. (2012) provides pivotal insights into the dynamics of cooperation and forgiveness [6]. The paper introduces the innovative concept of the "Forgiver" strategy, demonstrating that forgiveness can lead to long-term gains, even in the presence of exploitation and noise. The authors meticulously calculate payoffs for pairwise encounters and classify strategies into distinct categories, including the novel "Forgiver" strategy that triumphs in the mixed equilibrium. This perspective on forgiveness and cooperation aligns with our study's focus on understanding the intricacies of rewards and decision-making in the IPD.

Lin, Bouneffouf, and Cecchi (2020) explored the behaviors of online learning algorithms, including Bandits, Contextual Bandits, and Reinforcement Learning [5]. Through a round-robin computer tournament, the authors investigated the dynamics of cooperation and defection, revealing that decision-making based solely on the current situation is ineffective in the IPD. This insight resonates with our experimental setup, underscoring the importance of considering historical interactions, particularly when employing Q-learning techniques. By highlighting the need for a sufficient number of steps within episodes to capture the historical context and patterns of play, the study informs our approach to modeling and learning within the IPD. The research also emphasizes clinical validations and applications in neuropsychiatric conditions, connecting artificial intelligence algorithms with human behaviors. The findings provide valuable insights into the complexities of social dilemma games and demonstrate the potential of online learning to bridge the gap between computational models and human psychology. Furthermore, the implications of the study offer guidance in structuring Q-learning experiments, ensuring that the number of steps included without episodes is appropriately calibrated to capture the nuanced dynamics of the IPD.

In the domain of social dilemmas, the study by Valerio Capraro (2013) presents a novel model of human cooperation that transcends classical approaches, focusing on intrinsic human tendencies [4]. By exploring one-shot social dilemmas without external controls, the research uncovers the inherent nature of cooperation and the influence of payoffs on cooperation rates. This perspective is particularly relevant for our experimental setup, emphasizing the importance of considering historical interactions and the complexity of factors influencing cooperation. Capraro's findings challenge traditional divisions between proself and prosocial types, offering a model that accurately predicts population average behavior.

This research situates itself amidst these studies, aiming to contribute further to the understanding of agent behaviors in the IPD using reinforcement learning methodologies.

3 Methodology

This section delves into the methodology employed in this research, elucidating the fundamental principles underpinning the Iterated Prisoner's Dilemma, the Q-learning algorithm, and the corresponding experimental setup.

3.1 Iterated Prisoner's Dilemma (IPD)

The Prisoner's Dilemma is a canonical example of a game analyzed in game theory. In its simplest form, two players simultaneously choose to either "Cooperate" or "Defect". The rewards they receive depend on the combined choice:

- If both players cooperate, they both receive a reward R.
- If both defect, they both get a punishment payoff P.
- If one cooperates while the other defects, the cooperator receives the "sucker's" payoff S, while the defector receives the "temptation" payoff T.

In the Iterated Prisoner's Dilemma, this game is played repeatedly, allowing strategies to evolve over multiple interactions. The reward matrix used in this research is as follows:

	Cooperate	Defect
Cooperate	$(3, 3)$	$(0, 5)$
Defect	$(5, 0)$	$(1, 1)$

The entries in the matrix represent the following scenarios:

1. $(3, 3)$: Both agents decide to cooperate. This is a mutual cooperation scenario where both agents receive a moderate reward, indicating the mutual benefit of cooperative behavior.
2. $(0, 5)$: The first agent cooperates while the second defects. The cooperator receives no reward, becoming the "sucker", while the defector reaps a high reward of 5, exploiting the cooperator's trust.
3. $(5, 0)$: The roles are reversed from the previous scenario. The first agent defects, benefiting from the second agent's cooperation.
4. $(1, 1)$: Both agents choose to defect. This scenario represents mutual distrust or competitiveness, resulting in a minimal reward for both.

3.2 Nash Equilibrium

Nash Equilibrium, named after the mathematician John Nash, is a fundamental concept in game theory. In a game, a Nash Equilibrium is achieved when every player's strategy is optimal given the strategies chosen by all other players. In other words, no player has an incentive to deviate from their strategy, assuming that the strategies of the other players remain unchanged.

Mathematically, in a two-player game: Let s_1^* and s_2^* be the strategies chosen by Player 1 and Player 2, respectively. The strategies s_1^* and s_2^* constitute a Nash Equilibrium if and only if:

$$U_1(s_1^*, s_2^*) \geq U_1(s_1, s_2^*)$$

for all s_1 in the strategy set of Player 1, and

$$U_2(s_1^*, s_2^*) \geq U_2(s_1^*, s_2)$$

for all s_2 in the strategy set of Player 2, where U_1 and U_2 are the utility (or payoff) functions for Player 1 and Player 2, respectively.

In the traditional Prisoner's Dilemma, the Nash Equilibrium is achieved when both players choose to "Defect". Let's understand this with the help of the reward matrix:

	Cooperate	Defect
Cooperate	(R, R)	(S, T)
Defect	(T, S)	(P, P)

Given the typical payoff hierarchy of $T > R > P > S$:

- If Player 1 believes that Player 2 will cooperate, Player 1's best response is to defect, as $T > R$.
- If Player 1 believes that Player 2 will defect, Player 1's best response is still to defect, as $P > S$.

The same logic applies to Player 2. Consequently, both players have a dominant strategy to defect, making "Defect, Defect" the only Nash Equilibrium in the game. This is paradoxical because both players would be better off if they cooperated, but the rational choice for both, given the other's strategy, is to defect.

3.3 Nash Equilibrium in IPD

The Iterated Prisoner's Dilemma (IPD) introduces a temporal dimension to the game, where players interact over multiple rounds. This allows for a richer strategy space because decisions can be based on previous outcomes, allowing for cooperation to potentially emerge even if it's not the Nash equilibrium in any single round[12].

In many IPD studies, strategies compete in an evolutionary setting. Those that earn higher payoffs are more likely to be 'replicated' in subsequent generations, it is called Evolutionary Dynamics [18]. This dynamic can lead to cycles where certain strategies become dominant, only to be replaced by others.

One of the most famous strategies in IPD is "Tit-for-Tat" (TFT). Starting with cooperation, TFT then mirrors the opponent's last action in subsequent rounds[19]. TFT has been shown to be an "evolutionarily stable strategy" (ESS) under certain conditions[13], meaning that once a population adopts TFT, no single alternative strategy can outperform it.

Given the dynamic nature of IPD and the reliance on history, the concept of Nash equilibrium becomes more nuanced. If both players adopt the TFT strategy, neither has an incentive to unilaterally deviate in any given round, knowing that such a deviation will lead to retaliation and lower payoffs in future rounds.

The likelihood of future interactions influences behavior in IPD. The "shadow of the future" can promote cooperation.[7] If players expect many future interactions (or if interactions have no predetermined end), the future rewards of cooperation make defection less appealing. Conversely, if the end of interactions is known and near, defection becomes more likely.

In the IPD, the repeated nature of interactions and the ability to condition behavior on past outcomes introduce dynamics absent in a one-shot game. While defection is the dominant strategy in a single round, strategies like TFT that condition on past behavior can promote cooperation in the IPD. The shadow of the future, or the expectation of ongoing interactions, further amplifies this cooperative behavior. However, the precise outcome in terms of which strategies dominate or how cooperation levels evolve depends on the specific dynamics, parameters, and strategy set considered.

3.4 Q-learning

Q-learning is one of the foundational algorithms in the field of reinforcement learning (RL). Operating without a model of the environment, it provides a mechanism for an agent to learn optimal decision-making strategies by interacting with its surroundings.

In reinforcement learning, an agent seeks to determine the best action to take in each state to maximize its cumulative rewards over time. Q-learning achieves this by estimating the value, or the expected future reward, of taking a particular action in a given state. This is represented by the Q-value.

The Q-value, denoted as $Q(s, a)$, represents the expected cumulative reward of taking action a in state s and following the optimal policy thereafter. The goal of Q-learning is to accurately learn these Q-values for all state-action pairs.

Q-values are stored in a table, known as the Q-table. Each entry in this table corresponds to the Q-value of a particular state-action pair. Initially, this table is often populated with zeros, representing no knowledge about the environment.

As the agent interacts with the environment, it uses its experiences to update the Q-values. The update is governed by the Q-learning formula:

$$Q(s, a) \leftarrow (1-\alpha) \cdot Q(s, a) + \alpha(r + \gamma \cdot \max_{a'} Q(s', a'))$$

Where:

- α (learning rate): Determines how much of the new Q-value estimate we adopt. A higher α means the agent gives more importance to recent experiences.
- r (reward): The immediate reward received after acting a in state s.
- γ (discount factor): Controls the agent's consideration for future rewards. A value of 0 makes the agent short-sighted, while a value close to 1 makes it prioritize long-term reward.Just as the shadow of the future can promote cooperation in IPD, a high γ in Q-learning can encourage actions that might have lower immediate rewards but lead to higher cumulative rewards in the long run[22].
- s' is the next state.
- a' is the next action.

Over time, with sufficient exploration and consistent learning, the Q-values converge towards the true values, and the agent learns an optimal policy.

In the context of this research, Q-learning enables the agents to navigate the complexities of the Iterated Prisoner's Dilemma, adjusting their strategies based on both their individual experiences and the combined outcomes of their interactions.

3.5 Agent Design

The agents in this study are designed using Q-learning, a model-free reinforcement learning algorithm. Here's a detailed breakdown of the agent design:

Q-table Initialization: The Q-table, a two-dimensional matrix, maps each state-action pair to a Q-value. Initially, all values in this table are set to zero, indicating no prior knowledge about the environment's dynamics.

Action Selection: Agents use an epsilon-greedy strategy for action selection. With a probability equivalent to the exploration rate, an agent takes a random action (exploration). Otherwise, it chooses the action with the highest Q-value for the current state (exploitation). This trade-off ensures that the agent explores various strategies while learning the best ones.

Reward Mechanism: The agents receive rewards based on their actions and the actions of their opponent, as determined by the IPD reward matrix. However, in updating their Q-values, agents consider a combination of individual and team rewards. The combination is governed by the individual and team reward weight factors.

Q-value Update: After observing the reward for an action in a particular state, the Q-value of that state-action pair is updated using the Q-learning update rule. This rule incorporates the immediate reward and the maximum Q-value of the next state, allowing the agent to learn about the long-term implications of its actions.

Exploration Decay: To ensure convergence to an optimal strategy, the exploration rate decays over time. This decay ensures that as the agent gains more experience, it relies more on its learned knowledge (exploitation) and reduces random actions (exploration).

The design of the agents ensures they can adapt and evolve their strategies over time, optimizing their decisions based on both individual outcomes and the collective good.

3.6 Experimental Setup

The environment, modeled using the PettingZoo framework, simulates the interactions between two agents in the IPD. Each agent, implemented as a Q-learning agent, aims to maximize its cumulative reward over a series of rounds.

3.6.1 Action Space:

The action space is discrete with two possible actions for each agent: "Cooperate" (0) or "Defect" (1).

3.6.2 State Representation:

Each state is represented as a discrete value, capturing the combined action of both agents from the previous round.

1. State 0: Represents the scenario where both agents chose to "Cooperate" in the previous round. This mirrors the reward matrix entry of $(3, 3)$.
2. State 1: Denotes the scenario where the first agent chose to "Cooperate" and the second agent decided to "Defect" in the previous round. This corresponds to the reward matrix entry of $(0, 5)$.
3. State 2: Represents the opposite of State 1, where the first agent chose to "Defect" while the second agent "Cooperated" in the previous round. This is analogous to the $(5, 0)$ entry in the reward matrix.
4. State 3: Depicts the situation where both agents chose to "Defect" in the last round, aligning with the $(1, 1)$ reward matrix entry.

3.6.3 Training Parameters:

The agents' learning mechanisms were parameterized as follows:

- Learning rate (α) : 0.1
- Discount factor (γ) : 0.95
- Initial exploration rate (ϵ) : 1.0
- Exploration decay rate: 0.003

During the experiment, both agents were initialized with identical parameters, ensuring a consistent baseline for comparison. The primary variable of interest in this research was the balance between individual rewards and team rewards. To investigate the implications of this balance, the individual reward weights and the team reward weights were systematically adjusted.

3.6.4 Game Dynamics:

We set the Number of Rounds Per Episode to 100. This value was chosen to strike a balance between giving agents ample opportunity to adjust strategies based on prior interactions within the episode and computational efficiency. Setting this value too low might deprive the agents of the chance to learn from immediate past interactions, thereby possibly leading to suboptimal strategies. Conversely, setting it too high can result in diminishing returns in learning, where agents might get stuck in repetitive patterns without meaningful strategy evolution. Furthermore, a very high value would amplify the computational demand without significantly enhancing the learning outcomes.

We set the Total Number of Episodes (num_episodes) to 2000. This value was determined based on empirical findings from preliminary experiments. We discerned that the decision-making dynamics of the agents began to stabilize after approximately 1500 training sessions. Therefore, to ensure the robustness of the learned strategies and to account for any late-stage refinements in agent behaviors, we extended the training to 2000 episodes. This provides a buffer, ensuring that the agents have indeed converged to a stable strategy and are not merely in a transient phase [23].

The episode-based structure, combined with multiple rounds within each episode, ensures that agents can recalibrate their strategies in light of both immediate past interactions (within the episode) and longer-term outcomes (across episodes).

3.6.5 Q-table Updates:

The Q-table is updated at the conclusion of each round within an episode. This granular update approach allows the agents to refine their strategies continuously, drawing from the immediate feedback of their actions. The Q-value update is carried out using the standard Q-learning update rule, which amalgamates immediate rewards and the estimated value of future states [24]. The granularity of these updates ensures that the agents' learned strategies are representative of round-by-round interactions, adhering to the foundational mechanics of the IPD [13].

3.6.6 Experiment Procedure:

To ensure robustness and reliability in the findings, the experiment was conducted in multiple stages:

1. The individual reward weights and team

reward weights were adjusted ten times throughout the experiment.

2. For each set of weight adjustments, ten separate rounds of experiments were conducted. This repetition was critical to offset potential anomalies or outliers in any single experiment, thereby ensuring the observed behaviors were consistent and reproducible.

The overarching goal of these experiments was to understand the emergent behaviors and strategies of the agents under various reward weight configurations and to discern patterns that might provide insights into the broader dynamics of cooperative systems.

3.7 Visualization of results:

Visualizing the results and dynamics of reinforcement learning agents is essential to obtain a comprehensive understanding of their behavior and the strategies they evolve over time. In this research, visualization is leveraged in two primary ways to depict the performance and internal state of the agents.

3.7.1 Time Evolution of Agent Rewards:

To understand how agents adapt and refine their strategies over multiple episodes, it's crucial to plot the rewards they accumulate over time. However, since the raw reward data can be noisy, a moving average is employed to smooth the rewards, providing a clearer view of the underlying trends.

Moving Average of Rewards: Purpose: Smoothens the reward signal over time, making trends more discernible. Methodology: The moving average is computed using a window size of 50. This window can be adjusted to increase or decrease the smoothing level, depending on the granularity of insights desired.

Agent Rewards Visualization: For each agent, the smoothed rewards, representing the arithmetic mean of the accumulated rewards for every episode, are plotted over time. Each episode's average reward is computed by taking the mean of rewards from all rounds in that specific episode. This approach offers a consolidated view of the agent's performance over an episode, rather than individual round outcomes. This visualization aids in understanding the trajectory of individual agent strategies and their corresponding performance dynamics.

Team Rewards Visualization: The visualization of team rewards represents the arithmetic mean of the combined rewards of both agents for each episode. As with individual agents, this average is derived from the rewards of all rounds within a particular episode, providing an overview of the holistic outcome of their interactions. By evaluating this graph, insights into the global, cooperative dynamics of the agents can be gleaned, showcasing how their joint actions lead to collective outcomes over time.

3.7.2 Action Preference Visualization:

To gain a more nuanced understanding of the agents' evolving strategies during training, we visualized the difference in Q-values between the two available actions for each state. Specifically, the measure of Action Preference is derived by subtracting the Q-value for "Defect" (Action 1) from the Q-value for "Cooperate" (Action 0) for each state.

This Action Preference metric offers a straightforward way to interpret the agents' inclination towards cooperation or defection. A positive value indicates a bias towards cooperation, while a negative value suggests a preference for defection.

The horizontal line at $y = 0$ serves as a reference, demarcating the boundary between preference for cooperation (above the line) and defection (below the line). By analyzing these plots, readers can discern patterns in agent strategies across various states and understand how different circumstances (states) influence the agents' proclivity to cooperate or defect.

4 Results

4.1 Predicted Results Without Considering Collective Rewards

Based on the methodology and the structure of the reward mechanism, we can formulate predictions regarding the agents' behaviors. Understanding these predictions is paramount, as it provides a foundation against which the experimental results can be compared, allowing for a comprehensive assessment of the model's efficacy and insights into the dynamics of cooperative behaviors in multi-agent systems.

In scenarios where agents prioritize exclusively individual rewards, their strategies are influenced not just by the immediate benefits, but also by the iterative nature of their interactions in the IPD. While a single instance of the Prisoner's Dilemma suggests defection as the dominant strategy, the repeated structure of the IPD brings about a richer set of dynamics.

In the IPD, agents can condition their choices on past interactions. This means that while "Defecting" might provide an immediate higher reward in a single round, strategies like "Tit-for-Tat" (TFT) that respond based on the opponent's previous actions can foster cooperation over multiple rounds. The anticipation of future interactions — termed the "shadow of the future" — can further reinforce such cooperative tendencies.

Prediction: It is not straightforward to predict that both agents will always converge toward always "Defecting". While defection remains a strong strategy in the short term, the ongoing nature of interactions might lead to periods of cooperation. Nevertheless, without any weightage given to team rewards, the likelihood of sustained cooperation remains lower than in scenarios where collective benefits are considered.

4.2 Actual Results Without Considering Collective Rewards

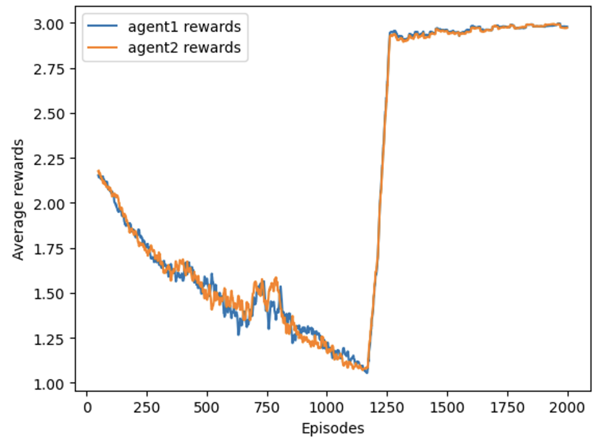


Figure 1: Average rewards for agents

Figure 1 illustrates the agent rewards over time. During the early training phase, up to approximately episode 1150, there's a discernible decline in the average rewards for both agents. As the episode count approaches 1150, the average reward seems to stabilize at a value close to 1.05. This pattern in rewards suggests that both agents predominantly lean towards a strategy of mutual defection during this phase. Such behavior, where both agents consistently choose to defect, is consistent with the Nash equilibrium observed in traditional game theory interpretations of the Prisoner's Dilemma. Following this state, there's an intriguing phase from episode 1150 to roughly episode 1250. In this window, the average rewards experience a sharp ascent, reaching a peak value of approximately 5.5 by episode 1250. Such a surge in rewards indicates a shift in agent strategies, potentially towards exploring more cooperative behaviors.

Post episode 1250, as the training progresses further towards episode 2000, the average rewards display a more gradual increase, eventually converging to a value of 3 by the end of episode 2000. In the context of the Prisoner's Dilemma, an average reward of 3 corresponds to mutual cooperation, suggesting that the agents have learned the benefits of collaboration over sustained interactions.

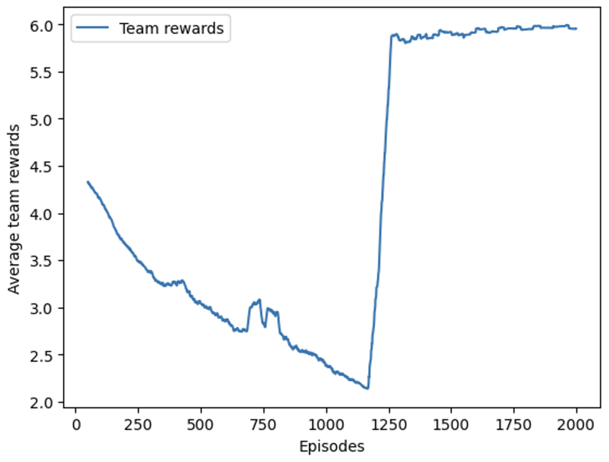


Figure 2: Average rewards for team

Figure 2... Average team rewards, this metric is an aggregation of the rewards obtained by both agents. Notably, the pattern mirrors the trends observed in the individual rewards, reinforcing the dynamics of their interactions. Initial episodes see a dip in team rewards, hinting at a shared strategy of mutual defection. However, as the episodes progress beyond 1150, there's a marked upswing, culminating in a convergence around a reward value that reflects cooperation. This progression illustrates the agents' evolving strategies and underscores the complex interplay between individual choices and collective outcomes in the Iterated Prisoner's Dilemma.

Figure 3, Figure 4... Figures analyzing Agent 1 and Agent 2's action preferences offer insight into their strategic decision-making throughout the episodes. The metric, calculated as the difference between the Q-values for Cooperate and Defect, clarifies the agents' tendencies in various states. For context, the four states represent different combinations of previous round decisions:

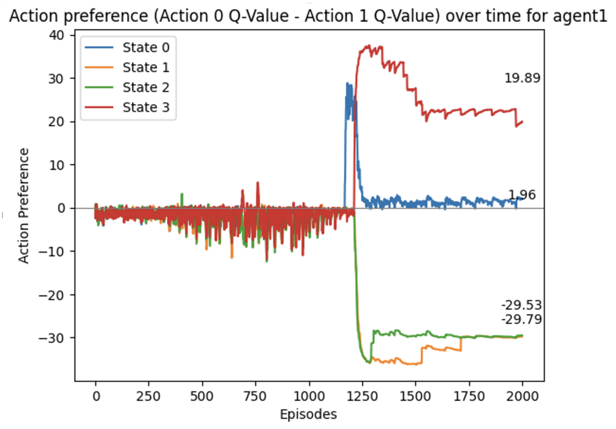


Figure 3: Action preference for agent 1

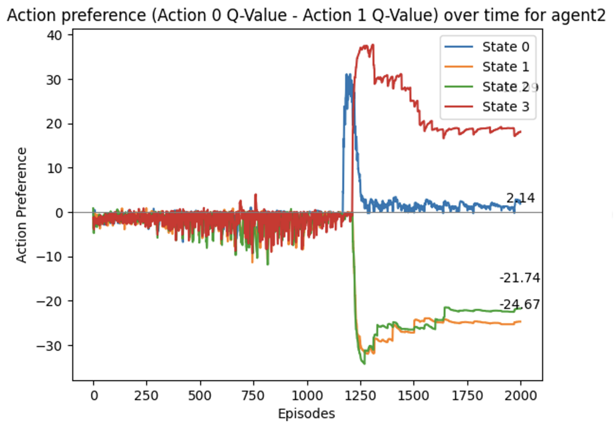


Figure 4: Action preference for agent 2

- State 0: Both agents chose to “Cooperate”.
- State 1: Agent 1 “Cooperated”, while Agent 2 “Defected”.
- State 2: Agent 1 “Defected”, while Agent 2 “Cooperated”.
- State 3: Both agents chose to “Defect”.

During the initial episodes, particularly up to episode 1150, the action preference values oscillate primarily between 0 and -10. A consistent value below zero across states indicates a pronounced tendency for both agents to defect, irrespective of the previous round’s decisions. This pattern substantiates the observed decline in agent rewards during this phase, suggesting a mutual defection strategy in line with the Nash equilibrium of the classic Prisoner’s Dilemma.

Post the 1150th episode, a significant shift in strategy is evident. For states 0 and 3, action preference values surge beyond 30, indicating a strong inclination for cooperation. However, this spike is transient for state 0, with values plummeting swiftly and stabilizing close to 2.14. In contrast, state 3 experiences a mild decline, settling around a value of 20, suggesting a predominant cooperation strategy for this state.

States 2 and 4 exhibit mirrored dynamics post the 1150th episode. Their action preference values sharply descend, finally consolidating between -20 and -30. This pronounced negative inclination is indicative of a consistent strategy of defection when either agent had previously defected while the other cooperated.

In essence, these figures illustrate the intricate dance of strategies that the agents navigate, with periods of mutual defection giving way to more nuanced interactions as the episodes progress.

The predicted outcome rightly anticipated periods of defection, especially in the early stages, it underestimated the propensity for agents to discover cooperative strategies over extended interactions. The agents’ behavior underscores the richness of the IPD’s dynamics and the capacity for strategies like TFT and GTFT to emerge even when individual rewards are prioritized.

4.3 Summary of strategies

State 0 (Both agents previously “Cooperated”):
The action preference values stabilize close to 2.14, suggesting a slight inclination for cooperation by both agents. This implies that when both agents have a history of mutual cooperation, they are likely to continue cooperating, albeit with a reduced certainty compared to other states. This behavior is reminiscent of the “Tit-for-Tat” (TFT) strategy, one of the most famous approaches in the IPD. TFT starts by cooperating and then replicates the opponent’s last action in subsequent moves. When both agents cooperate, TFT will continue to cooperate in the next round.

State 1 and State 2 (One agent “Cooperated” and the other “Defected”):
The action preference values stabilize in the negative range between -20 and -30. This indicates a strong tendency for agents to defect after having betrayed or been betrayed in the previous round. This reaction aligns with the punitive aspect of the TFT strategy. If one agent defects while the other cooperates, TFT dictates that the cooperating agent should defect in the next round as a form of retaliation. It aims to “teach” the defecting agent the cost of betrayal. The agents’ tendency to defect after one has betrayed the other mirrors this aspect of TFT.

State 3 (Both agents previously "Defected"):
The action preference values stabilize around 20, signifying a clear propensity for both agents to cooperate. This is particularly intriguing, as it suggests that after mutual betrayal, both agents recognize the suboptimal outcomes of continuous defection and pivot towards a cooperative strategy. This strategy is similar to "Generous Tit-for-Tat" (GTFT), a variant of TFT. GTFT occasionally cooperates even after the opponent has defected, introducing a level of forgiveness. The idea behind GTFT is to avoid long sequences of mutual defections that can occur when both agents are using TFT. By cooperating after mutual defection, the agents appear to be employing a strategy akin to GTFT to break the cycle of defections and steer the game back towards a cooperative state [12].

4.4 A Comprehensive Exploration of different weights

Having analyzed agent behaviors when driven solely by individual rewards, we now venture into exploring the dynamics introduced by varying balances between individualistic and collective incentives. By systematically modulating these weightings, this section seeks to uncover the interplay between self-interest and group welfare, and how subtle shifts in this balance can impact overarching agent strategies in the Iterated Prisoner's Dilemma.

4.4.1 The Inherent Randomness in Q-Learning

The outcomes of Q-learning exhibit variability due to several intrinsic factors. Chief among them are parameters like the exploration rate, learning rate, and the initial Q-value settings. These parameters, in tandem, influence the agents' decision-making processes throughout their learning journey.

For example, the exploration rate dictates the probability with which agents might opt for a random action, introducing variability in their learning trajectories. Different sequences of exploratory actions, even under identical settings, can lead agents down divergent paths.

To ensure the reliability and robustness of our findings amidst this inherent randomness, we conducted 10 independent training runs for each parameter configuration. This approach not only provides a comprehensive understanding of potential agent behaviors but also mitigates the risk of basing conclusions on singular, potentially anomalous results.

Detailed outcomes of these multiple runs are cataloged in the appendix for those seeking a deeper dive into the data.

4.4.2 Nash Equilibrium with collective rewards

Given the nature of the Prisoner's Dilemma, the rewards can be represented in a reward matrix. Let's denote the reward matrix as:

$$R = \begin{bmatrix} R & S \\ T & P \end{bmatrix}$$

- R is the reward for mutual cooperation.
- S is the temptation to defect when the opponent cooperates.
- T is the sucker's payoff, i.e., the reward for cooperating when the opponent defects.
- P is the punishment for mutual defection.

Assuming weights a and b for individual and team rewards, respectively, we'll adjust the rewards in the matrix:

$$R' = \begin{bmatrix} a \cdot R + b \cdot (R + R) & a \cdot S + b \cdot (S + T) \\ a \cdot T + b \cdot (T + S) & a \cdot P + b \cdot (P + P) \end{bmatrix}$$

The Nash Equilibrium can be found by determining when the expected utility for cooperating is equal to the expected utility for defecting. when the expected utility for cooperating is equal to the expected utility for defecting.

We assume that: p as the probability that the opponent will cooperate. $1 - p$ as the probability that the opponent will defect.

The expected utility U_C for an agent choosing to cooperate is:

$$\begin{aligned} U_C = {} & p \times (a \cdot R + b \cdot (R + R)) \\ & + (1 - p) \times (a \cdot S + b \cdot (S + T)) \end{aligned} \quad (1)$$

The expected utility U_D for an agent choosing to defect is:

$$\begin{aligned} U_D = {} & p \times (a \cdot T + b \cdot (T + S)) \\ & + (1 - p) \times (a \cdot P + b \cdot (P + P)) \end{aligned} \quad (2)$$

For a Nash Equilibrium in pure strategies, U_C should be equal to U_D. By setting the two equations equal and solving for p, we can determine the probability in which the opponent will cooperate to make both strategies equally desirable.
For our specific rewards matrix, we can get:

$$p = \frac{3 - 4a}{3 - a}$$

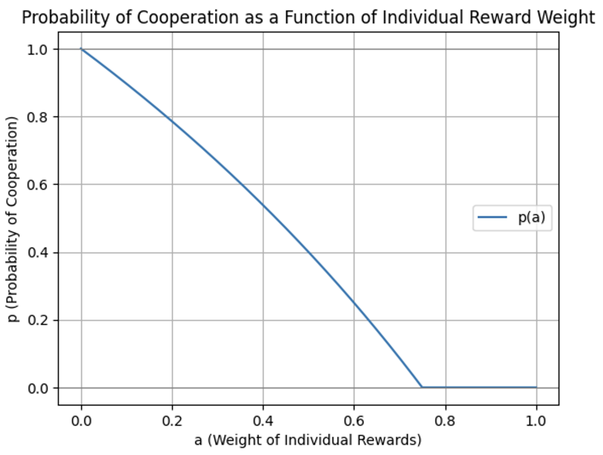


Figure 5

Figure5… Although $a \in [0,1]$, it's essential to note that a probability p cannot naturally be negative. In this context, a negative value for p would essentially mean a 0 probability of cooperation. So in practical terms, the range for p is $[0,1]$, where values of a closer to 1 will push p towards its lower bound (0), and values of a closer to 0 will push p towards its upper bound (1).

4.4.3 Predicted outcomes

Building on the established understanding of the Nash Equilibrium in the context of team rewards, we propose a novel method to predict the outcomes of different weights of individual rewards by analyzing the action preferences of the agents.

Specifically, we interpret the probability of cooperation, denoted by p, as a ratio reflecting the agents' action preference for cooperation. This ratio can be expressed as the area of the portion above the x-axis (indicating a preference for cooperation) divided by the overall area in the visualization of action preferences.

$$p = \frac{3-4a}{3-a} = \frac{\text{area above } x\text{-axis}}{\text{overall area}}$$

By utilizing this approach, we can identify the value of a, the weight of individual rewards, when $p = 0$, resulting in $a = 0.75$, we define a critical threshold for cooperation. This threshold delineates a critical point in the decision-making process, wherein agents transition from a propensity to defect to an emergent pattern of cooperative behavior. When $a < 0.75$ the action preference value for agents tends to be greater than 0. This inclination signifies that the agents might start to cooperate with each other, rather than defect. In predicting this behavior, the phenomenon we anticipate to observe specifically in the action preference image is a significant upward shift in the action preference curve.

It is essential to note that the area used in the formula $p(a)$ should not be the area when the behavioral preference curve has reached a state of convergence. Influencing factors such as the "Shadow of the Future," the eventual learned strategy of the agent, and other nuanced dynamics in the Iterated Prisoner's Dilemma contribute to the outcome. Our calculation of the Nash equilibrium, accounting for team rewards, only considers the reward matrix. Consequently, the area calculated by the equation should represent the area of the behavioral preference curve before it has changed drastically. In situations like the case of $a = 1.0$, the entire IPD training process does not apply the traditional Nash equilibrium uniformly. The predictions of the traditional Nash equilibrium are only met in the early stages of training when the agent has not yet discovered an optimal strategy.

This concept connects theoretical game-theoretic insights with the empirical observations from our simulations. It offers a practical way to calibrate our models and predict the agents' behavior under varying reward structures.

Further empirical validation and sensitivity analysis will enhance the robustness of this approach, opening new avenues for research into the complex dynamics of cooperation and defection within agent-based models.

4.5 Actual results of different weights

4.5.1 Individual rewards Weight $a > 0.75$

In the realm of our investigation, the weight for individual rewards, denoted by a. Initially, we have observed the outcomes at $a = 1$, and now we shall sequentially reduce the value of a in intervals of 0.1, down to $a = 0.75$, conducting experiments at each step.

Recognizing the inherent randomness that characterizes the Q-learning process, it was imperative to ensure the robustness and representativeness of our findings. Hence, we conducted ten rounds of experiments for each specified value of a, meticulously selecting those outcomes that exhibit typical characteristics reflective of the underlying agent behaviors. When we conducted experiments with $a = 0.8$ and $a = 0.9$. The outcomes revealed two distinct classes of results, each of which holds significance in understanding the dynamics of multi-agent learning in the Iterated Prisoner's Dilemma (IPD).

1. *Class 1: Cooperation*
Figure 6...The behavior trajectory in this class mirrors the patterns of convergence towards mutual cooperation seen previously. After a series of episodes, the agents appear to recognize the collective benefits of cooperation and align their strategies accordingly.

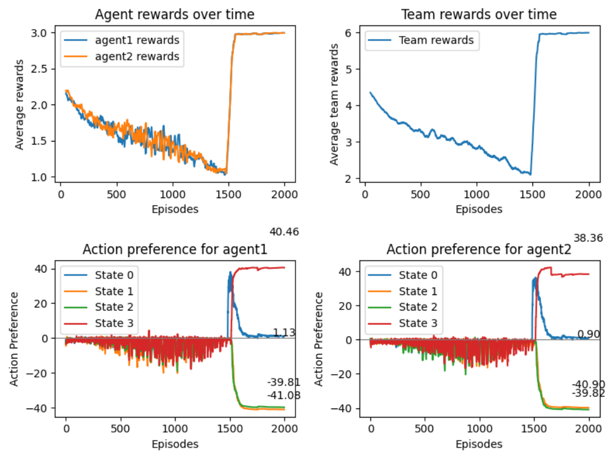


Figure 6: Action preference $a = 0.9$

2. *Class 2: Defect*
Figure 7...The other class of outcomes is marked by a failure to find strategies leading to cooperation. Instead, the action preferences of both agents converge towards defection. This result aligns with the traditional game theory understanding of the Prisoner's Dilemma, where mutual defection is the dominant strategy.

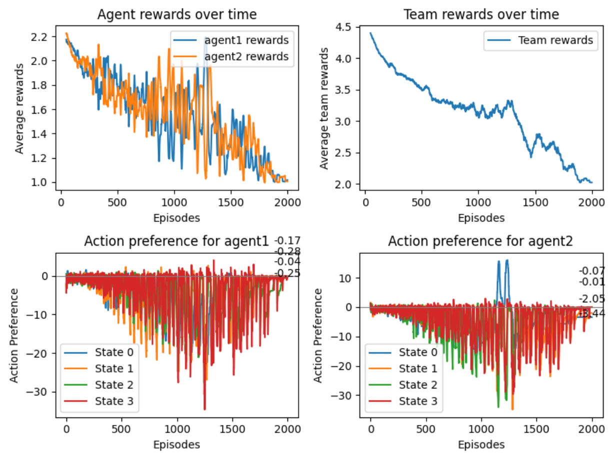


Figure 7: Action preference $a = 0.8$

Despite the systematic reduction in the weight allocated to individual rewards, ranging from $a = 1.0$ down to $a = 0.8$, the agent's choice of action did not exhibit significant alterations. According to our calculation of the Nash equilibrium (where $a = 0.75$), the current results are within the predicted range. When $a > 0.75$, the outcomes of the trained agents do not change significantly.

In the visualization of action preference, it can be observed that the area above the X-axis is greater than 0. This phenomenon arises within the context of Q-learning training, where agents may randomly select actions based on an exploration rate. However, this occurrence should not be construed as evidence that our Nash equilibrium calculation for the Prisoner's Dilemma, considering the individual reward weights, is incorrect. Because of the underlying dynamics of the learning algorithm and the stochastic nature of exploration throughout the training phase, it is still valid to represent the area above the X-axis as zero when it is small.

4.5.2 Individual rewards Weight $a < 0.75$

As we delve into the experimental results under specific weight values of $a = 0.7$, a discernible deviation from the previous outcomes emerges, where $a > 0.75$. We can observe the following phenomena:

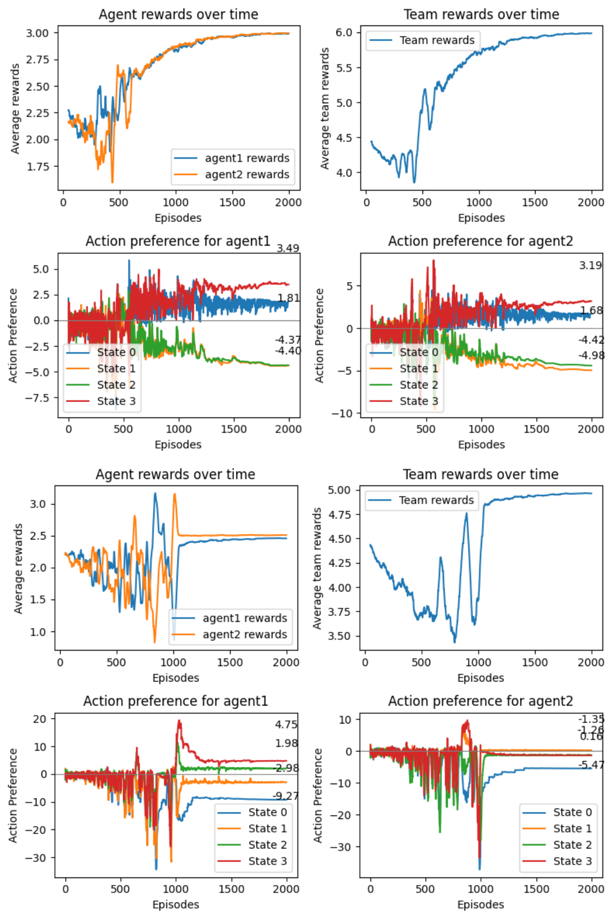


Figure 8: Action preference $a = 0.7$

- *Changes in convergence results:* Figure 8…Unlike previous results, instances of mutual betrayal have been eliminated under weights of $a = 0.7$. However, occurrences of unilateral betrayal have emerged. As shown in figure 8, there is a case of unilateral defection by turns at $a = 0.7$. When agent1 is betrayed, agent1 will choose defect in the next round. On the contrary, after agent2 betrays agent1, it tends to choose to cooperate. This causes agent2 to be betrayed in the next round. In this way, they form a balance of betraying and being betrayed in turn. In this case, although the agents will avoid mutual defection, they still have not found the optimal equilibrium (mutual cooperation), but have fallen into a local optimum. This suggests a subtle shift in agency behavior rather than an absolute transition to cooperation.
- *Increased Amplitude of Average Agent Reward:* The amplitude of the curves for the average agent reward is noticeably larger compared to the previously studied cases. As the weight of the team reward intensifies, agents gradually accommodate a strategy that allows for occasional betrayal, thereby amplifying the team reward. This change reflects a greater willingness to embrace cooperation even at the risk of betrayal, yielding a more substantial shift in amplitude.

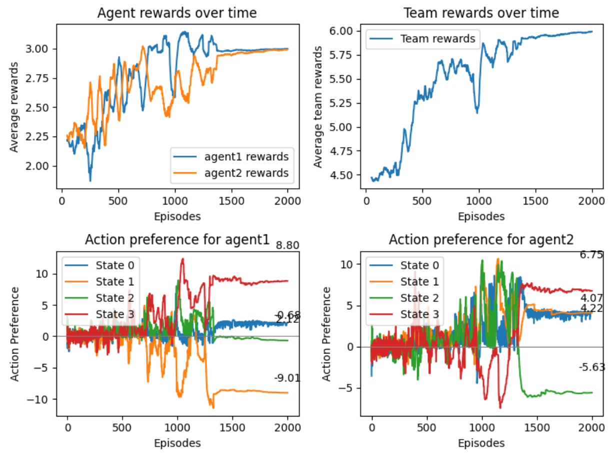


Figure 9: Action preference $a = 0.6$

We continue to reduce the weight of the individual reward a, and we observe the following changes:

- *Continuous Rise in Team Rewards:* Figure 9…When $a = 0.6$, the team rewards exhibit a consistent rise, even during the early exploration stages. Unlike scenarios with $a \geq 0.7$, where a period of mutual betrayal is often observed (consistent with the Nash equilibrium), this consistent increase in rewards represents a distinct pattern.
- *Accelerated Transition Time:* The time frame for the dramatic change in action preferences is observed to occur earlier in these experiments. While previous results with $a > 0.75$ typically witnessed significant shifts around the episode count of 1000, the experiments with $a = 0.6$ display a marked advancement in this transition. Specifically, the time point moves to episodes 250 and 500 for $a = 0.6$. This acceleration signifies that the agents can identify a cooperation strategy much more rapidly.

Continuing our investigation into the effects of further reducing individual reward weights, we find that the experimental outcomes at $a = 0.5$ are not markedly divergent from those observed at $a = 0.6$. However, some subtle yet significant changes emerge:

- *Upward Shift in Action Preference Curves:* A distinct upward shift in the action preference curves across each state becomes evident. This shift signifies a considerable increase in the agents' inclination to opt for cooperation. It not only reaffirms the trend toward cooperative behavior as the weight of individual rewards decreases but also demonstrates the agents' adaptability in aligning their strategies to the new weighting conditions.

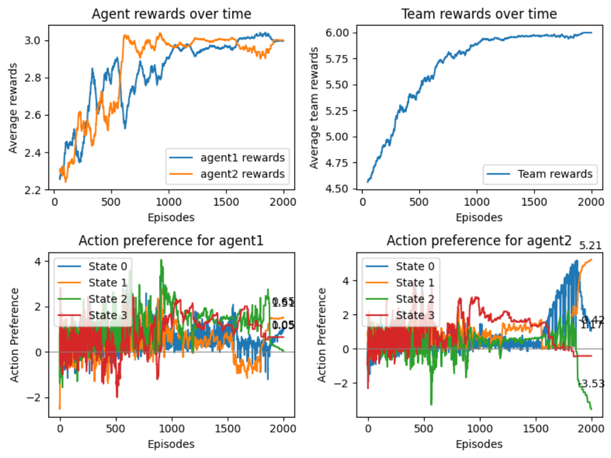


Figure 10: Action preference $a = 0.5$

- *The Emergence of Pure Cooperative Strategies:* As the action preference curve converges, more nuanced strategy combinations materialize. For instance, at $a = 0.5$, a "pure cooperation strategy" emerges, whereby both agents consistently elect to cooperate regardless of the prevailing state. This cooperative alignment contrasts starkly with the previous findings and reveals a potential optimal cooperation zone within the selected weight range. Moreover, another observed strategy in figure 10 comprises one agent employing a purely cooperative approach while the other adheres to the "Tit-for-Tat" strategy. This combination underscores the flexibility in agents' strategy selection and the influence of the weighting factor in shaping these decisions.

As the experiment progressed with a continued downward adjustment of the individual reward weights a, the following patterns emerged:

- *Stabilization on Pure Cooperation:* The final strategy for the convergence of the behavioral preference curves began to stabilize on a purely cooperative strategy. This pattern reflects an overarching trend towards collaboration and could signal an equilibrium state where the weight of individual rewards has diminished to a point that no longer incentivizes defection.

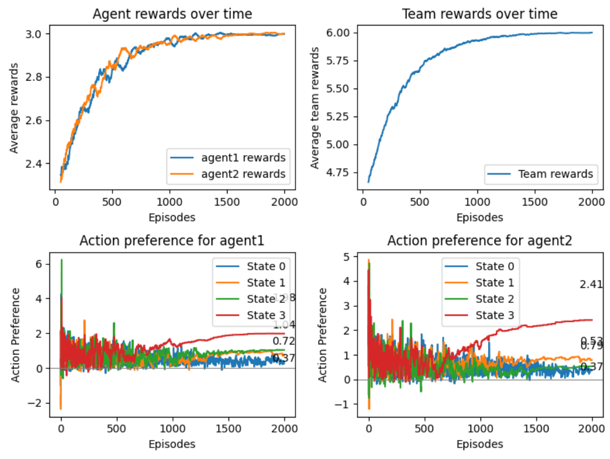


Figure 11: Action preference $a = 0.3$

Through a comprehensive series of iterative Prisoner's Dilemma experiments with varying individual reward weights, we have elucidated an extensive range of phenomena and behavioral variations. A salient transition favoring cooperation is discernible within the range of $a = 0.8$ to $a = 0.7$, aligning closely with our theoretically derived critical value of $a = 0.75$. As we continue to decrease the individual reward weights a, the observed shifts in agent behaviors and outcomes conform to our predictions, reinforcing the validity of our underlying models.

However, it is imperative to recognize that the computed turning point of $a = 0.75$ remains a theoretical construct. The data amassed in our experiments, though suggestive of this transition, are not unequivocally conclusive in verifying its exactitude. Furthermore, the intrinsic uncertainty within the Q-learning process introduces additional complexities in confirming this theoretical value. Such uncertainty underscores the nuanced interplay between theoretical predictions and empirical observations, affirming the need for further research to bolster our understanding of the underlying dynamics and to refine the accuracy of our models within the context of multi-agent interactions and the Prisoner's Dilemma.

4.5.3 Verify Nash Equilibrium with collective rewards

To pinpoint the turning point and validate the Nash equilibrium formula we postulated, we adopted a strategic analytical approach more precisely. This approach involves extracting the areas above and below the X-axis, representing the cumulative action preferences of both agents across the episodes of interest.

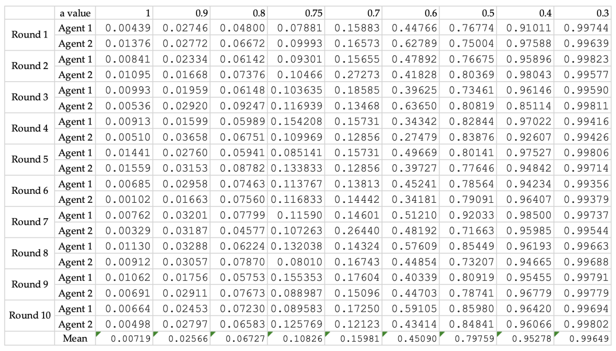
Our scope of analysis was particularly narrowed to episodes between 50 and 250. The rationale behind this selection stems from observed behavioral trends: the initial 50 episodes are characterized by agents displaying erratic actions, making the extraction of meaningful strategic patterns challenging. Conversely, any episode beyond the 250th mark often presents scenarios where agents have potentially settled on a distinct strategy. Thus, we chose this specific range to better understand the evolving strategies before they become fixed.
By evaluating the ratio:

$$\frac{\text{area}_p}{\text{area}_p - \text{area}_n} = p(a) = \frac{3-4a}{3-a}$$

where

- area_p : Area above X-axis
- area_n : Area below X-axis

Acknowledging the inherent variability and unpredictability in Q-learning, our methodology further includes ten experimental replications for each weight scenario. This comprehensive approach aims to offset random fluctuations and present a robust representation of the underlying agent behaviors and strategies.



	a value	1	0.9	0.8	0.75	0.7	0.6	0.5	0.4	0.3
Round 1	Agent 1	0.00439	0.02746	0.04800	0.07881	0.15883	0.44766	0.76774	0.91011	0.99744
	Agent 2	0.01376	0.02772	0.06672	0.09993	0.16573	0.62789	0.75004	0.97588	0.99639
Round 2	Agent 1	0.00841	0.02334	0.06142	0.09301	0.15655	0.47892	0.76675	0.95896	0.99823
	Agent 2	0.01095	0.01668	0.07376	0.10466	0.27273	0.41828	0.80369	0.98043	0.99577
Round 3	Agent 1	0.00993	0.01959	0.06148	0.103635	0.18585	0.39625	0.73461	0.96146	0.99590
	Agent 2	0.00536	0.02920	0.09247	0.116939	0.13468	0.63650	0.80819	0.85114	0.99811
Round 4	Agent 1	0.00913	0.01599	0.05989	0.154208	0.15731	0.34342	0.82844	0.97022	0.99416
	Agent 2	0.00510	0.03658	0.06751	0.109969	0.12856	0.27479	0.83876	0.92607	0.99426
Round 5	Agent 1	0.01441	0.02760	0.05941	0.085141	0.15731	0.49669	0.80141	0.97527	0.99806
	Agent 2	0.01559	0.03153	0.08782	0.133833	0.12856	0.39727	0.77646	0.94842	0.99714
Round 6	Agent 1	0.00685	0.02958	0.07463	0.113767	0.13813	0.45241	0.78564	0.94234	0.99356
	Agent 2	0.00102	0.01663	0.07560	0.116833	0.14442	0.34181	0.79091	0.96407	0.99379
Round 7	Agent 1	0.00762	0.03201	0.07799	0.11590	0.14601	0.51210	0.92033	0.98500	0.99737
	Agent 2	0.00329	0.03187	0.04577	0.107263	0.26440	0.48192	0.71663	0.95985	0.99544
Round 8	Agent 1	0.01130	0.03288	0.06224	0.132038	0.14324	0.57609	0.85449	0.96193	0.99663
	Agent 2	0.00912	0.03057	0.07870	0.08010	0.16743	0.44854	0.73207	0.94665	0.99688
Round 9	Agent 1	0.01062	0.01756	0.05753	0.155353	0.17604	0.40339	0.80919	0.95455	0.99791
	Agent 2	0.00691	0.02911	0.07673	0.088987	0.15096	0.44703	0.78741	0.96779	0.99779
Round 10	Agent 1	0.00664	0.02453	0.07230	0.089583	0.17250	0.59105	0.85980	0.96420	0.99694
	Agent 2	0.00498	0.02797	0.06583	0.125769	0.12123	0.43414	0.84841	0.96066	0.99802
	Mean	0.00719	0.02566	0.06727	0.10826	0.15981	0.45090	0.79759	0.95278	0.99649

Figure 12: Area ratios under different weights

Initial Variability: As the weight of individual rewards nears 1, both agents demonstrate little to no inclination towards cooperation, evidenced by a ratio close to 0. (See Figure 12)

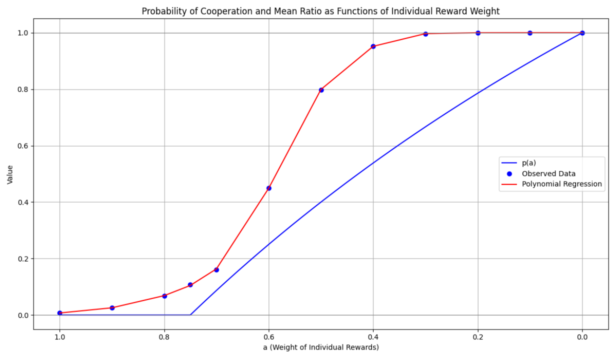


Figure 13: Comparison $p(a)$ and observed data

The emergence of Cooperative Strategy: Notably, at an individual reward weight of $a = 0.7$, the mean ratio begins to surpass 0.15 for both agents, suggesting a tangible change in their cooperative strategy. This growth is substantial enough to notice and can be considered a deviation from a predominantly defection-based strategy. This is not to say the agent has become predominantly cooperative, but it is certainly beginning to explore the cooperative strategy more frequently. From Figure 13, we can observe that the trend indicates a preference shift towards cooperation between the individual reward weightings of 0.8 and 0.7. This empirical observation is in close proximity to the theoretical value of 0.75 that was previously deduced.

Approaching Cooperative Equilibrium: By the time the weight is reduced to $a = 0.3$, both agents are overwhelmingly in favor of cooperation, with the mean ratio nearing 1. This indicates that reducing the value of a beyond this point may not offer any significant behavioral change. This is why we are not conducting experiments with individual reward weightings less than 0.3.

4.5.4 Summery:

While the empirical data and the theoretical $p(a)$ curve present certain similarities, they aren't always in close alignment. A significant deviation is observed especially in the range $a = 0.2$ to $a = 0.6$. Within this range, agents trained with the Q-learning algorithm manifest a proclivity towards cooperation that surpasses the corresponding theoretical values. This unexpected observation could arise due to:

- *Exploration vs. Exploitation Dilemma:* The inherent nature of Q-learning is to balance the exploration of uncharted actions with the exploitation of known rewards. This balance might lead to agents leaning more towards cooperation in the early stages of learning, even if it might not be theoretically optimal [23].
- *Model Simplicity:* The theoretical model for the $p(a)$ curve might be overly simplified. In real-world scenarios, many other factors, including agent-to-agent variability and stochasticity of the environment, can influence the decision-making process [25].
- *Learning Rate and Discount Factor:* The hyper-parameters used in the Q-learning algorithm, such as the learning rate and discount factor, might cause the agent to value immediate or future rewards differently, affecting the cooperation rate [26].

Despite these discrepancies, the empirical data offers invaluable insight. Namely, it accurately pinpoints the threshold $a = 0.75$, at which agents begin to pivot towards a cooperative strategy, consistent with the theoretical prediction.

5 Discussion

The central inquiry of this research revolves around understanding the dynamics of decision-making within multi-agent environments, particularly focusing on the Iterated Prisoner's Dilemma (IPD). Through the lens of the IPD, the study seeks to explore the behavior of agents under varying weights of individual and collective rewards, investigating how agents converge towards cooperation or defection.

The methodology employed in this research combines principles of game theory and reinforcement learning, specifically employing the Q-learning algorithm. The IPD was simulated through multiple interactions, allowing agents to evolve strategies over time. A reward matrix was defined to represent scenarios of mutual cooperation, mutual defection, and mixed strategies. Predictions were made for different scenarios based on the prioritization of individual or team rewards, providing a theoretical foundation for the experiment.

5.1 Analysis of results

Our research experiment consisted of two main parts. The first part is to study the iterated Prisoner's Dilemma without considering team rewards. The second part is to study how the behavioral decisions of the agents are influenced by the weight of team rewards, and to try to identify the turning point at which the intelligences develop a tendency to cooperate when they have not yet formed a strategy.

5.1.1 Interpretation of Findings: Individual Reward Weight at 100%

In the context of our study, where the individual reward weight is set at 100%, the dynamics of agent behavior exhibit a fascinating trajectory. During the initial phases of pre-training, agents are inclined to engage in defection, mirroring each other's betrayal in pursuit of maximizing individual gains. This tendency aligns with Kies's research on the exploitability of strategies within the IPD [1]. Kies's work particularly emphasizes the challenge of constructing the best response in situations characterized by imperfect public monitoring and observation errors. Imperfect public monitoring refers to the scenario where agents can observe each other's actions but with potential inaccuracies or noise. In the context of our study, the focus on individual rewards introduces a parallel challenge. When agents prioritize individual rewards exclusively, they are inclined to pursue strategies that maximize their immediate gains.

However, it is crucial to note that this propensity for defection is not static. As the number of training episodes accumulates, a transformative shift occurs in the agents' strategic landscape. The agents gradually transition from an environment dominated by defection to one where stable cooperative strategies are adopted. This evolution resonates with the concept of 'generous strategies,' where agents begin to forgive past defections in favor of long-term collaborative benefits [6].

The trajectory from initial defection to eventual cooperation underscores the dynamic nature of decision-making in IPD settings. It also highlights the adaptability of agents even when individual rewards are the primary focus, revealing the inherent complexities and opportunities for the emergence of cooperation in such constrained environments.

5.1.2 A Comprehensive Exploration for Different Weights

The comprehensive exploration of different weightings between individual and collective incentives in our study sheds light on the intricate dynamics that govern agent behavior within the IPD. By varying the balance between self-interest and group welfare, we have unveiled a complex landscape where agents continuously adapt and respond to one another.

This exploration has a resonance with Gupta's obedience-based learning mechanism, where agents receive additional rewards for obeying specific commands or following certain guidelines [2]. Obedience as a Coordination Mechanism: Gupta's obedience-based learning introduces an additional layer of coordination by providing incentives for agents to follow certain behaviors. In our study, the varying weights between individual and collective rewards serve a similar purpose. By adjusting these weights, we effectively guide the agents' behavior, creating a mechanism to foster cooperation or defection as desired.

Moreover, the exploration of varying weights revealed a critical turning point around $a = 0.75$, aligning with the predicted value.

5.2 Strengths and limitations

5.2.1 Strengths:

Innovative Visualization Method: One of the key strengths of this paper lies in the utilization of a relatively new visualization method, known as Action Preference. By calculating the agent's action preference directly through the Q-table, the study offers a novel and insightful perspective on the agent's strategic choices. This approach enhances the interpretability of the results and provides a more nuanced understanding of the underlying dynamics.

Theoretical Validation through Experiments: The paper takes a significant step in calculating the theoretical value of the Nash equilibrium of the IPD model, considering the case of team rewards. The subsequent verification of this equilibrium through experimental analysis is noteworthy. The convergence of the actual equilibrium point to the predicted value adds credence to the theoretical framework and showcases the synergy between theoretical constructs and empirical validation.

5.2.2 Limitations:

Algorithmic Constraints: While the application of Q-learning in studying the IPD is an intriguing approach, it may not necessarily be the optimal algorithm for this model. The inherent uncertainty within the IPD leads to a decrease in the accuracy of the results obtained. This limitation highlights the complexity of the model and suggests that alternative algorithms or enhancements to Q-learning might provide more precise insights.

Inconsistencies in Findings: A notable limitation emerges in certain experimental outcomes. Specifically, in the case of personal reward weights of 100%, ten rounds of experiments did not lead to the anticipated convergence to mutual betrayal. This finding is against the results of experiments with personal reward weights of 80% and 90%. The absence of a reasonable explanation for this phenomenon raises questions and underscores the need for further investigation.

Uniform Weights for Agents: A critical limitation of our research is the focus on two agents with the same weights for individual rewards and collective rewards. This approach, while simplifying the experimental design, may not fully capture the complexities of real-world scenarios where weights are often different for different individuals. By not exploring variations in individual reward weights, the study potentially overlooks diverse behavioral patterns and strategic interactions that might emerge in more heterogeneous settings. Conducting experiments with differing weights for different agents could provide a richer and more realistic understanding of multi-agent dynamics.

5.3 Connection to Broader Context

5.3.1 Historical Context of Cooperation in Game Theory

Our research findings, particularly the transition from initial defection to eventual cooperation, offer a nuanced contribution to the longstanding discourse in game theory on cooperation mechanisms. The adaptive behavior observed in our agents when individual rewards are the primary focus echoes the concept of 'generous strategies,' augmenting the ongoing conversation initiated by studies like those of Stewart and Plotkin [6].

5.3.2 AI Ethics and the Balance of Individual and Collective Welfare

Our exploration of different weightings between individual and collective incentives directly contributes to the ethical debate surrounding AI decision-making. The results show that it is possible to guide the behavior of AI agents towards either cooperation or defection by adjusting these weights. This has important ethical implications, as AI systems increasingly function in social and collective settings.

5.3.3 Societal Challenges Requiring Cooperation

Our study indirectly informs global challenges such as climate change and health pandemics, where cooperation among various entities is paramount. While our agents operate in a constrained, theoretical environment, the principles underlying their ability to transition from defection to cooperation could be applied to broader, more complex systems, including international diplomacy and multi-stakeholder negotiations.

5.3.4 Interdisciplinary Relevance

Our work intersects with various academic disciplines, including psychology, economics, and computer science. The concept of obedience-based learning mechanisms, similar to what was proposed by Gupta [2], shows the potential for interdisciplinary applications, especially in the study of human-agent interaction and behavioral economics.

5.3.5 Future Implications in Multi-Agent Systems

Our findings offer vital insights into the behavior of agents within multi-agent systems. The adaptability observed in our agents, especially their ability to shift from a defection-dominated environment to cooperative strategies, could inform the design of future systems where effective agent-agent interaction is crucial, be it in smart cities, healthcare, or automated transport systems.

6 Further Work

The exploration of multi-agent learning strategies in the Iterated Prisoner's Dilemma, considering the intricate balance between individual and team rewards, has opened several avenues for future research:

1. **Alternative Learning Algorithms:** While Q-learning offers a compelling starting point for studying agent interactions, introducing other reinforcement learning algorithms like Deep Q Networks (DQN), Proximal Policy Optimization (PPO), or Actor-Critic methods might offer more granular insights and potentially address some of the observed inconsistencies.

2. **Complex Environments:** Introducing more complex environments, possibly with more states or additional external factors, might provide a more holistic understanding of agent behavior. Situations where the environment itself evolves or responds to agent actions would add a layer of dynamism to the study.

3. **Heterogeneous Agents:** The current study focused on agents with identical weights for rewards. Future work could consider agents with varying preferences or biases, simulating more realistic scenarios where individuals prioritize different objectives.

4. **Temporal Dynamics:** Investigating how agents adapt over longer periods and how their strategies evolve with prolonged interaction might provide insights into the stability of learned behaviors.

5. **Incorporating Memory:** Introducing memory-based strategies, where agents can recall more than just the immediate past action of their counterparts, could lead to more sophisticated interactions and strategies.

6. **Impact of Communication:** Allowing agents to communicate, either explicitly or implicitly, might change the dynamics of the game. Exploring how information exchange influences cooperative and competitive behaviors would be a novel direction.

7. **Fine-tuning of Hyperparameters:** A deeper dive into the effects of varying hyperparameters like learning rate, exploration-exploitation balance, and discount factors in the Q-learning algorithm might offer refined control over agent behaviors.

8. **Interdisciplinary Approaches:** Integrating insights from behavioral economics, sociology, or even biology might offer a more interdisciplinary perspective on cooperation and competition within the framework of the Iterated Prisoner's Dilemma.

By delving into these areas, including the ethical and commercial considerations, we can further our understanding of multi-agent interactions and their broader societal and economic implications.

References

[1] Kies, M. (2020). Finding best answers for the iterated prisoner's dilemma using improved q-learning. *Available at SSRN*, 3556714.

[2] Gupta, G. (2020). Obedience-based multi-agent cooperation for sequential social dilemmas.

[3] Park, H., & Kim, K. J. (2016). Active player modeling in the iterated prisoner's dilemma. *Computational Intelligence and Neuroscience*, 2016.

[4] Capraro, V. (2013). A model of human cooperation in social dilemmas. *PloS one*, 8(8).

[5] Lin, B., Bouneffouf, D., & Cecchi, G. (2020). Online Learning in Iterated Prisoner's Dilemma to Mimic Human Behavior.

[6] Stewart, A. J., & Plotkin, J. B. (2012). A review of iterated prisoner's dilemma strategies. *Royal Society of London*.

[7] Press, W. H., & Dyson, F. J. (2012). From extortion to generosity, evolution in the Iterated Prisoner's Dilemma. *Proceedings of the National Academy of Sciences*.

[8] Hilbe, C., Traulsen, A., & Sigmund, K. (2015). Partners or rivals? Strategies for the iterated prisoner's dilemma. *Games and Economic Behavior*, 92, 41-52.

[9] Vassiliades, V., & Christodoulou, C. (2010). Multiagent Reinforcement Learning in the Iterated Prisoner's Dilemma: Fast cooperation through evolved payoffs. In *The 2010 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-8).

[10] Dresher, M. (1961). The mathematics of games of strategy: Theory and applications. *Dover Publications*.

[11] Rapoport, A., & Chammah, A. M. (1965). Prisoner's Dilemma: A study in conflict and cooperation. *University of Michigan press*.

[12] Axelrod, R. (1984). The Evolution of Cooperation.

[13] Nowak, M., & Sigmund, K. (1993). A strategy of win-stay, lose-shift that outperforms tit-for-tat in the Prisoner's Dilemma game. *Nature*, 364(6432), 56-58.

[14] Friedman, J. W. (1971). A Non-cooperative Equilibrium for Supergames. *Review of Economic Studies*, 38(1), 1-12.

[15] Littman, M. L., & Stone, P. (2001). Leading best-response strategies in repeated games. *IJCAI*.

[16] Fogel, D. B. (2006). Evolutionary Computation: Toward a New Philosophy of Machine Intelligence.

[17] Sandholm, T. W., & Crites, R. H. (1996). Multiagent Reinforcement Learning in the Iterated Prisoner's Dilemma.

[18] Lindgren, K., & Nordahl, M. G. (1994). Evolutionary dynamics of spatial games. *Physica D: Nonlinear Phenomena*, 75(1-3), 292-309.

[19] Axelrod, R., & Hamilton, W. D. (1981). The evolution of cooperation. *Science*, 211(4489), 1390-1396.

[20] Nowak, M. A. (2006). Five rules for the evolution of cooperation. *Science*, 314(5805), 1560-1563.

[21] Trivers, R. L. (1971). The evolution of reciprocal altruism. *The Quarterly Review of Biology*, 46(1), 35-57.

[22] Rand, D. G., & Nowak, M. A. (2013). Human cooperation. *Trends in Cognitive Sciences*, 17(8), 413-425.

[23] Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction. *MIT press*.

[24] Watkins, C. J. C. H., & Dayan, P. (1992). Q-learning. *Machine Learning*, 8(3-4), 279-292.

[25] Nowak, M. A. (2006). Evolutionary Dynamics: Exploring the Equations of Life. *Belknap Press*.

[26] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing Atari with Deep Reinforcement Learning. *arXiv preprint arXiv:1312.5602*.

[27] Nash, J. (1950). Equilibrium Points in N-Person Games. *Proceedings of the National Academy of Sciences*, 36(1), 48-49.

7 Appendix

A Tables

Table 1: Behavioral ratios for both agents over 10 experimental rounds

	a value	**1**	**0.9**	**0.8**	**0.75**	**0.7**	**0.6**	**0.5**	**0.4**	**0.3**
Round 1	**Agent 1**	0.00439	0.02746	0.04800	0.07881	0.15883	0.44766	0.76774	0.91011	0.99744
	Agent 2	0.01376	0.02772	0.06672	0.09993	0.16573	0.62789	0.75004	0.97588	0.99639
Round 2	**Agent 1**	0.00841	0.02334	0.06142	0.09301	0.15655	0.47892	0.76675	0.95896	0.99823
	Agent 2	0.01095	0.01668	0.07376	0.10466	0.27273	0.41828	0.80369	0.98043	0.99577
Round 3	**Agent 1**	0.00993	0.01959	0.06148	0.10363	0.18585	0.39625	0.73461	0.96146	0.99590
	Agent 2	0.00536	0.02920	0.09247	0.11693	0.13468	0.63650	0.80819	0.85114	0.99811
Round 4	**Agent 1**	0.00913	0.01599	0.05989	0.15420	0.15731	0.34342	0.82844	0.97022	0.99416
	Agent 2	0.00510	0.03658	0.06751	0.10996	0.12856	0.27479	0.83876	0.92607	0.99426
Round 5	**Agent 1**	0.01441	0.02760	0.05941	0.08514	0.15731	0.49669	0.80141	0.97527	0.99806
	Agent 2	0.01559	0.03153	0.08782	0.13383	0.12856	0.39727	0.77646	0.94842	0.99714
Round 6	**Agent 1**	0.00685	0.02958	0.07463	0.11376	0.13813	0.45241	0.78564	0.94234	0.99356
	Agent 2	0.00102	0.01663	0.07560	0.11683	0.14442	0.34181	0.79091	0.96407	0.99379
Round 7	**Agent 1**	0.00762	0.03201	0.07799	0.11590	0.14601	0.51210	0.92033	0.98500	0.99737
	Agent 2	0.00329	0.03187	0.04577	0.10726	0.26440	0.48192	0.71663	0.95985	0.99544
Round 8	**Agent 1**	0.01130	0.03288	0.06224	0.13203	0.14324	0.57609	0.85449	0.96193	0.99663
	Agent 2	0.00912	0.03057	0.07870	0.08010	0.16743	0.44854	0.73207	0.94665	0.99688
Round 9	**Agent 1**	0.01062	0.01756	0.05753	0.15535	0.17604	0.40339	0.80919	0.95455	0.99791
	Agent 2	0.00691	0.02911	0.07673	0.08898	0.15096	0.44703	0.78741	0.96779	0.99779
Round 10	**Agent 1**	0.00664	0.02453	0.07230	0.08958	0.17250	0.59105	0.85980	0.96420	0.99694
	Agent 2	0.00498	0.02797	0.06583	0.12576	0.12123	0.43414	0.84841	0.96066	0.99802
Mean		0.00719	0.02566	0.06727	0.10826	0.15981	0.45090	0.79759	0.95278	0.99649

B Images

Please refer to the images illustrating scenarios where individual rewards are weighted at:

- 100 percent: GitHub link
- 90 percent: GitHub link
- 80 percent: GitHub link
- 70 percent: GitHub link
- 60 percent: GitHub link
- 50 percent: GitHub link
- 40 percent: GitHub link
- 30 percent: GitHub link
- 20 percent: GitHub link
- 10 percent: GitHub link
- 0 percent: GitHub link

C Sample Code

```
# Install packages
!pip install pettingzoo
!pip install ray
!pip install -U tensorboardx
!pip install lz4
!pip install --upgrade protobuf
!pip install --upgrade tensorflow
!pip install --upgrade tensorboard
!pip install dm_tree
```

Listing 1: Create environment

```
## Create environment
from gymnasium.spaces import Discrete, Dict
from pettingzoo import ParallelEnv
import numpy as np
import functools

class IPDPettingZooEnv(ParallelEnv):
    def __init__(self, reward_matrix, n=1, render_mode=None):
        self.reward_matrix = reward_matrix
        self.n = 100  # Number of rounds to play
        self.current_step = 0  # Current game round
        self.render_mode = render_mode

        self.possible_agents = ["agent1", "agent2"]
        self.observation_spaces = {
            agent: Discrete(4)
            for agent in self.possible_agents
        }
        self.action_spaces = {
            agent: Discrete(2)  # Changed here
            for agent in self.possible_agents
        }

        self._agent_dones = {agent: False for agent in self.possible_agents}
        self._last_agent_actions = {agent: None for agent in self.possible_agents}

    def get_episode_length(self):
        return self.n

    @functools.lru_cache(maxsize=None)
    def observation_space(self, agent):
        return self.observation_spaces[agent]

    @functools.lru_cache(maxsize=None)
    def action_space(self, agent):
        return self.action_spaces[agent]

    def render(self):
        if self.render_mode == 'human':
            print(f'Current step: {self.current_step}, Agents: {self.agents}')

    def reset(self, seed=None, options=None):
        self.agents = self.possible_agents[:]
        self.current_step = 0
```

```python
        observations = {agent: np.array([0]*2*self.n, dtype=np.int8) for agent in self.agents}
        infos = {agent: {} for agent in self.agents}
        self._agent_dones = {agent: False for agent in self.agents}
        self._last_agent_actions = {agent: None for agent in self.agents}
        return observations, infos

    def step(self, actions):
        self.current_step += 1
        assert isinstance(actions, dict), "actions must be a dictionary"
        for agent, action in actions.items():
            assert agent in self.agents, f"{agent} is not a valid agent for this environment"
            assert action in [0, 1], f"{action} is not a valid action. The action must be either 0 or 1"
            self._agent_dones[agent] = True
            self._last_agent_actions[agent] = action

        last_actions = [self._last_agent_actions[agent] for agent in self.agents]

        # debugging information
        print(f"Last actions: {last_actions}")
        assert max(last_actions) < len(self.reward_matrix), "Actions exceed reward matrix dimensions."

        rewards = {
            self.agents[0]: self.reward_matrix[int(last_actions[0])][int(last_actions[1])][0],
            self.agents[1]: self.reward_matrix[int(last_actions[0])][int(last_actions[1])][1]
        }

        # Determine the next state based on the last actions
        next_state = 0
        if last_actions[0] == 0 and last_actions[1] == 1:
            next_state = 1
        elif last_actions[0] == 1 and last_actions[1] == 0:
            next_state = 2
        elif last_actions[0] == 1 and last_actions[1] == 1:
            next_state = 3

        observations = {
        agent: next_state
        for agent in self.agents
        }

        terminations = {agent: self.current_step >= self.n for agent in self.agents}

        infos = {agent: {} for agent in self.agents}
        if self.current_step >= self.n:
            self.agents = []

        if self.render_mode == "human":
            self.render()
        return observations, rewards, terminations, infos
```

Listing 2: Define the reward matrix

```
reward_matrix = [
    [[3, 3], [0, 5]],
    [[5, 0], [1, 1]]
]

if __name__ == "__main__":
    env = IPDPettingZooEnv(reward_matrix)
```

Listing 3: Reinforcement Learning

```
class QLearningAgent:
    def __init__(self, env, agent_name, individual_reward_weight=0.75,
       team_reward_weight=0.25, learning_rate=0.1, discount_factor=0.95,
       exploration_rate=1.0, exploration_decay_rate=0.003):
        self.env = env
        self.agent_name = agent_name
        self.individual_reward_weight = individual_reward_weight
        self.team_reward_weight = team_reward_weight
        self.learning_rate = learning_rate
        self.discount_factor = discount_factor
        self.exploration_rate = exploration_rate
        self.exploration_decay_rate = exploration_decay_rate

        # Initialize Q-table to be zeros. Assumes each agent has action and
            observation space of size 2.
        self.q_table = np.zeros((4, 2))

    def get_action(self, state):
        # Exploration vs exploitation
        if np.random.uniform(0, 1) < self.exploration_rate:
            # Exploration
            action = self.env.action_spaces[self.agent_name].sample()
        else:
            # Exploitation
            action = np.argmax(self.q_table[state])
        return action

    def update_q_table(self, state, action, reward, next_state, team_reward):
        # Combine individual and team rewards
        combined_reward = self.individual_reward_weight * reward + self.
            team_reward_weight * team_reward

        # Q-learning update rule
        self.q_table[state][action] = (1 - self.learning_rate) * self.q_table[
            state][action] \
            + self.learning_rate * (combined_reward + self.discount_factor * np.
                max(self.q_table[next_state]))

    def decay_exploration_rate(self):
        self.exploration_rate = self.exploration_rate * np.exp(-self.
            exploration_decay_rate)

# Initialize environment and agents
env = IPDPettingZooEnv(reward_matrix)
observations, _ = env.reset()
agents = [QLearningAgent(env, name) for name in env.agents]

import pandas as pd
# Define the moving average function
```

```
def moving_average(data, window_size):
    return pd.Series(data).rolling(window=window_size).mean().tolist()

# Initialize agent and team reward storage
agent_rewards = {agent.agent_name: [] for agent in agents}
team_rewards = []

# Number of episodes for training
num_episodes = 2000

q_table_history = {agent.agent_name: [] for agent in agents}
for episode in range(num_episodes):
    episode_rewards = {agent.agent_name: 0 for agent in agents}
    episode_team_reward = 0
    observations, _ = env.reset()

    while True:
        actions = {agent.agent_name: agent.get_action(observations[agent.
           agent_name]) for agent in agents}
        next_observations, rewards, done, _ = env.step(actions)
        team_reward = sum(rewards.values())

        episode_team_reward += team_reward
        for agent in agents:
            agent.update_q_table(observations[agent.agent_name], actions[agent.
               agent_name], rewards[agent.agent_name], next_observations[agent.
               agent_name],team_reward)
            episode_rewards[agent.agent_name] += rewards[agent.agent_name]

        if all(done.values()):
            print('/done')
            break

        observations = next_observations

    episode_length = env.get_episode_length()
    team_rewards.append(episode_team_reward/episode_length)
    for agent_name in agent_rewards:
        agent_rewards[agent_name].append(episode_rewards[agent_name]/
           episode_length)


    for agent in agents:
        agent.decay_exploration_rate()
        q_table_history[agent.agent_name].append(agent.q_table.copy())

for agent in agents:
    print(f"Q-table for {agent.agent_name}:")
    print(agent.q_table)
```

Listing 4: Visulization

```
import matplotlib.pyplot as plt
from matplotlib import gridspec

# Compute the moving averages of the rewards
window_size = 50  # Adjust the window size to change the smoothing level
smoothed_agent_rewards = {agent_name: moving_average(rewards, window_size) for
    agent_name, rewards in agent_rewards.items()}
```

```
smoothed_team_rewards = moving_average(team_rewards, window_size)

# Create a new figure with GridSpec layout
fig = plt.figure(figsize=(8, 6))  # Adjust the size as per your requirement
gs = gridspec.GridSpec(2, 2)

# Agent rewards over time
ax0 = fig.add_subplot(gs[0])
for agent_name, rewards in smoothed_agent_rewards.items():
    ax0.plot(rewards, label=f"{agent_name} rewards")
ax0.set_xlabel('Episodes')
ax0.set_ylabel('Average rewards')
ax0.legend()
ax0.set_title('Agent rewards over time')

# Team rewards over time
ax1 = fig.add_subplot(gs[1])
ax1.plot(smoothed_team_rewards, label='Team rewards')
ax1.set_xlabel('Episodes')
ax1.set_ylabel('Average team rewards')
ax1.legend()
ax1.set_title('Team rewards over time')

# Plotting the action preference for agent 1
ax2 = fig.add_subplot(gs[2])
agent_name = list(q_table_history.keys())[0]  # assuming the first agent is agent 1
q_tables = q_table_history[agent_name]

annotations = []
for i in range(4):
    action_diff = [q_table[i][0] - q_table[i][1] for q_table in q_tables]
    ax2.plot(action_diff, label=f"State {i}")
    annotation = ax2.annotate(f"{action_diff[-1]:.2f}", xy=(len(action_diff)-1, action_diff[-1]), textcoords="offset points", xytext=(0,0), ha='center')
    annotations.append(annotation)

adaptive_placement(ax2, annotations)
ax2.axhline(0, color='gray', linewidth=0.8)
ax2.set_title(f"Action preference for {agent_name}")
ax2.set_xlabel('Episodes')
ax2.set_ylabel('Action Preference')
ax2.legend()

# Plotting the action preference for agent 2
ax3 = fig.add_subplot(gs[3])
agent_name = list(q_table_history.keys())[1]  # assuming the second agent is agent 2
q_tables = q_table_history[agent_name]

annotations = []
for i in range(4):
    action_diff = [q_table[i][0] - q_table[i][1] for q_table in q_tables]
    ax3.plot(action_diff, label=f"State {i}")
    annotation = ax3.annotate(f"{action_diff[-1]:.2f}", xy=(len(action_diff)-1, action_diff[-1]), textcoords="offset points", xytext=(0,0), ha='center')
    annotations.append(annotation)

adaptive_placement(ax3, annotations)
```

```
ax3.axhline(0, color='gray', linewidth=0.8)
ax3.set_title(f"Action preference for {agent_name}")
ax3.set_xlabel('Episodes')
ax3.set_ylabel('Action Preference')
ax3.legend()

# Adjust the layout of the plots in the figure
plt.tight_layout()

# Save the figure with high resolution (300 dpi)
plt.savefig('combined_figure.png', dpi=300)
plt.show()
```

Listing 5: Calculation of area ratio

```
# Extracting the action preference values for episodes between 20 and 250
action_preferences = {agent_name: [q_table[i][0] - q_table[i][1] for q_table in
    q_tables[20:250]] for agent_name, q_tables in q_table_history.items()}

# Calculating the mean for values greater than 0 and less than 0 for each agent
area_values = {}
for agent_name, preferences in action_preferences.items():
    positive_values = [value for value in preferences if value > 0]
    negative_values = [value for value in preferences if value < 0]

    positive_area = sum(positive_values)  if positive_values else 0
    negative_area = sum(negative_values)  if negative_values else 0

    area_values[agent_name] = {
        "positive_area": positive_area,
        "negative_area": negative_area,
        "p":positive_area/(positive_area - negative_area)
    }

area_values
```

The source code for our algorithms and experiments can also be found at this GitHub Repository