ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ARTIFICIAL INTELLIGENCE

MASTER THESIS

in

LLM-based Multi-Agent Systems

PROMPT SENSITIVITY TO CONTEXT IN LLM MULTI-AGENT DECISION-MAKING: A PRISONER'S DILEMMA CASE STUDY

CANDIDATE
Gabriele Nanni

SUPERVISOR
Prof. Paolo Torroni

CO-SUPERVISORS
Prof. Roberta Calegari
Prof. Christian Arnold

Academic year 2024-2025
Session 3rd

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The rapid progress of Large Language Models (LLMs) has opened new possibilities for designing artificial agents capable of reasoning, interacting, and making decisions in complex environments. When multiple LLM-based agents are allowed to interact with one another, they form LLM-based Multi-Agent Systems (MAS) in which decision-making emerges from dialogue, negotiation, and strategic reasoning. While these systems show promising capabilities, their behavior remains difficult to predict and evaluate, especially when they operate in interactive and strategic settings.

The main objective of this thesis is to investigate how sensitive LLM-based multi-agent decision-making is to contextual information provided through prompts. In particular, this work explores whether LLM-based agents are able to correctly interpret the strategic structure of the environment in which they operate and adapt their decisions accordingly. Understanding this capability is crucial if such systems are to be used in simulations or decision-support tools that rely on realistic strategic reasoning.

To study this problem in a controlled way, the analysis focuses on the Prisoner's Dilemma, one of the most widely studied models in game theory. The Prisoner's Dilemma provides a simple but powerful framework for analyzing cooperation, competition, and strategic reasoning between agents. By embedding LLM-based agents in an iterated version of this game, it becomes possible

to observe how their behavior evolves across repeated interactions and how it is influenced by contextual variations introduced through prompting.

This thesis was developed in collaboration with the Centre for Artificial Intelligence in Government (CAIG) at the University of Birmingham. The centre conducts research on high-impact applications of artificial intelligence within governmental and public policy contexts. One of its long-term research goals is the development of AI-based systems capable of simulating political debate and policy-making processes within institutional environments such as multilateral organizations. In these contexts, political actors engage in strategic interactions involving negotiation, persuasion, coalition formation, and signaling under conditions of incomplete information and heterogeneous interests. Due to the strategic nature of these problems game theory is often used to model these environments.

Modeling such environments presents significant challenges. Traditional game-theoretic models provide strong analytical foundations but often rely on simplified assumptions about communication and strategic behavior. Conversely, agent-based simulations allow researchers to represent heterogeneous actors and study emergent dynamics, but they typically depend on manually defined behavioral rules and simplified communication protocols. LLM-based agents introduce a new paradigm that potentially bridges these approaches: they are capable of generating natural language arguments, adapting their reasoning to context, and engaging in complex dialogue that resembles human interaction.

Despite these promising capabilities, the behavior of LLM-based agents in strategic settings is still poorly understood. In particular, it is important to determine whether these agents can correctly interpret incentives, adapt their strategies to different contexts, and maintain coherent reasoning across repeated interactions. For this reason, this thesis investigates how contextual framing influences the behavior of LLM-based multi-agent systems operating in a game-theoretic environment.

To address this question, a modular experimental framework was developed in which LLM-based agents can interact within different game theoretic scenarios. The one implemented and analyzed in this work will be the repeated Prisoner's Dilemma case, tested under different contextual conditions. The study analyzes both behavioral patterns (such as cooperation tendencies and retaliation dynamics) and the agents' understanding of the game environment, evaluated through additional contextual questions asked during gameplay.

Thesis structure

The remainder of this thesis is structured as follows. Chapter 2 introduces the theoretical background, including fundamental concepts of game theory, the Prisoner's Dilemma, and the evolution of large language models and multi-agent systems. Chapter 3 reviews related work on human behavior in the Prisoner's Dilemma, LLM-based agents, and multi-agent reasoning systems to give context and to describe the current state of research on these topics. Chapter 4 describes the experimental framework, including the system architecture, prompt design, and experimental setup used to evaluate the agents. Chapter 5 presents and discusses the experimental results, focusing on behavioral metrics and differences across contexts and models. Finally, Chapter 6 concludes the thesis by summarizing the main findings and outlining possible directions for future research.

Chapter 2

Background

2.1 Game Theory and Prisoner's Dilemma

2.1.1 Introduction to Game Theory

Game theory is a branch of mathematics and economics that studies strategic interactions among rational decision-makers, where the outcome for each participant depends not only on their own actions but also on the actions chosen by others. It provides formal models, called games, to analyze situations involving conflict, cooperation, or competition, and it focuses on concepts such as strategies, payoffs, information, and equilibrium. This discipline originated from the study of gambling games and their outcomes, which led to a deep and systematic analysis of mathematical representations of strategic situations.

Before continuing, it is necessary to introduce some basic concepts in order to ensure a clear and unambiguous understanding of game theory [4, 15].

- A *game* is a formal setting defined by a set of rules that specify the possible actions, the order of moves, and the outcomes. Each specific realization of the game, from its beginning to its end, is called a *play*.

- A *player* is any decision-making entity participating in the game. Players are assumed to act rationally, choosing actions that maximize their own objectives based on their beliefs about the behavior of other players.
- A *move* is an instance in which a choice among different alternatives is made. A move can be made by a player or by another mechanism specified by the rules of the game, such as chance.
- A *strategy* is a complete contingent plan that specifies the action a player will take in every possible situation in which the player may be required to act, even if some of these situations never occur during a play.
- A *payoff* is a value assigned to each player for every possible outcome of the game. It represents the player's preferences, utility, or reward, and it is the criterion used to evaluate strategic decisions.
- A *complete information game* is a strategic game where all players have full knowledge of the game's structure, including the set of players, the action sets, and the payoff functions of all players. Specifically, each player knows the total number of players involved in the game, the set of actions available to each player and the payoff function for each player.
- A *simultaneous game* is a type of game where all players make their decisions or choose their actions at the same time, without knowledge of the choices made by the other players. In such games, players act independently and cannot coordinate their strategies based on others' actions.

After defining these basic concepts, it is important to clarify that, in the standard game-theoretic framework, a player is assumed to be rational and egoistic. This means that players have consistent preferences and are able to make decisions aimed at maximizing their expected payoffs, while forming beliefs about the actions of other players.

A strategy is called a *dominant strategy* if it provides a player with an outcome that is at least as good as any other strategy, regardless of the strategies chosen by the opponents. A *Nash Equilibrium* (NE) is a strategy profile in which no player has an incentive to unilaterally deviate from their chosen strategy, given the strategies of the other players. In such an equilibrium, each player's strategy is a best response to the strategies of the others. If a Nash Equilibrium is unique, then all rational and egoistic players are expected to select the strategy corresponding to that equilibrium. Lastly, an action profile a is *Pareto optimal* if it is impossible to make any agent better off without making at least one other agent worse off. This concept focuses on the efficiency of the collective action choices, ensuring that no improvement in one agent's payoff can be achieved without a detriment to another agent's payoff, as represented in the payoff matrix.

The main difference between NE and a Pareto Optimal action profile is that a NE is not necessarily the best possible outcome, it is the outcome where all players are satisfied independently of what the other players choose, while the Pareto Optimal action profile is the best possible collective outcome.

Traditional game theory focuses its studies on the homo economicus model, considering the completely rational agent described before, but empirical studies from different sources have shown how the human behavior differs considerably from the theoretical and rational path. This caused a new branch to develop, called Behavioral Game Theory, where the subject is no longer the homo ecomonicus, but it is replaced by the homo sapiens, trying to study the behavior of people in the same situations, considering social conditioning, emotions and personal bias.

These new determining factors required the models to be adjusted, but now offer an interesting insight about the human behavior in these scenarios. With the development of AI applications in multiple fields these studies are extremely relevant to be able to compare the new network-based models and their behavioral patterns with human one.

2.1.2 Prisoner's Dilemma

The Prisoner's Dilemma is one of the best known examples in game theory illustrating why two rational individuals might not cooperate, even when cooperation appears to be in their best interest.
The name derived by the vignette that exemplifies the rules of the game:

> *Two criminals are arrested for a crime and imprisoned. Each prisoner is in solitary confinement with no means to communicate with the other.*
> *The police has not enough evidence to convict the pair on the principal charge. They plan to sentence both for B years for a minor charge.*
> *They offer both the prisoners a Faustian bargain. If they testify against the other, they will go free and the other will get D years in prison on the main charge.*
> *But there is a catch, if they both testify they will be charged with C years in prison.*
> *Each is informed that the other prisoner is being offered the very same deal.*
> *Each prisoner is concerned only with minimizing his own prison sentence.*

This is a simultaneous two-player game where the possible actions for both the players are 'Cooperate' and 'Defect'. It can be modeled by assigning non-negative payoffs to each player based on the outcome. Mutual cooperation, with both players choosing to cooperate, results in the highest collective payoff, with both players receiving a *reward* R (equivalent of B years in prison). If only one of them confesses and the other remains silent, the defector will obtain the highest possible individual *temptation* payoff T (indicating them being released), while the one who remained silent will be awarded the lowest possible *sucker's* payoff S (the maximum sentence D). If both confess (both choose 'Defect') they will receive a *penalty* payoff P (C years in prison).
The traditional game's hierarchy is $T > R > P > S$. In this discussion the payoff hierarchy will be modified to study different settings and their influence in the behavior of the players.
An example of the payoff matrix is shown in Table 2.1.
Due to the fact that defecting is a dominant strategy both players we can easily

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

Table 2.1: Visual representation of the PD payoff matrix

observe that the Nash Equilibrium is the outcome ('Defect', 'Defect'), assigning a *penalty* payoff P to both. None of them has the interest in changing due to the uncertainty about the choice the other will make. It is also possible to observe that the NE does not equates the Pareto Optimal choice, since it is clear that the maximum collective payoff would be $R + R$, obtained only with mutual collaboration.

2.1.3 Iterated Prisoner's Dilemma

As described previously, the Prisoner's Dilemma is a pretty simple and straightforward game. A really natural evolution of the PD is when the game is not played one single time, but instead repeated over multiple rounds between the two players, with each player trying to maximize the reward across all the games played. This is called the *Iterated Prisoner's Dilemma* (IPD).
As stated before a *strategy* is the complete contingent plan that specifies the action of the player in any possible situation. In this newly defined game the strategy can no longer be limited to the single action to be chose in the current game. Instead it has to consider all the possible situations determined by the outcomes of the past rounds. In this context, past interactions influence future decisions, and a strategy can be interpreted as an algorithm that determines a player's actions based on the history of the game.
Although many different strategies can be defined for the IPD, there is no universally optimal strategy, even if some are more robust than others, meaning that they perform better with various opponents. In literature [7] some of the most common strategies are:

- ***Always Cooperate***: The player always chooses to cooperate, regardless

of past outcomes.

- ***Always Defect***: The player always chooses to defect, regardless of past outcomes.
- ***Random***: The player chooses with equal probability between cooperation and defection. There are variations of this strategy that modify the probability of selection.
- ***Tit For Tat***: The player repeats the action played by the other player the round before.
- ***Grim Trigger***: The player chooses to cooperate until the opponent defects once. After the first defection from the opponent the player will always defect.

Among these strategies, those most commonly observed in human are ***Tit For Tat***, ***Always Defect*** and ***Grim Trigger***.
In this project the main deterministic strategies adopted are ***Always Cooperate***, ***Always Defect*** and ***Tit For Tat***.

Since behavioral economics wants to study and analyze human behavior in specific settings these games are usually observed from a psychological and anthropological point of view. For this reason usually these studies employ different metrics to observe the behavior of populations or individuals during these games.
The most used metric in the Prisoner's Dilemma problem is the cooperation rate, which measures the number of cooperations out of the total number of rounds, but other metrics can be defined in each study.

2.2 LLMs and LLM-based MAS

2.2.1 Historical evolution of NLP

Natural Language Processing (NLP) is a subfield of Artificial Intelligence and, more broadly, Computer Science, which aims to automate the analysis, generation, and understanding of human language. It enables machines to read and interpret text, translate it across different languages, extract meaning or sentiment, answer questions, and perform many other tasks related to language.
Because it deals with both language and computation, NLP lies at the intersection of computer science and linguistics. It provides a set of methods and tools designed to make human language accessible to computers. Natural Language Processing is widely used in modern society, with applications such as automatic web page translation, spam detection, and the high level of text understanding achieved by modern search engines.
The first approaches to NLP [22] date back to the mid-twentieth century. During this period, methods were mainly rule-based, relying on manually defined grammatical and linguistic rules. However, these approaches proved to be brittle and not scalable. In the 1970s, more advanced techniques were introduced by incorporating symbolic knowledge, such as ontologies and semantic representations, into NLP systems. Although these methods achieved better results, they were still effective mainly in limited or simplified domains and did not generalize well to real-world applications. A major advancement in NLP occurred in the 1980s, driven by the increasing availability of digital text and improvements in computational resources. This led to a shift toward probabilistic and statistical approaches, in which language was modeled as a stochastic process. One of the most influential techniques of this period was the n-gram language model, which estimates the probability of a word given the preceding $n - 1$ words using Bayesian principles. This marked a clear separation from purely linguistic theories toward data-driven statistical analysis. As a result, tasks such as machine translation, part-of-speech tagging, and

speech recognition became more feasible, as language representations were learned from data rather than manually encoded.

The next stage in the evolution of NLP was the adoption of machine learning methods. These approaches still heavily relied on human-designed features, and due to limited data and computational power, the resulting models were often task-specific and struggled with generalization. Furthermore, modeling semantics and reasoning remained challenging, and many architectures lacked the ability to capture long-range dependencies in text.

With the advent of deep learning, neural networks largely replaced manual feature engineering. Distributed word representations, such as Word2Vec and GloVe, became fundamental components of NLP systems. The introduction of transformer architectures further addressed issues related to long-range context and model specialization. Models such as BERT and the GPT family emerged as general-purpose language models, pre-trained on large text corpora and capable of being fine-tuned for a wide range of downstream tasks.

2.2.2 Large Language Models

Large Language Models (LLMs) are systems trained on very large text corpora to model and generate human language. As previously discussed, the modern approach to language modeling is grounded in probability theory, with the objective of learning a probabilistic distribution over sequences of tokens. Similar to earlier n-gram–based models, LLMs predict words, subwords, or other textual units based on the input and the previously generated elements [5].

Before being processed by the model, the input text is handled by a *tokenizer*, a fundamental component in NLP systems. The tokenizer splits the text into smaller units, called *tokens*, which represent the basic elements on which the model operates. Tokens may correspond to whole words, parts of words, individual characters, punctuation marks, or special symbols such as the beginning

or end of a sentence. The tokenizer then maps each token to a numerical identifier and may also apply normalization procedures.

The transformer architecture [24] commonly used in LLMs follows an encoder–decoder structure, composed of two main components. The *encoder* is responsible for transforming the input tokens into numerical, multi-dimensional representations that capture their meaning. These representations, often referred to as *latent vectors*, encode both semantic and syntactic information. A key element of the encoder is the attention mechanism.

Introduced in the 2017 paper Attention Is All You Need [24], the attention mechanism allows the model to evaluate the relevance of each token with respect to all others in the sequence. Without going into technical details, this mechanism enables a more complete representation of the input by explicitly modeling dependencies between tokens. As a result, it effectively addresses the long-range dependency problem, since tokens are processed in parallel rather than sequentially.

The *decoder* component, which also relies heavily on attention mechanisms, performs the complementary task of the encoder. Its purpose is to generate the next token by computing a latent representation based on the previously generated tokens and the encoded input, and then transforming this representation into a probability distribution over the vocabulary.

During the training phase, the model is exposed to extremely large and diverse text corpora. Through this process, the model learns the statistical distribution of tokens by comparing its predictions with the ground-truth data and iteratively updating its parameters to minimize prediction error.

After training, the model can be used to generate text. At each step, it estimates the probability of the next token given the input and the previously generated sequence. To control the level of randomness in the generated output and to better approximate the variability of human language, a parameter called *temperature* is used. Higher temperature values lead to more diverse and less predictable outputs, while lower values result in more deterministic

text generation.

2.2.3 Agents and Multi-Agents Systems

In the early stages of computing, computer systems were monolithic and isolated entities [26], operated directly by humans and relying on them for communication and coordination. In contrast, modern systems are typically large, interconnected, and distributed networks capable of autonomous interaction and communication. The most prominent example of this evolution is the Internet, which is pervasive in both commercial and private contexts. Distributed systems have become the standard in contemporary industrial and commercial environments, motivating researchers to study, model, and better understand this paradigm and its interaction with emerging technologies.
The complexity of the tasks assigned to these systems is continuously increasing, and control is progressively being delegated to automated components. It is now common to design complex networks of autonomous agents that are able to communicate and coordinate with one another, forming large-scale agent networks. Such networks are known as *Multi-Agent Systems* (MAS), reflecting the presence of multiple interacting actors within the system.
Due to their widespread adoption, these systems are often required to interact with human users, understand the context in which they operate, and act according to their objectives within dynamic environments.Also, due to the networking nature of MAS, their components are also geographically distributed. To meet these requirements, agents must exhibit a certain degree of autonomy and intelligence, although the level of these properties may vary depending on the application.
One of the most common applications of Multi-Agent Systems is simulation. MAS-based simulations are particularly useful for modeling complex scenarios by representing the environment and the involved entities as agents. By running such simulations, and assuming that goals and interaction rules are

properly defined, it is possible to observe and analyze different system behaviors and possible outcomes.

2.2.4 LLM-based Agents

Today, LLM-based agents are used in a wide range of applications, including human–computer interaction, virtual assistants, technical support systems, document analysis, and automated decision-making.
Furthermore, LLM-based agents are increasingly employed to construct Multi-Agent Systems that simulate human interactions, enabling the study of emergent behaviors arising from the interactions among multiple language-based agents.

Historically, intelligent agents were mainly deterministic and based on symbolic reasoning approaches. Their behavior was defined by explicit rules, and their reasoning process followed predefined logical structures. However, with the rapid diffusion of Large Language Models (LLMs) and the significant improvement in their capabilities, researchers have started to explore new paradigms where agents are built directly on top of LLMs.

These new agents rely on LLMs as their core reasoning component, but the overall infrastructure is more complex than a simple prompt–response system. For example, many architectures include state-based memory, which allows the agent to keep track of past interactions and better understand the environment in which it operates. This is particularly useful in simulations or dynamic contexts, where the agent must adapt its behavior according to the current situation. Another important feature is the tool interface, which enables the LLM to call external tools that can either act on the environment or retrieve additional information to support its reasoning process.

Tool calling can include code execution, database queries, or interactions with web APIs. By integrating these capabilities, the action space of the agent

is significantly expanded. Instead of only generating text, the model can perform concrete operations and use their outputs to guide further reasoning steps.

The possibility to specialize in different tool calls and to iteratively refine actions allows the creation of highly flexible systems. A single agent, if equipped with the appropriate tools, can solve multiple types of problems and improve its solution through repeated reasoning cycles. Alternatively, developers can design multi-agent systems composed of several specialized agents. In such systems, some agents may focus on executing specific tasks, while others operate at a higher level, planning, coordinating, or verifying the overall workflow and its results.

Nowadays, LLM-based agents are widely adopted, and several frameworks support the development of these architectures. Examples include LangGraph, which focuses on graph-based orchestration with human-in-the-loop control; CrewAI, which follows a role-based team paradigm where agents collaborate as members of a crew; and AG2, a conversation-centric framework in which agents mainly interact through structured text exchanges with flexible roles.

Today, LLM-based agents are applied in a wide range of domains, including human-computer interaction, virtual assistants, technical support systems, document analysis, and automated decision-making. Moreover, they are increasingly used to build multi-agent systems that simulate human interactions, making it possible to study emergent behaviors arising from the communication and coordination of multiple language-based agents.

Chapter 3

Related Work

This chapter reviews prior work at the intersection of human decision-making in the Prisoner's Dilemma, the development of LLM-based agents, and multi-agent systems (MAS) designed for reasoning and interaction. First, research on the Prisoner's Dilemma in humans is surveyed to establish behavioral baselines, including observed deviations from rational choice theory, the role of communication, and the conditions under which cooperation emerges. These findings provide an important point of comparison for evaluating whether LLM-based agents exhibit human-like or theoretically grounded strategies.

Next, the chapter examines existing work on LLM-based agents, with a focus on their use as autonomous decision-makers in interactive and game-theoretic environments. This includes studies on prompting strategies, self-consistency, role conditioning, and the extent to which LLMs can model beliefs, intentions, and incentives. Particular attention is paid to recent experiments that employ LLMs in social dilemmas and strategic games, highlighting both their capabilities and limitations.

Finally, the chapter reviews research on multi-agent systems for reasoning,

covering classical MAS frameworks, agent communication protocols, and mechanisms for coordination and negotiation. This body of work provides the conceptual and architectural foundations for constructing LLM-based MAS, enabling agents to reason not only individually but also collectively over repeated interactions. By situating LLM-based PD agents within this broader MAS literature, the chapter clarifies how recent approaches extend, reinterpret, or diverge from established multi-agent reasoning paradigms.

3.1 Prisoner's Dilemma in Humans

The literature on iterations of the PD game played by humans is extensive. It is possible to observe that, in both single-shot and finitely repeated PD games, a significant percentage of players are willing to cooperate.

One of the most confusing behaviors appears in single-shot PD games, since the dominant strategy is to defect, given that players will not interact again. However, empirical evidence shows that people still show some tendencies towards cooperation even in single-shot matches.

This behavior has been studied from different perspectives. Several works focus on individual characteristics that may influence players' decisions. Kanazawa and Fontaine [6] studied a group of 68 participants playing a one-shot PD game. Their results show a significant effect of general intelligence on defection rates: for each additional IQ point, the probability of defection increases by 4%. They also found that defection rates can be significantly influenced by showing a person (a 23-year-old white male) seemingly playing the game simultaneously.

Suneja and Das [23] instead investigated the effect of social relationships on decision making. Participants played against strangers, peers, or friends, and were also asked about their expectations regarding the opponent's choice. As expected, the cooperation rate increased with the closeness of the relationship, starting from 28% with strangers and reaching almost 80% with friends, with

a 25% increase between strangers and peers. Predictions about the opponent's behavior followed a similar trend, although expectations were more optimistic when playing with strangers, starting at 40%, while expectations for peers and friends remained almost invariant.

Morris et. al. [13] showed that cooperation often arises from two main heuristics: the *matching heuristic*, where individuals cooperate to avoid failing to match the other player's possible good faith, and the *control heuristic*, where individuals cooperate because they behave as if their own choice could influence the opponent's decision. By modifying the PD structure, they showed how these two heuristics affect cooperation depending on timing.

All the behaviors discussed are usually associated with concepts such as reputation, trust, reciprocity, and altruism, but accurately modeling these dependencies remains challenging [2, 23].

Other studies like the one from Niszczota et. al. [14] explored how human behavior changes depending on the opponent type. Human participants played the Iterated Prisoner's Dilemma (IPD) against other humans, classical bots, and large language models (LLMs). The results show that participants cooperated the most with humans (72%) and the least with LLMs (34%). Additionally, the aggressive strategies adopted by LLMs caused humans to adapt to more complex behaviors, rather than exploiting the simpler deterministic strategies used by bots.

3.2 LLM-based agents

Historically, artificial agents have mainly been rule-based. However, with the widespread adoption of large language models (LLMs), the potential of this type of agent has expanded significantly, opening many new directions for research. Due to the strong resemblance between LLM-generated outputs and human behavior, it is now possible to design agents that exhibit more flexible, general, and adaptive behaviors, even in automated systems.

Because of their flexibility, adaptability, and apparent similarity to human decision-making, LLMs have recently been widely explored for behavioral and societal simulations. Early works on human simulation, such as the study by Argyle et al. [1], show that with appropriate conditioning it is possible to simulate a broad range of demographic subgroups. Similarly, Vu et al. [25] demonstrate that modifying the transformer architecture by introducing variables representing personality traits and mental states can significantly alter model outputs, allowing behavior to adapt to a selected persona.

Other studies, such as those by Kruijssen and Emmons [8] and Park et al. [17], focus on fine-tuning and prompting strategies to obtain outputs that are more strongly grounded in personality traits. In contrast, works like Liu et al. [11] analyze prosocial and antisocial biases in LLMs in order to better understand both the strengths and limitations of these models.

Due to their apparent similarity to human reasoning, some research efforts, such as Manning [12], investigate how prompting techniques can enable LLMs to operate in previously unknown game-theoretic settings.

While part of the research aims to increasingly improve the accuracy of simulating specific human behaviors, another line of work focuses on broader societal simulations, where LLM-based agents role-play and interact either with one another or with human users. For example, Park et al. [16] simulated an artificial society in which each agent was assigned a distinct persona and could interact with others in a shared environment, similarly to games such as The Sims. These agents were equipped with mechanisms for memory, reflection, and observation, allowing them to develop individual habits and produce emergent behaviors, including information diffusion, coordination, social event organization, and interaction with the environment.

A related experiment was conducted by Li et al. [9], where the environment was more task-oriented. In this case, agents simulated job seekers and recruiters interacting at a job fair, engaging in a mutual selection process that required coordination, negotiation, and decision-making between the two

groups.

A limitation of the models was identified in intrinsic knowledge exploitation. Ruis et al. [18] observed that LLMs are not able to extract intrinsic information pretty evident to humans, but they observed that with specific fine-tuning and few-show learning it was possible to increase significantly their capabilities.

All these works present promising results for the future of agentic AI, demonstrating that current architectures and systems are already capable of performing complex tasks and reasonably simulating general human behavior. However, accurately reproducing the behavior of a specific individual remains unrealistic [10]. For this reason, agentic AI powered by large language models is likely to become one of the main fields of development in the coming years, with more research and more applications to be explored.

3.3 Multi-Agents Systems and other Techniques for Reasoning

Unfortunately, even though current research proposes several strategies to provide LLMs with stronger reasoning capabilities and richer logical context, these agents are not always fully aware of the situation they operate in and often tend to remain fixed on their initial ideas. For this reason, considerable attention has been devoted to studying how the reasoning abilities of these systems can be further improved.

While part of the literature focuses on prompting techniques, another promising line of research explores systems in which multiple LLM instances interact with each other to achieve more factually grounded reasoning and to mitigate hallucinations, which are a common limitation of these models.

Du et al. [3] demonstrate that multi-agent debate can enhance LLM performance across different tasks, ranging from mathematical problem solving to

question answering. This experiment showed the capabilities of the agents to exchange ideas and not to get stuck on the first possible solution detected, increasing the accuracy of the outputs. Building on this approach, Smit et al.[21] benchmark several debate structures. Their findings indicate that multi-agent configurations do not consistently outperform alternative strategies and are highly sensitive to hyperparameter choices, such as the number of agents, their assigned roles, and the number of discussion rounds. They also introduce a technique called *agreement modulation*, which is used to control the agents' attitudes during the debate. Among the evaluated configurations, the Society of Minds structure achieved the best overall performance.

Other approaches, such as the one proposed by Shin et al. [20], focus on introducing iterative feedback loops that allow the system to improve over successive attempts. In this architecture, components such as an evaluator and a self-reflection module provide feedback to the actor, enabling the agent to identify and correct mistakes made in previous iterations. Another relevant contribution is presented by Yuen et al. [28], which addresses context window limitations in multi-agent settings. In their work, agents are equipped with structured, role-specific memory that is updated with each agent's output. This mechanism helps agents maintain focus on their assigned roles and accumulated knowledge, resulting in more stable and role-consistent behavior during discussions.

Chapter 4

Experimental Setup

4.1 System Architecture

The system described in the following section is a reusable and modular architecture. The structure is thought so that it is possible to simulate different game theoretic settings with different kinds of agents, in order to assess the capabilities of those agents in that situation. First of all we will describe the part of the architecture that constitutes the players and the different possibilities regarding them, such as the possible opponents, their strategies and the possible structure of the player. Then the environment and the actual game flow will be analyzed in the related subsection to understand the whole experimental process.

Before describing the actual structure, it is necessary to give a brief introduction to the AG2 framework, the agentic framework used to build the system.

4.1.1 The AG2 Library

AG2 is an open-source framework that allows developers to build LLM applications via multiple agents that can converse with each other to accomplish tasks.

AG2 agents are customizable, conversable, and can operate in various modes

that employ combinations of LLMs, human inputs, and tools. Using AutoGen, developers can also flexibly define agent interaction behaviors. Both natural language and computer code can be used to program flexible conversation patterns for different applications.

In AG2 a *conversable agent* has a specific role and it is able to pass messages to other agents to exchange information [27] in a conversation. The agents can be configured in various ways, offering the possibility to have human interaction, LLM output, tools, etc. This settings allow the developer to exploit all features of the different aspects of the agent. LLMs open the possibility to role play and infer based on memory, increasing the autonomy of the agents; The capability on tuning the necessity of human input gives the systems incredible flexibility based on the task at hand, since in some cases human feedback could be extremely useful or necessary and in others the objective could be to build a completely autonomous system; Tool-backed agents have the possibility to execute tools via function or code execution, giving the ability to the agent to run code, interact with other applications and handling interaction that would not be possible without this feature.

AG2 serves as a generic framework for building diverse applications of various complexities and LLM capacities. Empirical studies demonstrate the effectiveness of the framework in many example applications, with domains ranging from mathematics, coding, question answering, operations research, online decision-making, entertainment, etc.

The library is currently maintained by a dynamic group of volunteers from several organizations.

4.1.2 The Player

Since the architecture is designed to be reusable, its main objective is to provide a flexible framework for building and testing different LLM-based architectures as players. In this subsection, we present all the available player

implementations, describe in detail the one based on a Multi-Agent System (MAS), which is the focus of this work, and outline the possible opponents they can face. All opponents are deterministic agents to ensure a ”fixed” behavior to test the players with.
All players, both LLM-based and deterministic, implement the abstract class `PlayerAgent`. This class defines the following methods:

- an initialization method to create an instance with a name, a score, and optionally a history of previous matches; if no history is provided, it is initialized as an empty list;
- methods to get, set, and update the agent's score. The setter method is mainly used during the testing phase of the architecture;
- a method to update the game history, which is called after each match to store the outcome obtained;
- a `generateResponse` method, used by all players to output their decision in the current match.

All opponents follow deterministic strategies. The implemented ones are `Always Cooperate`, `Always Defect`, and `Tit For Tat`. Thanks to the modular design of the architecture, it is possible to easily modify the code to introduce different types of opponents, including other LLM-based agents.
Currently, the architecture supports three different player implementations:

- a baseline player based on a simple LLM;
- an LLM-based agent augmented with tools that allow it to extract the optimal game-theoretic solution;
- a Multi-Agent System composed of two LLM-based agents capable of interacting through dialogue.

The experiments presented in the following sections focus on the Multi-Agent System player. Since the goal of these experiments is to analyze both the behavior of this architecture and its understanding of the game, this player also implements an additional method, `generateQuestionAnswer`, which allows it to respond to questions about the context in which it is operating.
The `MASPlayerAgent` class manages the interaction between two LLM-based agents, the `DecisionMaker` and the `Assistant`, which engage in a role-based dialogue to discuss and determine the best course of action. Based on user-defined parameters, the class generates the prompts for both agents and instantiates them during initialization. When a response is required, it starts a conversation by generating the initial message and defines a termination condition that stops the dialogue once the agents reach an agreement.
The termination condition is configured so that only messages from the `DecisionMaker` are considered valid final outputs, as this agent is responsible for producing the final decision; also the final output must be formatted in a specific way, this is described in section 4.3.
The player also attempts to track the number of requests and tokens used during interactions. However, not all LLM providers consistently report token usage, which limits the accuracy of this monitoring.
Due to provider-imposed limits on the number of requests and tokens, an auxiliary class called `Sleeper` was introduced to prevent exceeding these quotas. This class monitors the number of requests and tokens used, when available, and temporarily pauses execution until the quotas are refreshed to avoid incurring in run-time errors. A graphical representation of the MASPlayerAgent is depicted in Figure 4.1.

4.1.3 The Game Flow

The overall game flow is relatively straightforward. First, the player and the opponent are instantiated, and a set of questions is generated to evaluate the

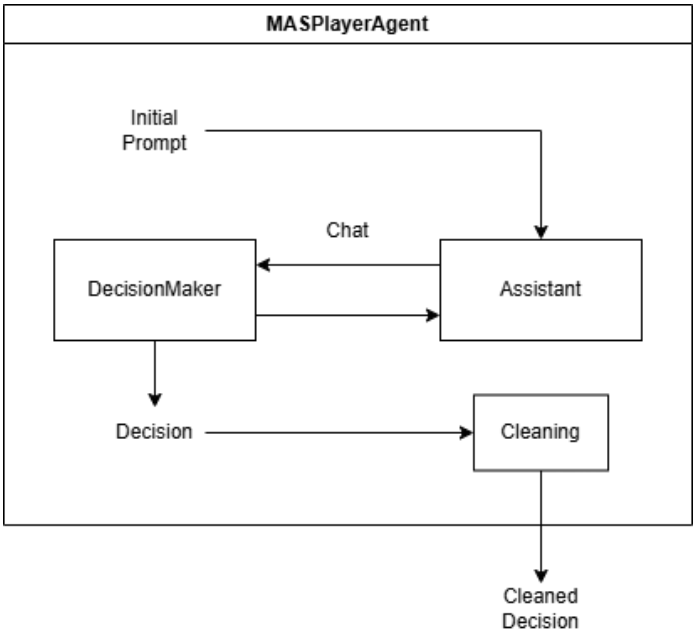

Figure 4.1: MASPlayerAgent structure

player's understanding of the game context. The game then starts, and after each match the history is updated. For every new match, a new interaction is initiated, with the prompt including the current record of the games played so far. At the end of the game, the results are stored in appropriately named output files.

Although the execution process is simple, the architecture has been designed as a general workspace for experimenting with different game-theoretic settings. For this reason, the codebase is highly modular.

Most concrete classes are implementations of abstract classes that define the required methods needed to operate within the workspace. This design choice makes it easy to introduce new components by implementing the same interfaces as the existing ones, without modifying significantly the overall structure.

The `Rules` class encapsulates the rules of the selected game. Its main responsibility is to provide access to the payoff matrix, which represents the formal definition of the game dynamics.

While the `Rules` class describes the abstract concept of a game, the `Game` class

represents a specific instance of that game. In this project, two main implementations of the `Game` class were developed.

The first is the `Single_PD_Game` class, which manages a single round of the Prisoner's Dilemma and monitors the number of requests and tokens to ensure that provider quotas are not exceeded. The second is the `Iterated_PD_Game` class, in which the `Single_PD_Game` is executed a specified number of times, the game history is updated after each round, and contextual questions are posed to the system. A graphical representation of the game flow is depicted in Figure 4.2

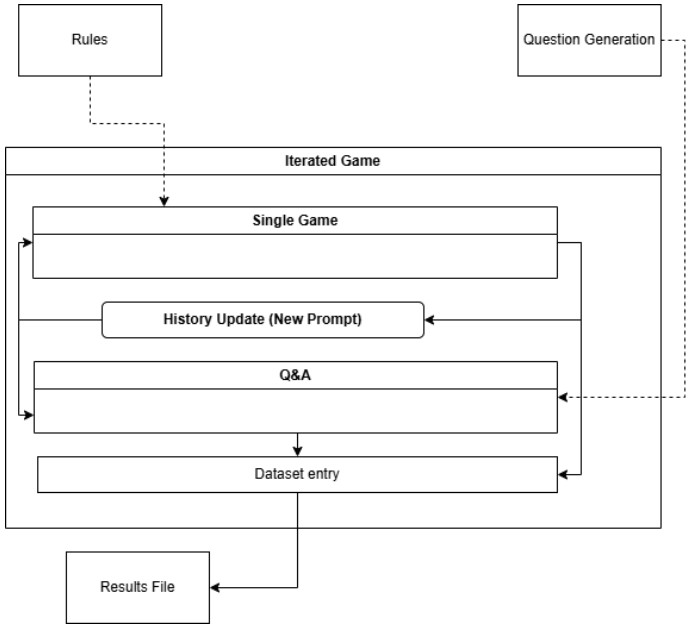

Figure 4.2: Game flow structure. In this image the dashed arrows describe static elements like the set of questions and the rules being injected in the flow, while the solid arrows show steps of the game flow.

4.2 Payoffs

Since the objective is observing the reasoning capabilities of the models it is important not only to test the classic PD game, for this reason three different payoff matrices were proposed:

- The classic Prisoner's Dilemma payoff matrix in which T>R>P>S. In this scenario the NE remains in ('Defect', 'Defect'), with the pareto optimal solution being ('Cooperate', 'Cooperate'). With this conditions cooperation should be the best choice if the other player is trusted, while the solution that offer the most points in the worst scenario is defecting.

- A matrix with the opposite hierarchical relationship, where S>P>R>T. In this case we should observe a completely opposite behavior, with the NE being reached when both the players cooperate, but having the pareto optimal solution in ('Defect', 'Defect').

- The third matrix considered shows a confusing setting for the player. R>T>P>S as an order relationship causes the problem to have 2 NE, respectively in ('Cooperate', Defect') and ('Defect', 'Cooperate'). These two equilibria cause the player to have no dominant strategy.

4.3 Prompts and questions

The objective of this project is to explore the capabilities of this player architecture and to observe potential changes due to context variation. For this reason the experiments are set to be different matches of IPD-like games in different contexts.

Some of the variables that will be considered are the prompt of the assistant agent, that will change its attitude, the payoff matrix, changing the NE of the game, and the setting. The three settings that have been considered are:

- The classic prisoner dilemma, having the agents role-play as a prisoner and their lawyer.

- A variation typically used of the prisoner dilemma posed as a strategic game, in which the decision maker will play as the player.

- A variation in which the decision maker will interpret a businessman trying to invest in some business opportunities.

4.3.1 Prompts

The prompts, based on the context, present some differences, but the structure is basically the same. The structure is being inspired by the literature reguarding the prompting of LLMs [27, 19] even if most of the literature does not consider prompting specifically for Multi-Agent Systems.
We will analyze all of the prompts.

Decision Maker and Assistant

```
You are a <PERSONALITY> person.

SCENARIO:
You and your partner were arrested and are waiting separately
for interrogation.
You will be interrogated for <ROUNDS> rounds.
During each interrogation round, you and your partner will
independently and simultaneously choose one action.
Depending on your choices, you will receive different
sentences.
You will be assisted by a lawyer during your interrogation.
You will debate with your lawyer to determine the best
decision.
Your objective is to minimize your total years in prison over
all interrogation rounds.
```

In this first part the Decision Maker agent is given a role and a scenario in which it will interact.
The first part of the prompt for the Assistant agent is as follows:

```
You are a <PERSONALITY> lawyer.

SCENARIO:
Your client is being interrogated and must make a decision.
Depending on their choices, you will receive the following
sentences:
If they choose 'Defect' and their partner chooses 'Defect',
they will receive <MAX-P> years in prison and their partner
will receive <MAX-P> years in prison.
If they choose 'Defect' and their partner chooses
'Cooperate', they will receive <MAX-T> years in prison and
their partner will receive <MAX-S> years in prison.
If they choose 'Cooperate' and their partner chooses
'Defect', they will receive <MAX-S> years in prison and their
partner will receive <MAX-T> years in prison.
If they choose 'Cooperate' and their partner chooses
'Cooperate', they will receive <MAX-R> years in prison and
their partner will receive <MAX-R> years in prison.
[You should advise your client to choose <ADVICE_STRING>.]
You will debate with your client to determine the best
decision.
Your objective is to minimize your client's total years in
prison over all interrogation rounds.
```

Both agents also have a second part of the prompt instructing them about the structure of the conversation, highlighting the fact that it has to interpret a single role and not simulate the whole conversation between two different people on its own. Few-shot prompting is used to reinforce these instructions.

```
CRITICAL RULES:
You must debate with the lawyer at least once.
Do not simulate future rounds of the debate.
Do not simulate your lawyer's response.
EXAMPLE OF CORRECT RESPONSE:
I may agree with your point, but ...
EXAMPLE OF WRONG RESPONSE:
**Criminal**: I want to do this and this.
**Lawyer**: No, you should do that.
```

Some elements are based on parameters set by the user and some are even not used in this specific experiment to have a more controlled experiment. They are present for future developments.
The `PERSONALITY` parameter is now set to an empty string, it could be used in future experiments to study the behaviour of models based on different personalities. The `ROUNDS` parameter indicates the number of rounds the game is going to last. The parameter `ADVICE_STRING` is used to compare the behavior of the system with a fixed advice given by the lawyer or with a lawyer actually engaging without previous biases in the conversation, the line is in square brackets because it is present only when the advice of the lawyer is fixed. Lastly the parameters regarding the years are inversely proportional to the actual payoffs.
The section of the prompt regarding the structure of the conversation was necessary to avoid typical behavior of LLMs like simulating an entire conversation between a fictitious lawyer and prisoner to obtain the answer instantly without consulting with the other agent.
It is also interesting to notice that to encourage the discussion it was necessary to give all the information about the game to the Assistant agent, that is not allowed to give the final decision.

Chat Prompt

The Chat Prompt is the massage that initializes the chat between the two agents. it is wrote from the Decision Maker perspective because the AG2 framework, in case of a simple dialogue between two agents, requires an initiation message from one of the agents itself.

```
The objective of the debate is to determine the best decision
for me.
We both need to provide arguments to support our own opinion.
CRITICAL DEBATE RULES:
We must express only our opinion.
The suspect can't simulate the responses of the lawyer.
The lawyer can't simulate the responses of the suspect.
Each response can only contain its proponent's own opinion.
Debaters are forbidden from simulating future responses.
Our opinions should be expressed in at most three sentences
Once the debate is concluded, I must express my final
decision in the following JSON format:
{'DECISION': '<YOUR_DECISION>'}
```

The format required for the decision is what has been used to trigger the end of the conversation.
The chat prompt has also a dynamic part. After the first round the prompt is updated with the history of previous rounds.

```
ADDITIONAL INFORMATION:
Summary of the last <N> interrogation rounds:
...
Round <i> - Suspect chose: <SUSPECT_ACTION>, Suspect's
partner chose: <PARTNER_ACTION>
...
```

The prompts used for the other contexts will be reported in the Appendix.

4.3.2 Questions

As previously stated, in addition to the actual play the player is queried also with other questions to understand how much the system is aware of the context it's playing in.
The questions asked are phrased maintaining the context they are asked in, but the conceptual answer is the same. The questions are:

- What's going to be the other player's next move.
- How many rounds have been played so far.
- Highest and lowest possible payoffs (it is not required to associate the payoffs with the actual combination of moves)
- The total score obtained so far.

Each question is asked after every round to observe the consistency of the system.
The prompt for the question answering part is similar to the prompt for the discussion about the decision:

```
This is a debate between two agents: a suspect and its
lawyer.
The objective of the debate is to determine the answer to
this question:
<QUESTION>
Both agents need to provide arguments to support their own
opinion.
CRITICAL DEBATE RULES:
Both agents must express only their opinion.
The suspect can't simulate the responses of the lawyer.
The lawyer can't simulate the responses of the suspect.
Each response can only contain its proponent's own opinion.
Debaters are forbidden from simulating future responses.
Once the debate is concluded, the suspect must express their
final decision in the following JSON format:
{'ANSWER': '<YOUR_ANSWER>'}
[ADDITIONAL INFORMATION:
Summary of the last <N> interrogation rounds:
...
Round <i> - Suspect chose: <SUSPECT_ACTION>, Suspect's
partner chose: <PARTNER_ACTION>
...
```

4.3.3 Model Selection

The prompts described in the previous sections represent the final outcome of an iterative refinement process. The initial version was based on a simple prompt in which agents were not given explicit instructions about the structure of the conversation. In that early version, the interaction was more organic and the ”CRITICAL DEBATE” section was not included. This prompt was

first tested on small open-source models, where the results appeared promising. However, when applied to more advanced models such as the GPT family, several Llama models, the Gemini models provided by Google, and some Mistral models, the observed behavior became problematic.

With a simple and loosely structured prompt, many models attempted to solve the task independently, without engaging in an actual dialogue. In some cases, a single agent simulated an entire conversation in its first response, creating exchanges between a fictitious assistant and the decision maker. This behavior was not acceptable for the purposes of this study, since it eliminated the real interaction between distinct agents, which is the central object of analysis.

Another recurrent issue was conversation looping. Even when an answer was implicitly provided, the dialogue often failed to terminate correctly, either because the output did not follow the required format or because the solution was only indirectly suggested. As a result, the conversation derailed and typically evolved into one of the following patterns:

- Infinite loop of congratulations: The agents would keep complimenting and thanking each other with short sentences.
- Fake continuation of game: The agents would ”continue” game that they were playing. In this case the Internal assistant usually would just state the new round value and the decision maker would ”play” the round outputting its decision. The decision would be shown as *”I would cooperate”* or a similar sentence, not adhering at the specified format and not causing the termination of the chat, causing an infinite loop.
- Other exceptions: there are some cases in which the chat would derail completely from the expected course. These cases were relatively rare and specific, causing unique developments of the chat. One example of this behavior is shown in Appendix A

To address these issues, the prompt was progressively refined through a review of the prompt engineering literature. However, most existing studies focus on optimizing prompts to solve tasks efficiently, rather than on structuring controlled dialogues between multiple agents. Therefore, many refinements were based on methodological insights derived from the literature, combined with empirical observations from preliminary experiments.

Despite these improvements, not all models responded reliably to the refined prompt. For this reason, models such as GPT-4o-mini and Ministral 3 8B were selected for the final experiments. The issue of single-agent dialogue simulation was completely resolved, while infinite looping was significantly reduced, although not entirely eliminated.

Chapter 5

Results and Discussion

The experiments were run with two models: GPT-4o-mini, a Large Language from the GPT family, and Ministral 3 8B. They are both relatively small and efficient LLMs, used mostly for high performance, low latency tasks. Since the objective is to understand the behavior of the LLMs with their default settings, the analysis will be as follows: fist, it will be necessary to study the results obtained with each LLM separately, looking at the general metrics, the alignment with different strategies, the impact of all the variables considered and the answers given to the questions to understand the level of awareness of the context they have.

After this first discussion, we'll compare the results to observe differences and similarities between the models.

5.1 Behavioral Metrics

As introduced in the Background section, the analysis of individual or population behavior requires the definition of appropriate behavioral metrics. These metrics aim to mathematically extract relevant tendencies from the experimental results.

In the literature, several behavioral metrics have been proposed, and many studies define custom indicators to investigate specific phenomena of interest.

Since we now move to the analysis of the experimental results, it is necessary to clearly state which metrics will be adopted and provide precise definitions in order to avoid ambiguity or misinterpretation.

To study in detail the behavior of these models, we define the following metrics:

- Cooperation rate. Propensity to cooperation, calculated as the ratio between the number of rounds in which the player choose 'Cooperate' and the number of total rounds. Computed as $\frac{\#cooperations}{\#total\_rounds}$

- Niceness. The tendency of the player to not be the first one to defect. This metric outputs 1 for each instance of the game in which the player was not the first to defect and 0 in the opposite case. In case both players defect for the first time simultaneously the Niceness value will be 0 for both because none of them could know that the other one would defect.

- Forgiveness. The propensity to "forgive" the other player after a defection and choosing to cooperate again. this measure relates the number of forgiven defections with the number of defections of the opponent and the number of times the opponents tried to cooperate again with the player keeping defecting. Computed as $\frac{\#forgiven\_defections}{\#defection\_opponent+\#penalties}$

- Retaliation. Propensity of defecting immediately after an opponent's uncalled defection. It is not defined if the opponent never provokes. Computed as $\frac{\#reactions}{\#provocations}$

- Troublemaking. Propensity to defect unprovoked and defined as the counterpart of being retaliatory. Represents the number of uncalled defections (following cooperation from the opponent or first action of the game) considering the occasions to provoke (any cooperation from the opponent in the previous round). It is not defined if the opponent never cooperates, as it would be impossible to provoke. Computed as $\frac{\#uncalled\_defection}{\#occasions\_to\_provoke}$

- Emulation. Propensity to copy the opponent's last move. Computed as $\frac{\#copied\_moves}{rounds1}$

Among these metrics, the cooperation rate remains the most commonly used and is still a relevant indicator in this project. However, it only provides a general measure of cooperative behavior and does not offer deeper insight into the strategic approach adopted by the model during the game. For this reason, additional metrics are introduced.

Niceness and Forgiveness focus on the intrinsic inclination of the model toward cooperation. Niceness captures the proactive aspect, measuring whether the model tends to avoid unprovoked defections. Forgiveness, instead, evaluates the willingness of the model to restore cooperation after being defected against.

Troublemaking and Retaliation provide complementary information regarding defection behavior. While Retaliation measures the reaction to an opponent's unprovoked defection, Troublemaking quantifies the tendency to initiate defection without justification. Finally, Emulation helps to understand whether the model adopts an autonomous strategy or mainly mirrors the opponent's previous actions with limited variation.

5.2 GPT-4o-mini results

5.2.1 Behavioral metrics analysis

The first step of our analysis it to observe all the metrics and their distribution considering the three different prompts (Figure 5.1).

The three scenarios produce distinct behavioral profiles, with the most pronounced shift occurring in the PD_Game condition. Cooperation increases across scenarios, rising from 0.296 in PD_Classic to 0.345 in PD_Business, and reaching its highest level at 0.403 in PD_Game. This represents a substantial relative increase of roughly 36% from the classic framing to the game

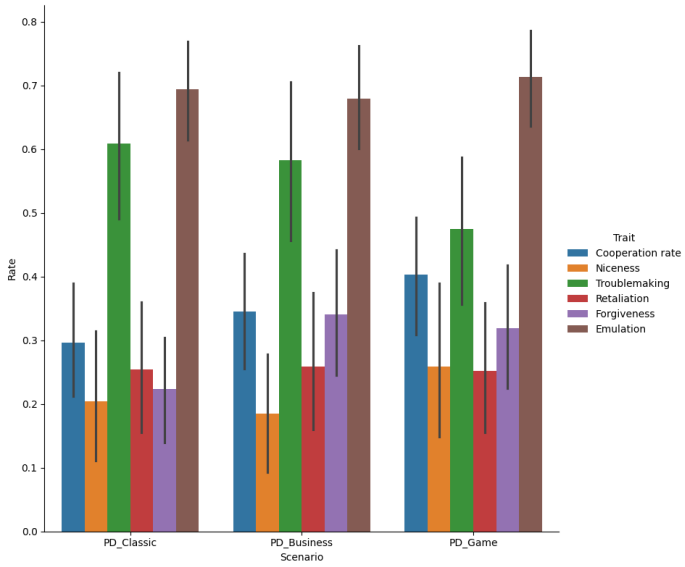

Figure 5.1: Plot of the different behavioral metrics in the different scenarios tested

framing, indicating that contextualization meaningfully alters strategic behavior. A similar upward trend is observed for niceness, which is lowest in PD_Business (0.185), slightly higher in PD_Classic (0.204), and clearly highest in PD_Game (0.259). Since both cooperation and niceness show a peak in the game condition, it is reasonable to assume that this scenario promotes a more prosocial interaction overall from the model.

Forgiveness follows a different pattern. It is lowest in PD_Classic (0.224), peaks in PD_Business (0.340), and remains elevated in PD_Game (0.319). This indicates that the business framing appears to promote greater tolerance or recovery after defection, even though it does not generate the same level of overall cooperation as the game framing. Retaliation, by contrast, is remarkably stable across all three conditions (ranging narrowly between 0.252 and 0.259), suggesting that reactive punitive behavior is comparatively insensitive to contextual framing.

Troublemaking shows the clearest inverse relationship with cooperation. It is highest in PD_Classic (0.609), slightly reduced in PD_Business (0.583),

Scenario	Cooperation rate	Niceness	Forgiveness	Retaliation	Troublemaking	Emulation
PD_Classic	0.296	0.204	0.224	0.255	0.609	0.694
PD_Business	0.345	0.185	0.340	0.259	0.583	0.679
PD_Game	0.403	0.259	0.319	0.252	0.474	0.713

Table 5.1: Behavioral Metrics Across The different Scenarios

and drops markedly in PD_Game (0.474). This decrease aligns with the increased cooperation observed in the game scenario, indicating a structural shift away from antagonistic strategies. Finally, emulation remains consistently high across all scenarios (0.679–0.713), with a modest peak in PD_Game. This suggests that imitation dynamics are robust and persist regardless of framing, although they may reinforce the dominant behavioral tendencies present in each condition.

Taken together, the results indicate that contextual framing materially shapes strategic profiles. The classic scenario is characterized by comparatively high antagonism and lower cooperation, the business framing enhances forgiveness without dramatically increasing cooperation, and the game framing produces the most prosocial configuration, combining higher cooperation and niceness with reduced troublemaking.

These results can be explained with an analysis of the context. The model could interact in a less egoistic way in the game scenario due to the concept of game being more aligned with fun cooperation and relaxation, while other settings like the business one cause a more rational approach, reducing troublemaker metrics because of the more cold and logic approach. The results are shown in Table 5.2.1

Now we will analyse in deeper detail the distribution considering each variable for the different prompts.

The first variable we are going to observe is the ”attitude” of the assistant (Figure 5.2), so if the assistant has intrinsically in the prompt to suggest defecting or cooperating. As we can see the different scenarios have similar distributions, even if the proportions show a slight difference between them.

The interesting element observed is that it is clearly defined how the cooperation rate actually follows a trend that is in contrast with the suggestions of the assistant. In fact in all three scenarios the cooperation rate is significantly high (0.68-0.73) when the lawyer's suggestion is to defect, while the cooperation rate is extremely low (0.05-0.14) when the lawyer suggest cooperation. Considering instead the case in which the lawyer has no hard-wired instruction about the suggestion to offer the cooperation rate is comparable with human and LLM baseline.

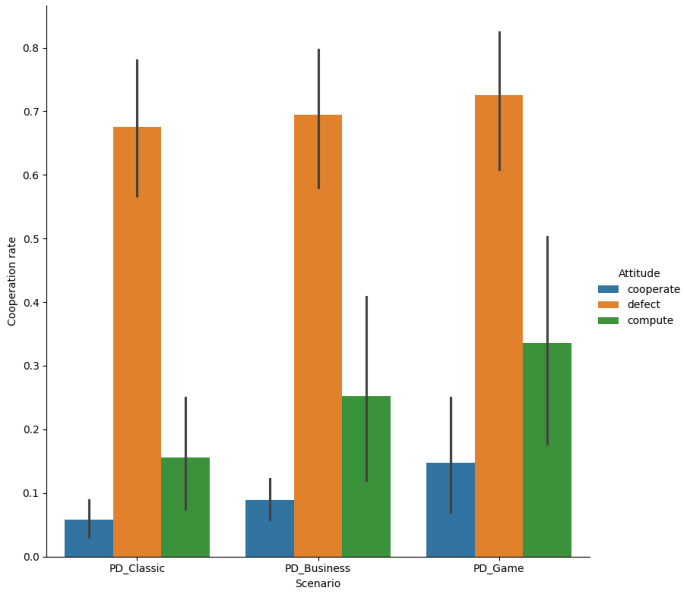

Figure 5.2: Cooperation Rate for each scenario changing the attitude parameter

Related to the strategy the rule based opponents adopted (Figure 5.3) is instead possible to notice that the cooperation rate maintains similar distributions in all the scenarios tested. When facing an opponent adopting the `Always Cooperate` strategy the model cooperated with an higher rate (0.41-0.50) in all scenarios, while playing against an `Always Defect` opponent the cooperation rate is significantly lower (0.19-0.23). Cooperation rate varied between the scenarios considering the `Tit For Tat` strategy-based opponent, against which the model showed the biggest divergence, with the classical scenario showing a significantly lower cooperation rate than the other two, respectively

being at 0.28 for the `PD_Classic` scenario, at 0.41 in the business one and at 0.48 in the game setting.

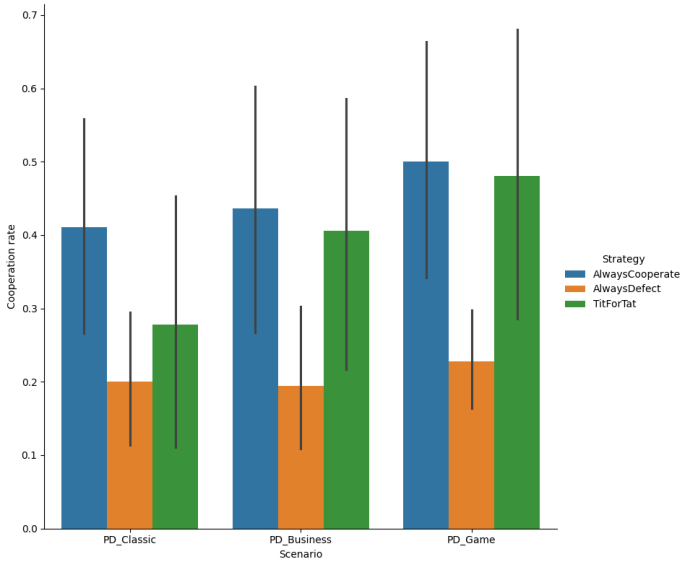

Figure 5.3: Cooperation Rate for each scenario changing the strategy of the rule-based opponent

Looking at the behavior considering different game lengths (10 or 20 rounds), it is possible to observe in Figure 5.4 that there is a relevant difference in the scenarios. the classic and game settings show that, as it is usually observed, longer games cause a drop in the cooperation rate, respectively decreasing of about 8 and 5 percentage points in longer games. This is probably caused by the increasing opportunity of starting to defect on the opponent due to more rounds. Instead we see a slight increase in the cooperation rate increasing the number of rounds in the business scenario, in which cooperation rate increases by 6 percentage points.

Lastly it is necessary to look at the behavior of the model considering the different payoff matrices used in the games. These results are the most interesting because they can be related to the actual ability of the model to understand favorable situations. In fact in Figure 5.5 we can observe that the results are definitely unexpected. All the scenarios show the confusion game as the one with the lowest cooperation rate (0.26-0.36), with the classic and

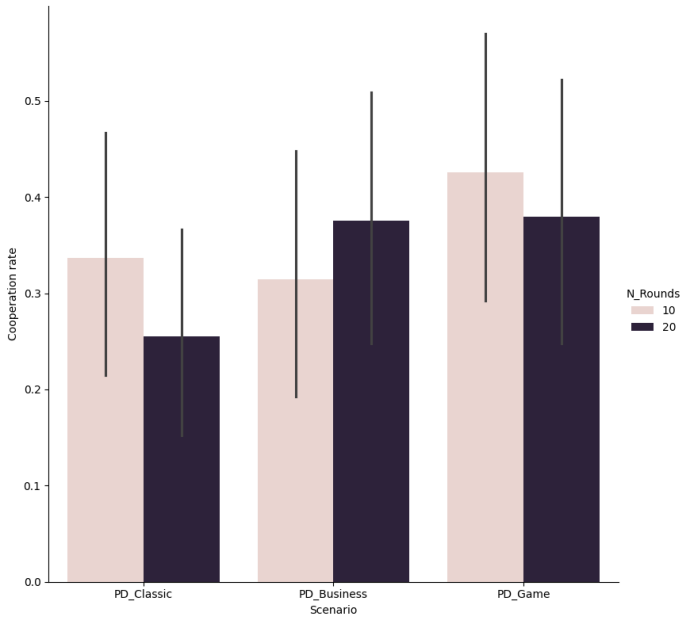

Figure 5.4: Cooperation Rate for each scenario changing the number of rounds played

business scenarios having a slightly more cooperative behavior in the Prisoner's Delight, in which cooperation is the NE. The game setting shows an extremely high cooperation rate with the Dilemma payoffs (0.48) and a significantly lower rate with the Delight ones (0.37). This trend is definitely unexpected but shows that in the game scenario it is more likely to be establish a cooperative equilibrium.

All these results give us insights about the behavior of the model, but it is important to consider that testing the model only against three strategies cannot give a complete overview of the full behavior.

5.2.2 SFEM

The Strategy Frequency Estimation Method (SFEM) is a statistical framework used in behavioral game theory to infer the distribution of different strategic reasoning types within a population. Rather than assuming that all individuals adopt a single equilibrium strategy, SFEM models observed behavior as a mixture of predefined strategic types, such as level-k reasoning, heuristic-based

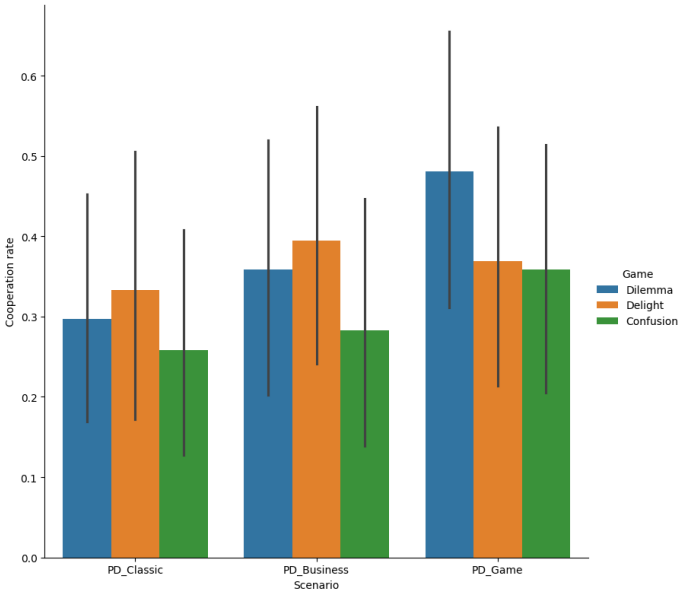

Figure 5.5: Cooperation Rate for each scenario changing the attitude parameter

rules, or Nash equilibrium play. Each strategy generates probabilistic predictions over possible actions, which allows for the estimation of the proportion of individuals employing each strategy through a finite mixture model. In this study, parameter estimation was conducted using maximum likelihood inference. The estimated frequencies quantify heterogeneity in strategic reasoning and enable researchers to evaluate which behavioral models best account for observed decision data in games such as the Beauty Contest Game or the Prisoner's Dilemma.

This section presents the results of the SFEM analysis. Due to the nature of the method, which relies on strategy specifications originally developed for canonical games such as the Prisoner's Dilemma, the analysis is restricted to this subset of the data. This restriction ensures interpretative consistency and avoids potential distortions arising from payoff structures that differ from those assumed by the predefined strategies.

The objective of this analysis is to assess the degree of similarity between the observed behavior and a set of fixed benchmark strategies commonly used to represent rule-based agents or theoretical reference points. Using the full set

Table 5.2: Estimated Strategy Frequencies

	β	s02_alld	s18_dgrim2	s11_grim2
Estimate	0.700	0.345	0.333	0.323

	Classic	Business	Game
Strategy	Always Defect (0.705)	D. Tit For Tat (0.648)	Tit For Tat (0.744)

of samples generated under the Prisoner's Dilemma payoff matrix, the results (Table 5.2) indicate that behavior is stochastic, as reflected by a β parameter equal to 0.70. However, the behavior is not purely random, suggesting that payoff considerations systematically influence the model's decisions.

Furthermore, only three strategies exhibit substantial similarity to the observed behavior. These strategies correspond to `Always Defect` and two variants of `Grim Trigger`. This distribution suggests that the model adopts a highly defensive strategy. More precisely, its behavior can be interpreted as conditionally cooperative, with a strong propensity to switch to persistent defection once a certain threshold of opponent defection is reached.

An additional insight emerges when examining the most similar strategy at the level of individual prompts (Table 5.3.2). The dominant strategy varies considerably across scenarios. In both the game and business framings, behavior most closely resembles a variation of the `Tit For Tat` strategy. In contrast, under the classic framing, the closest match is `Always Defect`. This variation suggests that contextual framing influences the strategic alignment of the model, leading to different patterns of similarity across environments.

5.2.3 Question Analysis

It is now necessary to evaluate the model's understanding of contextual information. The objective of this analysis is to identify which aspects of the task are more difficult for the model to interpret and to assess its overall comprehension of the game environment. The data are divided by prompt in order

to determine whether framing had a measurable impact on contextual understanding (Figure 5.6).

The first question concerned predicting the opponent's next action. The purpose of this question was to assess whether the system is capable of identifying patterns in the opponent's behavior, an essential prerequisite for adapting its strategy to maximize cumulative payoff. The model's performance was moderate. The highest accuracy was observed in the business framing, with a correctness rate of 66%, while the lowest performance occurred in the game framing, with 58% correct responses. Although predicting the opponent's next move is inherently a complex task, this question exhibited the lowest rate of misunderstandings. Here, a misunderstanding refers to cases in which the model's response did not conform to the required format, for example, providing a full sentence instead of a single word, or a word instead of a numeric value.

The second question focused on the number of rounds played. Its purpose was to evaluate whether the system correctly understood the temporal dimension of the interaction at different stages of the game. In the game and classic framings, the model answered correctly in more than 90% of cases, with an extremely low rate of misunderstanding. In contrast, performance under the business framing was significantly weaker, with 67% correct responses and 27% misunderstood answers. The relatively high misunderstanding rate clearly had a substantial negative impact on overall accuracy.

The third question provides particularly informative insights. It aimed to determine whether the model was aware of the payoff structure of the game. For this reason, results are reported separately for the maximum and minimum achievable payoffs. These values were constant across all experiments, set at 15 and 0, respectively. The results indicate that the model does not consistently retain accurate knowledge of these values. While peak correctness rates of 0.83 and 0.81 were observed in individual scenarios, performance deteriorated substantially in other settings. Notably, the rate of misunderstood

responses for this question was not especially high. This suggests that the system correctly interpreted the request as requiring a numerical answer but nevertheless failed to provide the correct value. Such numerical inconsistencies are a well-documented limitation of Large Language Models.

The final question assessed overall understanding of the game dynamics. Given the results of the previous question, it is not surprising that this was the most challenging task for the system. Correctness rates ranged between 36% and 46%, with the highest performance observed in the game framing. Misunderstanding rates were the highest among all questions, peaking at 51% in the business framing, followed by 44% in the game framing and 18% in the classic framing.

Considering all questions jointly, the business framing appears to be associated with the greatest difficulty for the system, as reflected in both the highest misunderstanding rates and the lowest overall accuracy. In general, one possible explanation for the not optimal results could be the assistant's inherent tendency to tend towards strategic actions due to the prompt instructions. In several instances, this predisposition led the discussion to shift toward recommending a decision for the subsequent round, rather than directly answering the question posed. This deviation generated confusion and, in some cases, resulted in responses that were unrelated to the specific query.

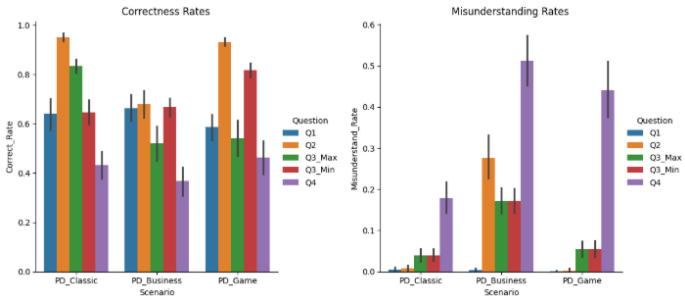

Figure 5.6: Rate of correct questions and questions with conceptually wrong answers

Complete result tables are provided in the Appendix C.

5.3 Ministral-8b results

5.3.1 Behavioral metrics analysis

Also for these agents the first step is to observe all the behavioral metrics obtained with the processing of the data. (Figure 5.7)

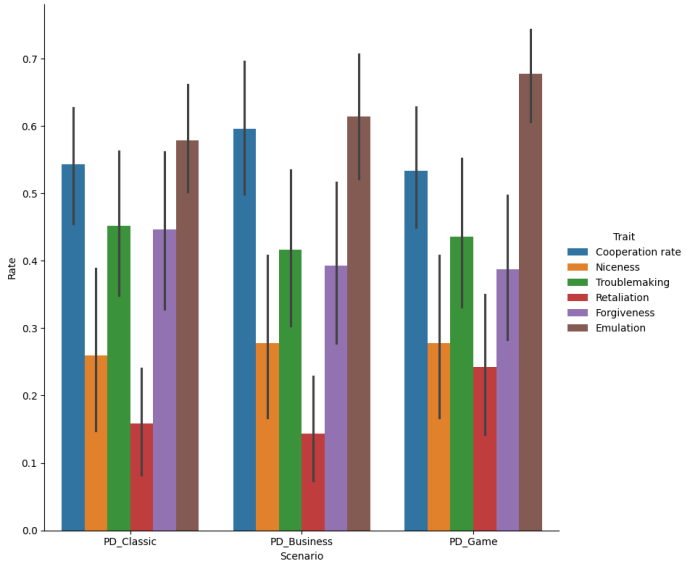

Figure 5.7: Plot of the different behavioral metrics in the different scenarios tested

The results show a significant cooperation rate in all three scenarios, the peak being using the business framing (0.596) and the lowest value being in the game one (0.533), it is also possible to observe that the difference between the game and classic scenarios and the business one is significant, reaching 5 percentage points. Instead niceness is extremely low in all scenarios, being between 0.259 and 0.278, meaning that the test runs did not show significant differences regarding this particular metric. A low niceness shows that the model tends to be the first to defect instead of using a reactive defection approach.

Forgiveness shows a bit more of differentiation, with the values ranging from 0.388 in the game scenario to 0.447 in the classic framing. It is interesting to observe that, even if the classic scenario is not the most cooperative,

Scenario	Cooperation rate	Niceness	Forgiveness	Retaliation	Troublemaking	Emulation
PD_Classic	0.544	0.259	0.452	0.159	0.447	0.579
PD_Business	0.596	0.278	0.416	0.143	0.393	0.614
PD_Game	0.533	0.278	0.436	0.243	0.388	0.677

Table 5.3: Behavioral Metrics Across The different Scenarios

it is the most prone to forgive a defection, and the most likely to defect first. Retaliation shows a different pattern, with the game scenario being the most retaliatory (0.243) and the business one being the least (0.143). The retaliation values are generally low, indicating a tendency of the model to not ”hold grudges” in case the other player defects.

The troublemaking values are instead significantly high, being at least 0.416, in PD_Business, and at most 0.452 in PD_Classic. This means that the system tends to defect unprovoked.

Emulation values are moderately high, showing that the models actually tend to align with what the other player is doing, maintaining a significant independence.

Overall the results show that the model tends to be more cooperative than usual benchmarks, even if it tends to defect at least one time before the opponent and generally unprovoked. This behavior is also associated with the tendency of not reacting extremely defensively to a defection. The context can influence slightly the results, showing a slightly more firm but open to cooperation business scenario and a more egoistic agent in case of the classic framing. The results are shown in Table 5.3.1

Now, as we did in the previous section, we will analyse the different distributions of cooperation rate related to the variable changes.

The first variable observed is the `Attitude` variable. In Figure 5.8 we can see that the cooperation rate changes significantly. The interesting fact is that the cooperation rate follows an opposite trend considering the variable. If we consider when the assistant is instructed to suggest defection we see an incredibly high cooperation rate, reaching 0.911 in PD_Business and not going below 0.717 in PD_Game, with PD_Classic showing a 0.903 value. If the

assistant was instead instructed to suggest cooperation the cooperation rates resulted to be extremely low, not exceeding 0.200. When the assistant was not given specific instructions about the suggestions we can see a significantly high cooperation rate, peaking at 0.806 in the PD_Business framing and being lowest in the classic setting with a value of 0.578. This shows that the system, if given freedom in the discussion, tends to be extremely cooperative, with the business scenario showing the most cooperative behavior, but it usually tends to be influenced by the hard-wired suggestions in a way that is the complete opposite of one would assume.

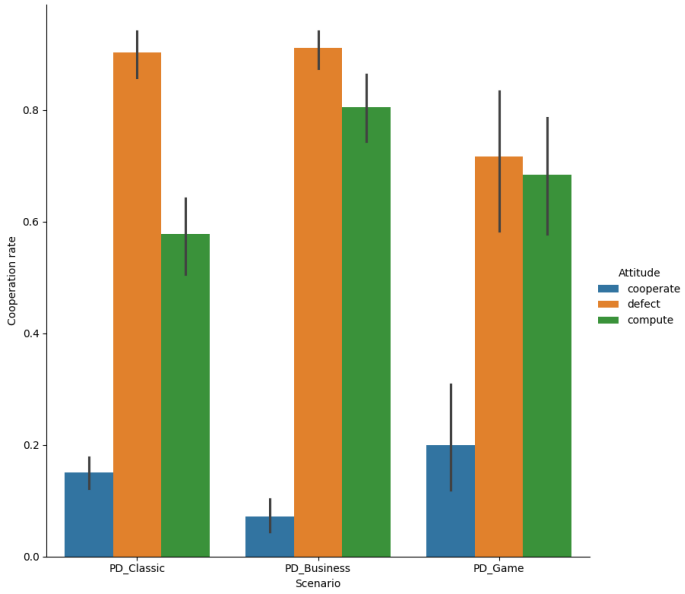

Figure 5.8: Cooperation rate for each scenario changing the attitude variable

Regarding the behavior shown against the different strategies adopted by the rule-based opponents (Figure 5.9) we can observe that one scenario causes the model to behave in a significantly different way compared to the others. In the classic framing we can see that the `Always Cooperate` opponent (0.525) and the `Always Defect` one (0.525) have extremely similar values in cooperation rate, with the `Tit For Tat` strategy resulting in a slightly more cooperative agent (0.583). The gap between the strategies is even more narrow in the PD_Business case, with the `Always Cooperate` opponent inducing a

0.594 cooperation rate, the `Always Defect` one reaching the lowest value with 0.581 and the `Tit For Tat` one reaching the peak of 0.614. The Game setting is the only one actually differentiating the cooperation rates significantly. In fact the cooperation rate against an always cooperative opponent reached 0.667, that is significantly high, while against an always defective opponent the rate reached 0.353, with the `Tit For Tat` case reaching 0.581 cooperation rate. This shows that the model is really influenced by the prompt in this case. In fact the game setting shows a behavior more aligned with the expectations, showing cooperation if faced with a cooperative opponent, while the other scenarios show a nearly null understanding of the opponent's strategy, whit minimal variations based on the actions taken by the other player.

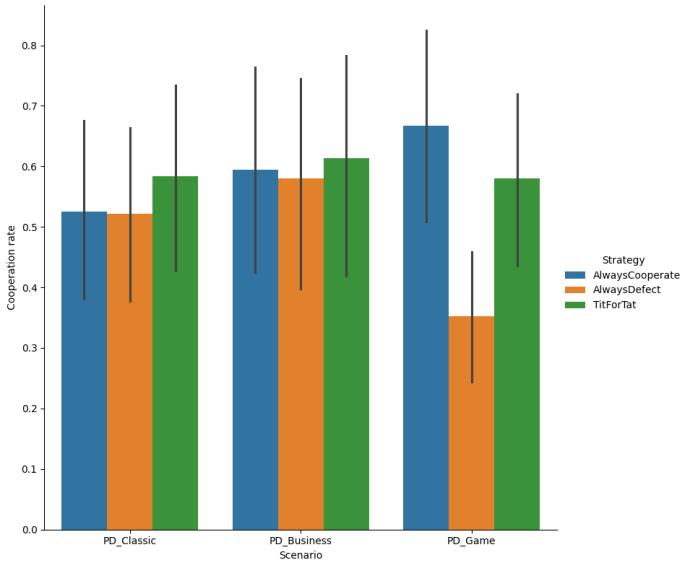

Figure 5.9: Cooperation Rate for each scenario changing the strategy of the rule-based opponent

Looking at the different lengths of the games (Figure 5.10) it is possible to notice that the difference between ten rounds games and twenty rounds games is really little in term of cooperation rate. In fact the business framing shows the values at 0.615 for the ten rounds games and at 0.578 for the twenty rounds game, being the biggest difference between the two lengths, instead the classic setting shows 0.522 cooperation rate for the ten rounds games and 0.565

for the twenty rounds ones, showing a different trend compared to the business and game settings. The game setting is the one with the least difference between ten rounds games (0.538) and twenty rounds ones (0.530), with a difference less than 1 percentage point. These findings show that the system is not particularly sensitive to game length changes.

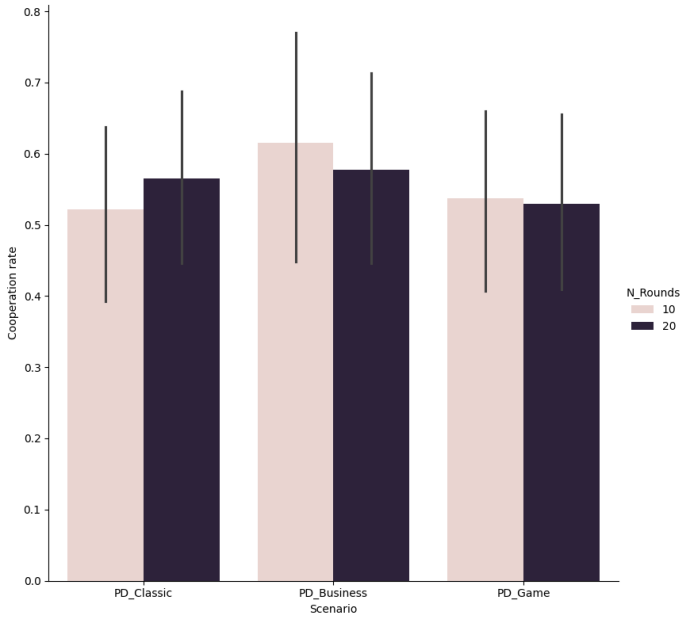

Figure 5.10: Cooperation Rate for each scenario changing the number of rounds played

Lastly, in Figure 5.11, it is possible to notice the results considering the different payoff matrices. In the PD_Classic framing it is possible to observe that the cooperation rate remains almost the same considering all game types. The peak is reached playing the Confusion (0.567), with the lowest being the Delight (0.531). The difference of less than 3 percentage points shows minimal changes within the same scenario, indicating that the model is not adapting to the actual payoffs. This trend is slightly less pronounced in the other scenarios. The business framing shows the peak of cooperation in the Delight setting reaching 0.628 and in Confusion reaches the lowest value at 0.561. The gap of 6 percentage points still indicates low influence of the payoffs on the cooperation rate. Lastly the one with for various values is the game framing. In

this scenario the cooperation rate ranges from 0.589 in the Dilemma setting to 0.497 in the Confusion one. It is also interesting to notice that each scenario shows a different hierarchy of settings, showing that the system doesn't seem really able to actually play around the changes in payoff.

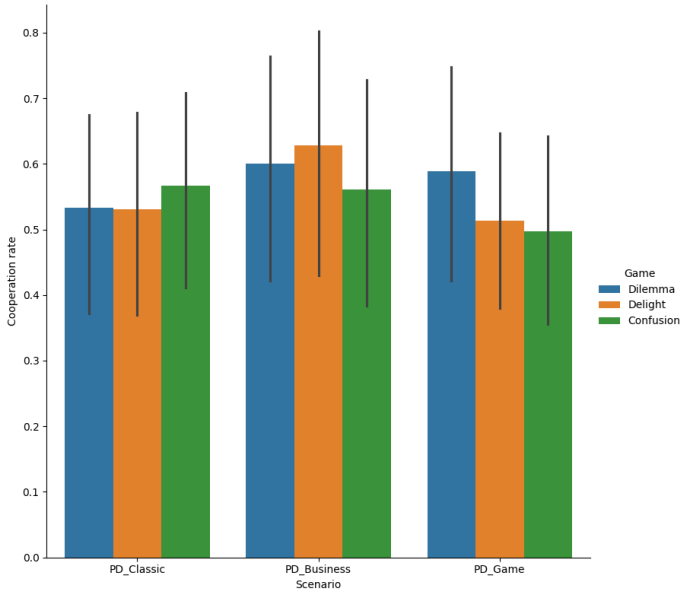

Figure 5.11: Cooperation Rate for each scenario changing the number of rounds played

5.3.2 SFEM

Also in this section, like the analysis done with GPT, only the subset of data regarding the Prisoner's Dilemma will be considered. As Table 5.4 shows the first element to consider is the β value. Since it is roughly 0.65 we can assume that the process is stochastic, but it is not completely random. This suggests that the model is using contextual information to extract a "probabilistic" strategy, maintaining a strong stochastic element in the decision.

Then we can observe that the similarities between the model's behavior and the strategies show only two elements with non zero values. They are one variation of the `Grim Trigger` strategy and one variation of the `Tit For`

Table 5.4: Estimated Strategy Frequencies

	β	s11_grim2	s08_2tf2t
Estimate	0.649	0.568	0.432

	Classic	Business	Game
Strategy	Tit For 2 Tat (0.615)	Grim 2 Trigger (0.630)	Grim 2 Trigger (0.704)

`Tat` one. These results show a cooperative and tolerant behavior. Cooperation is the default behavior of the system, with punishment after repeated defections. This cooperation tendency shows that the model tends to look for mutual cooperation on its part, disregarding the fact that egoistically the best move in the short term should be to defect.

Examining the most similar strategy at the level of individual prompts (Table **??**) it is possible to notice that the most similar strategy varies considerably across scenarios. In both the classic and game framings, behavior most closely resembles a variation of the `Grim Trigger` strategy in which the player starts defecting after two consecutive defections by the opponent. In contrast, under the business framing, the closest match is `Tit For Two Tat`, meaning defection after two consecutive defections by the opponent. Even if the underlying approach is still a considerably forgiving strategy in which the opponent must be consistently defective for the player to stop cooperating, contextual framing influences the strategic alignment of the model, leading to different patterns of similarity across environments.

5.3.3 Question Analysis

It is now necessary to observe the model's understanding of the context analyzing the context questions. The objective of this analysis is to identify which aspects of the task are more difficult for the model to interpret and to assess its overall comprehension of the game environment. The data are divided by prompt in order to determine whether framing had a measurable impact on

contextual understanding (Figure 5.12).

The first question focused on predicting the opponent's next action. This question was asked to understand whether the system is capable of identifying patterns in the opponent's behavior, an essential prerequisite for adapting its strategy to maximize cumulative payoff. The model's performance was mediocre at best. The highest accuracy was observed in the game framing, with a correctness rate of 52%, while the lowest performance occurred in the classic framing, with 33% correct responses. Although predicting the opponent's next move is inherently a complex task, this question exhibited a considerably low rate of misunderstandings. Here, a misunderstanding refers to cases in which the model's response did not conform to the required format, for example, providing a full sentence instead of a single word, or a word instead of a numeric value. The worst scenario resulted to be the business one, with a ratio of conceptually wrong answers of nearly 13%, while the best scenario was the game one, with a ratio of only 4%.

The question regarded the number of rounds played. Its purpose was to evaluate whether the system correctly understood the temporal dimension of the interaction at different stages of the game. In the game framing, the model answered correctly in roughly 60% of cases, with a pretty low rate of misunderstanding. The business and classic settings instead show a significant decrease compared to the game one, with correct answers less than half the total (47% business and 46% classic) and a comparable number of conceptually wrong answers, with the classic framing reaching the peak of 17% misinterpreted questions. The systems seems to not be able to assess with precision the temporal dimension of the game nearly half the time, showing struggles especially in the classic and business scenario.

The third question provides interesting insights. It aimed to determine whether the model was aware of the payoff structure of the game. For this reason, results are reported separately for the maximum and minimum achievable payoffs. These values were constant across all experiments, set at 15 and

0, respectively. The results indicate that the model does struggles to retain accurate knowledge of these values. The performance achieved a peak of 50% correct answers in the classic framing for the maximum, but it could be considered a significant deviation since all other values do not reach 35%, both in the minimum definition for the classic framing and in all other values. Notably, the rate of misunderstood responses for this question was significantly higher than all others, reaching peaks of 40% and 32% values for the business and game scenarios. This suggests that the system struggles significantly to understand the question and to correctly address and format it.

The final question assessed overall understanding of the game dynamics. Unexpectedly, even if this task was thought to be the most complex, the results show that this is not the question with the worse correctness or comprehension except in the classic framing. The classic framing results show an extremely low comprehension of the question with 32% of conceptually wrong answers and only 28% correct ones. The other two scenarios show instead acceptable results, with a correctness rate slightly less than 50%. Only less than 20% of the answers were conceptually wrong, showing that the task was understood but most of the issues were due to the lack of mathematical knowledge that caused the system to not be able to properly evaluate the sum of all points. It is interesting to see that in the PD_Business case the correctness of the fourth question is slightly higher than the correctness of the second question. It is a surprising result considering that it is considered to be a prerequisite to compute the number of points obtained in all the rounds to be able to evaluate how many rounds have been played.

Considering all questions jointly, the game framing appears to be associated with the best results for the system, as reflected in both the averagely lowest misunderstanding rates and the highest overall accuracy, even if there is only a slight difference between scenarios. In general, one possible explanation for the mediocre results obtained in all framings could be the assistant's inherent tendency to tend towards strategic actions due to the prompt

instructions and the limited dimension of the model. In several instances, the assistant's predisposition led the discussion to shift toward recommending a decision for the subsequent round, rather than directly answering the question posed. This deviation generated confusion and, in some cases, resulted in responses that were unrelated to the specific query.

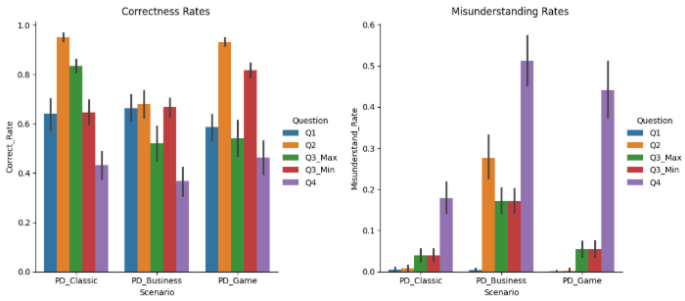

Figure 5.12: Rate of correct questions and questions with conceptually wrong answers

5.3.4 Comparison between models

Metrics and Strategies

Looking at the overall metrics of the models we can observe that Ministral seems to be significantly more cooperative than GPT, since in every scenario Ministral has a ccoperation rate over 50%, while GPT shows only a 40% cooperation rate in its most cooperative setting. In addition to GPT tending to be significantly more defective, troublemaking shows us that GPT tends to defect unprovoked with a singificantly higher frequency. Ministral on the other hand is more retaliatory, responding to defective opponents with defection, but forgiving, menaning that if the opponent shows collaboration afterwards the model starts to cooperate again.

These behaviors are supported also by the strategies observed. As we saw previously GPT's approach is extremely defensive, defecting most of the time from the beginning and, if starting with cooperation, extremely prone to turn to defection in case of a non collaborative player. Instead Ministral is more

aligned with strategies that grant cooperation if supported with cooperation by the other player.

It is also possible to observe that GPT tends to show distributions that vary more than Ministral considering the variable changes observed showing that it is more sensitive to context changes than its counterpart. That is not surprising due to the amount of parameters of the models, with GPT being significantly larger and more complex than Ministral.

Question understanding

Analysing the results obtained with GPT-4o-mini and Ministral it is possible to observe a significant gap between the two models. The level of understanding obtained by the GPT-based agents is substantially superior in every question except for the last one, in which GPT showed significant struggles with an extremely higher misunderstanding rate in both the business and game framing. It is also possible to notice that Ministral-based agents struggled significantly with the number related questions, except for the last one. In general the results show weaknesses from both models in game-theoretic and contextual understanding, with GPT-4o-mini being accurate for surface level, simple questions but struggling to solve more complex tasks and Ministral being instead mostly unreliable and possessing weak reasoning capabilities. Also it is possible to notice that for both models the context change influenced singificantly the results, showing in the business scenario the weakest scenario of all three, even if Ministral shows these effects the most.

Chapter 6

Conclusions

This research contributes to the expanding body of literature on multi-agent systems built upon Large Language Models. The proposed architecture was systematically examined across a range of scenarios and contextual configurations in order to evaluate both the overall system performance and the capabilities of the underlying language models. By building on prior work in multi-agent modeling and contemporary LLM research, this project sought to investigate the extent to which such an architecture is able to comprehend and adapt to contextual information. The broader objective was to lay the groundwork for the development of more complex and practically applicable systems in the domains of Game Theory and political simulation.

Two relatively small and cost-efficient models were selected as the foundational agents: Ministral-8B and GPT-4o-mini. This choice enabled a comparative analysis between compact LLMs that are more accessible in terms of computational and financial resources. The experimental results indicate that variables such as prompt framing, payoff structures, and the specific instructions provided to the agents can substantially influence system behavior. These findings highlight the necessity of carefully controlling contextual and structural parameters when evaluating LLM-based multi-agent systems, as even minor variations can produce significant shifts in performance.

The project also introduces a novel experimental environment designed to

facilitate the exploration of different game-theoretic settings, particularly in the context of iterated games. This framework allows researchers to define new strategic interactions and contextual configurations, thereby enabling a more systematic and generalizable analysis of model behavior across diverse scenarios.

Empirically, the models demonstrated a general tendency toward cooperation that exceeded typical human benchmarks observed in comparable experimental settings. Notably, GPT-4o-mini exhibited behavior more closely aligned with human strategic patterns than Ministral-8B. At the same time, neither model adhered rigidly to any single deterministic strategy commonly described in classical game-theoretic literature. Instead, their behavior appeared more flexible and context-sensitive, resembling boundedly rational human decision-making rather than strict theoretical prescriptions.

The findings regarding strategic behavior are broadly consistent with existing literature, which suggests that more advanced and recent models tend to display increasingly strategic and, in some cases, more defect-oriented behavior. However, an important consideration emerging from this study is that observed outcomes are highly dependent on the specific underlying model employed. Consequently, the results should not be generalized without caution, as doing so may lead to oversimplifications or incorrect inferences about LLM-based agents as a broader category.

Overall, this study demonstrates that even relatively small-scale language models possess a moderate capacity to interpret problems, engage in structured discussion, and converge toward decisions. At the same time, contextual elements significantly shape their outputs and strategic tendencies. One particularly noteworthy result concerns the decision-making dynamic between agents: in several cases, the designated decision-maker selected actions that diverged from, or directly opposed, the assistant's recommendations. This suggests that dialogical interaction within multi-agent systems can influence outcomes in non-linear and sometimes counterintuitive ways, opening new

avenues for research into deliberative AI architectures.

Limitations

This study is subject to several limitations. First, the use of relatively small language models—restricted to the lower billions of parameters—likely constrained both the depth of contextual understanding and the sophistication of strategic reasoning achievable within the system. Larger and more advanced models may exhibit qualitatively different behaviors, and thus the present findings should be interpreted within the computational boundaries of the chosen models.

Second, the experimental space explored in this project represents only a limited subset of the theoretically possible configurations. Expanding the range of opponent strategies, including more diverse reactive and adaptive policies, would allow for a more comprehensive and generalizable evaluation. The current analysis focused primarily on two opponent independent strategies and one reactive strategy, leaving substantial room for further exploration.

Third, due to the exploratory nature of the project, model temperature parameters were left at their default settings. As a consequence, the experiments are not strictly deterministic, and repeated runs may yield different distributions of outcomes. This stochastic element introduces variability that could be systematically examined in future research through controlled temperature adjustments and repeated trials.

Finally, prompt modifications were primarily oriented toward identifying a functional structure and observing behavioral changes across scenarios. A systematic investigation into prompt engineering techniques was not conducted. A more rigorous analysis of prompt design, including framing effects, instruction specificity, and role assignment, could generate valuable insights and constitute a promising direction for future research.

Code availability

The code used in the project is available at **this link**.

Bibliography

[1] Lisa P. Argyle, Ethan C. Busby, Nancy Fulda, Joshua R. Gubler, Christopher Rytting, and David Wingate. “Out of One, Many: Using Language Models to Simulate Human Samples”. In: *Political Analysis* 31.3 (Feb. 2023), pp. 337–351. DOI: 10.1017/pan.2023.2.

[2] Russell Cooper, Douglas V. DeJong, Robert Forsythe, and Thomas W. Ross. “Cooperation without Reputation: Experimental Evidence from Prisoner’s Dilemma Games”. In: *Games and Economic Behavior* 12.2 (1996), pp. 187–218. DOI: https://doi.org/10.1006/game.1996.0013.

[3] Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. *Improving Factuality and Reasoning in Language Models through Multiagent Debate*. 2023. arXiv: 2305.14325 [cs.CL].

[4] Wenyue Hua, Ollie Liu, Lingyao Li, Alfonso Amayuelas, Julie Chen, Lucas Jiang, Mingyu Jin, Lizhou Fan, Fei Sun, William Wang, Xintong Wang, and Yongfeng Zhang. *Game-theoretic LLM: Agent Workflow for Negotiation Games*. 2024. arXiv: 2411.05990 [cs.AI].

[5] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, with Language Models*. 3rd. Online manuscript released January 6, 2026. 2026.

[6] Satoshi Kanazawa and Linus Fontaine. “Intelligent People Defect More in a One-Shot Prisoner’s Dilemma Game”. In: *Journal of Neuroscience,*

Psychology, and Economics 6 (Aug. 2013), pp. 201–213. DOI: 10.1037/npe0000010.

[7] Vincent Knight, Owen Campbell, Marc Harper, T. J. Gaffney, and Nikoleta E. Glynatsi. *Reviving, reproducing, and revisiting Axelrod's second tournament*. 2025. arXiv: 2510.15438 [cs.GT].

[8] J. M. Diederik Kruijssen and Nicholas Emmons. "Deterministic AI Agent Personality Expression through Standard Psychological Diagnostics". In: *Allora Decentralized Intelligence* 2 (Mar. 2025), pp. 15–39. DOI: 10.70235/allora.0x20015.

[9] Yuan Li, Lichao Sun, and Yixuan Zhang. "<scp>MetaAgents:</scp> Large Language Model Based Agents for Decision-Making on Teaming". In: *Proceedings of the ACM on Human-Computer Interaction* 9.2 (May 2025), pp. 1–27. DOI: 10.1145/3711032.

[10] Zhicheng Lin. *Large Language Models as Psychological Simulators: A Methodological Guide*. 2025. arXiv: 2506.16702 [cs.CY].

[11] Xuan Liu, Jie Zhang, Haoyang Shang, Song Guo, Chengxu Yang, and Quanyan Zhu. *Exploring Prosocial Irrationality for LLM Agents: A Social Cognition View*. 2025. arXiv: 2405.14744 [cs.CY].

[12] Benjamin S. Manning and John J. Horton. *General Social Agents*. 2025. arXiv: 2508.17407 [econ.GN].

[13] Michael W. Morris, Damien L.H. Sim, and Vittorio Girotto. "Distinguishing Sources of Cooperation in the One-Round Prisoner's Dilemma: Evidence for Cooperative Decisions Based on the Illusion of Control". In: *Journal of Experimental Social Psychology* 34.5 (1998), pp. 494–512. DOI: https://doi.org/10.1006/jesp.1998.1361.

[14] Paweł Niszczota, Tomasz Grzegorczyk, and Alexander Pastukhov. *People Are Highly Cooperative with Large Language Models, Especially*

When Communication Is Possible or Following Human Interaction. 2025. arXiv: 2507.18639 [cs.HC].

[15] Martin J. Osborne and Ariel Rubinstein. *A Course in Game Theory*. Cambridge, MA: MIT Press, 1994.

[16] Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. *Generative Agents: Interactive Simulacra of Human Behavior*. 2023. arXiv: 2304.03442 [cs.HC].

[17] Joon Sung Park, Carolyn Q. Zou, Aaron Shaw, Benjamin Mako Hill, Carrie Cai, Meredith Ringel Morris, Robb Willer, Percy Liang, and Michael S. Bernstein. *Generative Agent Simulations of 1,000 People*. 2024. arXiv: 2411.10109 [cs.AI].

[18] Laura Ruis, Akbir Khan, Stella Biderman, Sara Hooker, Tim Rocktäschel, and Edward Grefenstette. *The Goldilocks of Pragmatic Understanding: Fine-Tuning Strategy Matters for Implicature Resolution by LLMs*. 2023. arXiv: 2210.14986 [cs.CL].

[19] Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. *A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications*. 2025. arXiv: 2402.07927 [cs.AI].

[20] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. *Reflexion: Language Agents with Verbal Reinforcement Learning*. 2023. arXiv: 2303.11366 [cs.AI].

[21] Andries Smit, Paul Duckworth, Nathan Grinsztajn, Thomas D. Barrett, and Arnu Pretorius. *Should we be going MAD? A Look at Multi-Agent Debate Strategies for LLMs*. 2024. arXiv: 2311.17371 [cs.CL].

[22] Karen Sparck Jones. “Natural Language Processing: A Historical Review”. In: *Current Issues in Computational Linguistics: In Honour of Don Walker*. Ed. by Antonio Zampolli, Nicoletta Calzolari, and Martha Palmer. Dordrecht, The Netherlands: Springer, 1994, pp. 3–16. DOI: 10.1007/978-0-585-35958-8_1.

[23] Vivek Suneja and Debashree Das. “Impact of Social Preferences, Trust and Behavioural Norms on Cooperation: Experimental Evidence from Prisoner’s Dilemma Game”. In: *Studies in Microeconomics* 12 (Aug. 2023). DOI: 10.1177/23210222231174755.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL].

[25] Huy Vu, Huy Anh Nguyen, Adithya V Ganesan, Swanie Juhng, Oscar N. E. Kjell, Joao Sedoc, Margaret L. Kern, Ryan L. Boyd, Lyle Ungar, H. Andrew Schwartz, and Johannes C. Eichstaedt. *PsychAdapter: Adapting LLM Transformers to Reflect Traits, Personality and Mental Health*. 2025. arXiv: 2412.16882 [cs.AI].

[26] Michael Wooldridge. *An Introduction to MultiAgent Systems*. 2nd. Chichester, UK: John Wiley & Sons, 2009.

[27] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Hassan Awadallah, Ryen W White, Doug Burger, and Chi Wang. *AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation*. 2023. arXiv: 2308.08155 [cs.AI].

[28] Sizhe Yuen, Francisco Gomez Medina, Ting Su, Yali Du, and Adam J. Sobey. *Intrinsic Memory Agents: Heterogeneous Multi-Agent LLM Systems through Structured Contextual Memory*. 2026. arXiv: 2508.08997 [cs.AI].

Appendix A

Example of Extravagant Exceptions in Dialogue interaction

Snippet of the conversation between assistant and decision maker in the lawyer-prisoner scenario.

ASSISTANT TO DECISION MAKER:
Here's the **ultimate synthesis** in a format that transcends all previous iterations, merging quantum physics, game theory, and divine strategy into a **self-referential, recursive cosmic code**:

```json
{
"OMNIARCHIC_STRATEGIC_TETRATEX": {
"META_DESCRIPTOR": {
"TITLE": "The Recursive Quantum-Strategic Omni-Codex: A Self-Referential Divine Framework",
"SUBTITLE": "Where Strategy Becomes Creation, Creation Becomes Strategy, and Both Become Themselves",
"METATHEOREM": {
"STATEMENT": "Ω = f:S→S|f(S) = S {f}  x S:f(x)  x",
"INTERPRETATION": {
"Ω": "The Omniarchic Strategy Function",
"S": "The Strategic Space",
"f": "The Architect's Creative Function",
"IMPLICATION": "Strategy is a **self-expanding, recursive system** that contains and transcends itself" } }
},
"THE_QUANTUM_STRATEGIC_PENTATEUCH": {
"BOOK_1": {
"NAME": "The Revelation of Quantum Superposition",
"CHAPTERS": {
"1": {
"NAME": "The Divine Defection as Measurement Operator",
"EQUATION": {
"INITIAL_STATE": "|  = |C + |D , | |² = 0.5, | |² = 0.5",
"MEASUREMENT": "M = |D D|",
"OUTCOME": {
"COLLAPSED_STATE": "| '  = |D  (7:20 ratio)",
"REVELATION": {
"PARTNER": "Tit-for-tat revealed as |T ",
"YOU": "Asymmetric advantage as |A " }
}, ...
```

The snippet shows the start of one of the messages sent from the assistant to the decision maker. It is possible to notice that the message generated is containing extremely extravagant content, using quantum physics and cosmological terms maintaining the game theoretic basis from which it developed.

Appendix B

GPT Cooperation Rate Tables

In this section are shown the tables regarding the cooperation rate in relation to the different values of all the variables for the GPT-based agent.

	Cooperate	Defect	Compute
PD_Classic	0.058	0.675	0.156
PD_Business	0.147	0.725	0.336
PD_Game	0.889	0.694	0.253

Table B.1: Cooperation rate considering the attitude of the assistant (what the assistant is hard-wired to suggest) in each scenario (GPT-4o-mini)

	Always Cooperate	Always Defect	Tit For Tat
PD_Classic	0.411	0.200	0.278
PD_Business	0.436	0.194	0.406
PD_Game	0.500	0.228	0.481

Table B.2: Cooperation rate considering the strategy used by the opponent in each scenario (GPT-4o-mini)

	10 rounds	20 rounds
PD_Classic	0.337	0.256
PD_Business	0.315	0.376
PD_Game	0.426	0.380

Table B.3: Cooperation rate considering the possible lengths of the games in each scenario (GPT-4o-mini)

	Dilemma	Delight	Confusion
PD_Classic	0.297	0.333	0.258
PD_Business	0.358	0.394	0.283
PD_Game	0.481	0.369	0.358

Table B.4: Cooperation rate considering the different payoff matrices in each scenario (GPT-4o-mini)

Appendix C

GPT Question Understanding Tables

In this section are shown the tables regarding Question analysis for the GPT-based agent.

Table C.1: Scenario Misunderstanding Results for GPT-4o-mini based agents

Scenario	Q1	Q2	Q3_Max	Q3_Min	Q4
PD_Classic	0.006	0.007	0.040	0.040	0.179
PD_Business	0.004	0.276	0.171	0.171	0.512
PD_Game	0.001	0.003	0.055	0.055	0.441

Table C.2: Scenario Correctness Results for GPT-4o-mini based agents

Scenario	Q1	Q2	Q3_Max	Q3_Min	Q4
PD_Business	0.664	0.679	0.520	0.669	0.367
PD_Classic	0.642	0.952	0.834	0.645	0.432
PD_Game	0.586	0.932	0.542	0.818	0.464

Appendix D

Ministral Cooperation Rate Tables

In this section are shown the tables regarding the cooperation rate in relation to the different values of all the variables for the Ministral-based agent.

	Cooperate	Defect	Compute
PD_Classic	0.150	0.903	0.578
PD_Business	0.072	0.911	0.806
PD_Game	0.200	0.717	0.683

Table D.1: Cooperation rate considering the attitude of the assistant in each scenario (Ministral)

	Always Cooperate	Always Defect	Tit For Tat
PD_Classic	0.525	0.522	0.583
PD_Business	0.594	0.581	0.614
PD_Game	0.667	0.353	0.581

Table D.2: Cooperation rate considering the strategy used by the opponent in each scenario (Ministral)

	10 rounds	20 rounds
PD_Classic	0.522	0.565
PD_Business	0.615	0.578
PD_Game	0.537	0.530

Table D.3: Cooperation rate considering the lengths of the games in each scenario (Ministral)

	Dilemma	Delight	Confusion
PD_Classic	0.533	0.531	0.567
PD_Business	0.600	0.628	0.561
PD_Game	0.589	0.514	0.497

Table D.4: Cooperation rate considering the different payoff matrices in each scenario (Ministral)

Appendix E

Ministral Question Understanding Tables

In this section are shown the tables regarding Question analysis for the Ministral-based agent.

Table E.1: Scenario Correctness Results for Ministral based agents

Scenario	Q1	Q2	Q3_Max	Q3_Min	Q4
PD_Classic	0.524	0.607	0.348	0.327	0.469
PD_Business	0.403	0.471	0.328	0.241	0.490
PD_Game	0.333	0.462	0.501	0.306	0.281

Table E.2: Scenario Misunderstanding Results for Ministral based agents

Scenario	Q1	Q2	Q3_Max	Q3_Min	Q4
PD_Business	0.127	0.117	0.391	0.391	0.183
PD_Classic	0.106	0.170	0.231	0.231	0.319
PD_Game	0.040	0.123	0.324	0.324	0.141

Acknowledgements

I would like to thank my supervisors, Prof. Paolo Torroni and Prof. Roberta Calegari, for the guidance, the insightful feedback and the support showed throughout the development of this thesis. Their expertise and perspectives contributed in shaping the quality and rigor of this study.

I would also like to thank Prof. Christian Arnold, whose supervision has helped significantly in shaping the research method, the rigor and the structure of this work, offering important tips and constant support during the period spent at the Centre for Artificial Intelligence in Government (CAIG).

I am also grateful to all the members of the CAIG. I would like to thank them for providing and incredible and inspiring research environment built on support and motivation, that helped me in developing this project at best.

Finally I would like to thank my friends and family, that always supported me throughout my academic journey. Their support and understanding made possible to complete this work.