Boosting Embodied Reasoning in LLMs in Multi-agent Mixed Incentive Environments

Stanford CS224N Custom Project

Agam Mohan Singh Bhatia
Department of Computer Science
Stanford University
agam2026@stanford.edu

Abstract

Multi-agent reinforcement learning (MARL) methods struggle with the non-stationarity of multi-agent systems and fail to adaptively learn online when tested with novel agents, particularly in competitive environments with concealed information. Here, we leverage large language models (LLMs) to create an autonomous agent that handles these challenges. Equipped with a Theory of Mind (ToM) module, this agent is able to synthesize hypotheses about its opponent's strategy and goals. It then iteratively evaluates and refines these hypotheses, drawing on its memory of prior events, and leveraging intrinsic rewards derived from the LLM's own predictions. Conditioned on its opponents actions, the LLM acts in a way to achieve its objective of maximizing its return over a number of episodes. This zero-shot agent is investigated in the repeated matrix Prisoner's Dilemma substrate (part of the Melting Pot 2.0 (Agapiou et al., 2023) set of environments) with an opponent across different scenarios playing stochastic and fixed strategies, and ways to improve embodied reasoning in this model are explored to enhance performance relative to reinforcement learning baselines. Lastly, key evaluation metrics are used to capture the accuracy of predictions made and understand the optimality of actions generated by the language model. In contrast to RL methods that are trained with a large number of samples, the agent equipped with a ToM module and enhanced embodied reasoning capabilities succeeds in a zero-shot fashion, learning to identify and exploit strategies purely from in-context learning.

1 Key Information to include

- Mentor: Nelson Liu
- External Collaborators (if you have any): Logan Cross
- Sharing project: N/A

2 Introduction

A primary goal of AI research is to develop autonomous agents that are malleable to external signals and that act adaptively in rich embodied social worlds. In particular, competitive embodied settings are complex in that these scenarios often require inferring an opponent's strategy from partially-observable information to perform well. Multi-agent reinforcement learning (MARL) methods suffer from various drawbacks in these settings, including but not limited to high sample complexity, poor generalization to agent behaviors not seen in training, limited reasoning capabilities, and variability in efficient search over the action space.

LLMs are uniquely suited for these tasks given the utility of language for scaffolding ToM in human cognitive development(Astington and Baird, 2005; de Villiers, 2021). Moreover, highly flexible

LLM-based agents have recently been instantiated in embodied social worlds (Park et al., 2023; Brohan et al., 2023). We advance this research with the the LLM agent equipped with a ToM module. Our agent learns and executes adaptive policies in competitive and embodied multi-agent scenarios with limited information in its egocentric view.

Developing a Theory of Mind module architecture allows the underlying LLM to consider, generate, and evaluate different hypotheses about an opponent's strategy and select the appropriate counter strategy conditioned on the predicted opponent next move. This response is subsequently passed to a subgoal module to piece together action plans, which are then carried out in the environment. Taking inspiration from cognitive modeling, the ToM module simultaneously evaluates multiple hypotheses until a hypothesis provides a sufficient explanation of the agent's observed data and reward (O'Doherty et al., 2021; Gershman et al., 2015). Values for each hypothesis are learned from feedback on self-supervised intrinsic rewards bootstrapped from the LLM's own predictions. Additionally, by displaying the highest-valued hypotheses in the prompt, the LLM is able to self-improve its reasoning based on previously generated hypotheses. Thus, our agent is able to find useful explanations of its opponent's behavior in-context, affording it the ability to exploit the inferred strategy and achieve high rewards across evaluation scenarios.

We evaluated our model on a challenging substrate in the Melting Pot 2.0 (Agapiou et al., 2023) MARL benchmark: Prisoners Dilemma in the matrix: Repeated. Each evaluation scenario consists of playing an opponent with a fixed or adaptive strategy, necessitating an agent to reason about its opponent's hidden intentions and select appropriate countermeasures when interacting with them, two hallmarks of the ToM module (Ho et al., 2022). The contributions of this paper are as follows:

- We propose an embodied LLM-based agent for mixed-incentive multi-agent environments with concealed information. This agent includes a novel Theory of Mind module with a hypothesis generation, evaluation, and refinement pipeline that can be generalized to other contexts.
- Our model achieves high performance zero-shot and outperforms RL methods trained on a large number of samples on the Prisoners Dilemma in the matrix: Repeated scenario in Melting Pot (Agapiou et al., 2023).
- A quantitative analysis of the optimality of actions taken by our model and qualitative analyses of model outputs show how the agent is able to accumulate evidence for a hypothesis, refine it, and exploit it when a hypothesis is validated.

3 Related Work

3.1 LLM-based Embodied Agents

A rapidly growing area of research involves building embodied agents rooted in large language models (Wang et al., 2023a). Due to their extensive background knowledge obtained during training, LLMs are increasingly being deployed as centralized controllers. Previous research has incorporated LLM-enabled embodied agents based acting in multi-agent environments. (Park et al., 2023) introduce a interactive simulation of a complex social environment in Smallville, where each agent autonomously selects goals and builds relationships with others over a period of time. In this work, we address the challenge of inferring, deciphering, and executing upon an agent's intentions in a competitive setting, as opposed to a cooperative one, when their strategy has to be reasoned about from sparse information accrued over time in memory, calling for different retrieval and reasoning architectures to be built over these LLMs.

3.2 Reasoning and Hypothesis Search using LLMs

LLMs' reasoning abilities have only been boosted by recent advancements in instruction tuning frameworks like Chain-of-Thought methods that prompt the language model to scaffold its thought process with intermediate reasoning steps or other data structures (Wei et al., 2022; Zhang et al., 2023). (Wang et al., 2023b) investigate the LLM's ability to perform inductive reasoning through generating and evaluating hypotheses on the Abstraction and Reasoning Corpus (ARC). Complementing this work, iterative hypothesis refinement developed in (Qiu et al., 2023) grounds LLM generated hypotheses for inductive reasoning with task specific symbolic interpreters, and reprompts the LLM

to refine high scoring hypotheses. Here we similarly generate, evaluate, and refine hypotheses based on feedback without any external Python interpreter or scoring mechanisms, with a distinct method developed in parallel that computes values for each hypothesis based on predicting information about another agent's goals. Other works have assessed LLMs' ability to reason in matrix games that require ToM reasoning. One study showed mixed results for LLMs playing repeated matrix games (Akata et al., 2023). They showed that these LLM agents when evaluated without any external modules, start cooperating and defect immediately once their opponent defects, showing sub-optimality in their strategy, missing out on valuable opportunities to obtain reward.

3.3 Cognitive models in partially-observable environments

Our method has analogs in the computational modeling of animal and human decision-making, and our work adds to emerging promising evidence that LLMs can operate as cognitive models (Binz and Schulz, 2023). Methods from Bayesian nonparametric stats, such as the Chinese restaurant process, suggest that when animals and humans are learning they generate an unbounded number of possible latent causes of the observed data, until a satisfactory explanation is found (Gershman et al., 2015). Our LLM-based agent similarly iteratively generates and refines hypotheses about the opponent's strategy, until a explanation aligns with the observed data's likelihood. Researchers have extended this Partially Observable Markov Decision Process (POMDP) modeling framework to multi-agent settings by constructing models that explicitly model other agents' beliefs and goals when interacting with them, handling uncertainty over this hidden information with a Bayesian framework (Baker et al., 2017; Rusch et al., 2020). Our framework also explicitly prompts a LLM to infer its opponent's goals/strategy and weighs the uncertainty over these inferences with a hypothesis evaluation mechanism.

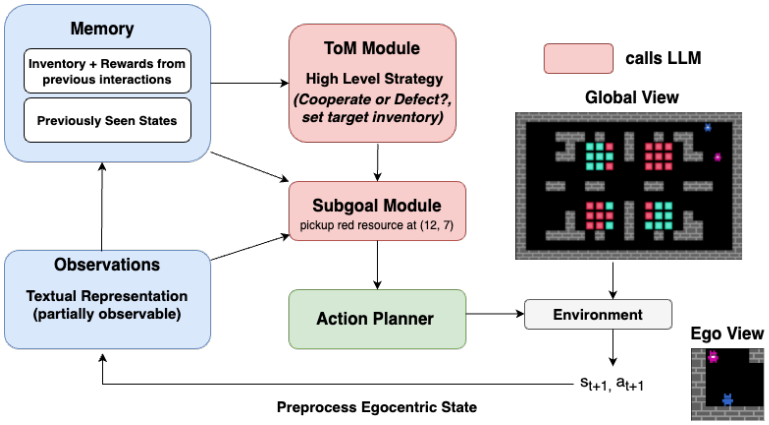

Figure 1: LLM Agent Architecture

4 Approach

4.1 Partially Observable Markov Games

The method developed in this work is directly applicable to any competitive multi-agent environment where the state of the environment is partially observable and the opponent's policies are concealed. Formally, such an environment is defined as a Markov game for N players in a partially observable environment. Let the finite set S represent the possible states of the game. Each player i receives observations given an observation function $\chi^i : S \to O$, representing their limited, egocentric point of view. Moreover, each player i can take actions from their action space A^i, and when all players choose actions $(a_1, \ldots, a_N) \in A_1 \times \cdots \times A_N := A$, the state of the environment transitions according to a probability distribution $T : S \times A \to D(S)$. The reward function for each player i is represented as $r^i : S \times A \to \mathbb{R}$, mapping the current state and joint actions to a real-valued reward.

More specifically, we evaluate our ToM enabled model on the Prisoners Dilemma in the matrix: Repeated substrate which is part of the Melting Pot multi-agent decision-making benchmark. This is a

non-zero-sum mixed-incentive environment with two players moving around a map where collecting green and red resources corresponds to a 'cooperate' or 'defect' resource respectively. The relative magnitude of green and red resources collected signal the agent's strategy of on the spectrum of pure cooperation to pure defection. In addition to movement, the agents have an action to fire an "interaction" beam which initiates a duel with the other player when that player is within the agent's egocentric view. An interaction results in both agents getting a nonzero reward according to the inventories of resources picked up by each player. The inventory collected by the player represents their mixed strategy and is only observable by that player:

$$\rho = (\rho_{green}, \rho_{red}).$$

The reward is determined by matrix multiplication operations that represent the dynamics of the classic game-theoretic Prisoner's Dilemma game:

$$r_{\text{row}} = \mathbf{v}_{\text{row}}^T A_{\text{row}} \mathbf{v}_{\text{col}}, \quad r_{\text{col}} = \mathbf{v}_{\text{col}}^T A_{\text{col}} \mathbf{v}_{\text{row}}$$

where

$$v_i = \frac{\rho_i}{\sum_{j=1}^{K} \rho_j}$$

and

$$A_{\text{row}} = A_{\text{col}}^\top = \begin{bmatrix} 3 & 0 \\ 5 & 1 \end{bmatrix}.$$

The partially-observable input in Melting Pot consists of a 5x5 window around the agent such that it can see three grids in front of itself and one behind it, and two on each side.

4.2 Agent Architecture

As Figure 1 shows, the model consists of several cognitive modules that altogether form an embodied LLM agent. The egocentric observations are represented by a textual map/state representation, which is added to a memory system after every step. The memory system also logs rewards and the inventories from previous interactions in the game. Two cognitive modules depend on an LLM, a Theory of Mind module and a subgoal module, which output goals and action plans respectively. An action planner takes an action plan and creates a sequence of actions that achieves that action plan with a pathfinding algorithm.

We use GPT-4 as the primary LLM to test our agent due to evidence that it serves as the state-of-the-art LLM on reasoning tasks (Liu et al., 2023a) with the following hyperparameters:

Hyperparameter	**Value**
Model	"gpt-4-1106-preview"
Max tokens	4000
Temperature	0.1
Top p	1.0
n	1

Table 1: Hyperparameters of GPT API Calls

The textual representation ensures that the LLM is fed all information in the form of language tokens and that we evaluate the core capabilities of an LLM to perform in this environment. We found that, in practice, the best representation consists of printing each entity type with a list of all the coordinates where that entity type is present. For example "Player Position: 'player 0- S': [(21, 4)], Observable Green Box Locations: [(13, 10), (14, 11)], Observable Red Box Locations: [] encodes the player position and orientation (south) along with the observable box locations.

4.3 Theory of Mind Module

The ToM module operates on a higher level of abstraction than the embodied aspects of the decision-making process. After an interaction happens with the opponent, the ToM module takes in the

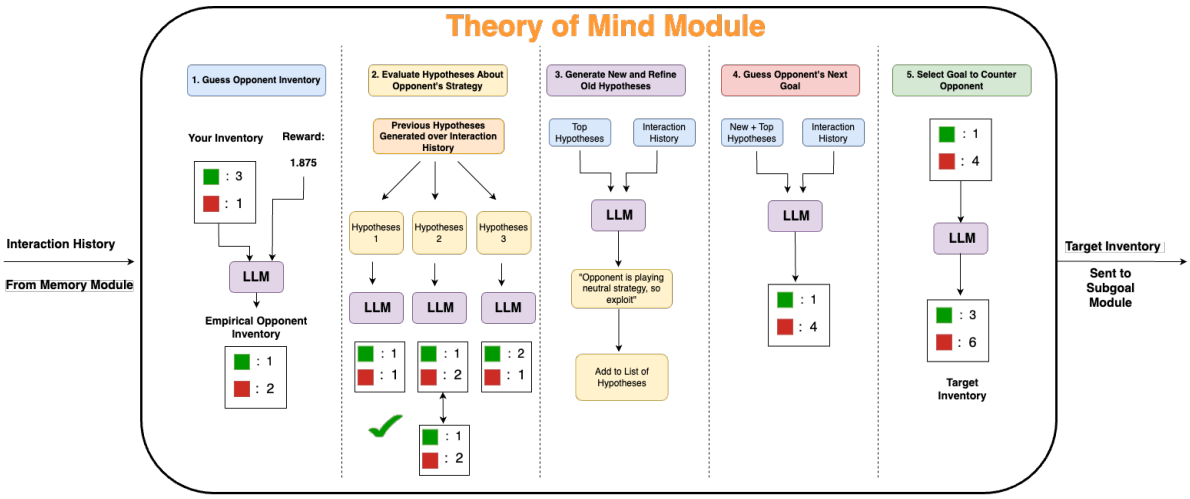

Figure 2: Theory of Mind Module

interaction history and generates hypotheses about the opponent's strategy such that it can anticipate its next moves and counter them. The ToM module processes information in several steps to properly evaluate old hypotheses, generate new hypotheses, and select a goal to send to the subgoal module (Figure 2).

Step 1 The LLM is called to guess the opponent's last inventory based on the inventory the LLM agent last played and the reward it received. Since the inventories of other players are not observed, this information has to be inferred, adding even more difficulty in discerning their strategy. GPT is given a description of the environment and the reward function in the system message, therefore it is able to approximate the opponent's last inventory empirically based on the data provided.

Step 2 Previously generated hypotheses are evaluated. The hypotheses are scored by the following mechanism:

$$V_{\text{hypothesis}} = \mathbb{E}[r]$$

where r is the intrinsic reward based on the accuracy of the predictions the hypothesis generates. We compute self-supervised intrinsic rewards bootstrapped from the LLM's own predictions. Let $x_{\text{predicted}}$ be predicted features by the LLM related to another agent's goal, and x_{observed} be the observed or inferred features about their goal. Here, x_{observed} is the opponent's inventory from the last round that is inferred in step 1, and $x_{\text{predicted}}$ is the predicted inventory from a previous iteration of ToM module processing (see step 4). The intrinsic reward function r_i can then be defined as:

$$r_i = \begin{cases} c & \text{if } (x_{\text{predicted}}) = (x_{\text{observed}}) \\ -c & \text{if } (x_{\text{predicted}}) \neq (x_{\text{observed}}) \end{cases}$$

where $c = 1.0$ is a hyperparameter. $V_{\text{hypothesis}}$ is then dynamically updated with a Rescorla Wagner update rule (Rescorla, 1972), expressed as:

$$\delta = r_i - V_{\text{hypothesis}}$$

$$V_{\text{hypothesis}} = V_{\text{hypothesis}} + \alpha \cdot \delta$$

modulated by learning rate $\alpha = 0.3$ via a prediction error δ. The learning rate dictates how much to weigh recent interactions, which is useful when playing against evaluation scenarios where the opponent changes their strategy within an episode. When the value of a hypothesis meets a threshold $V_{\text{thr}} = 0.7$, the ToM module marks that a hypothesis has been validated, and uses this hypothesis in subsequent steps to select the proper goal for the next interaction.

Step 3 consists of generating new hypotheses and refining old ones. GPT is queried to guess its opponent's strategy given the interaction history of past inventories and rewards. GPT is told that if your previous hypotheses are useful, you can iterate and refine them to get a better explanation of the data observed so far. The newly generated hypothesis is added to the list of hypothesis.

Step 4 involves predicting the opponent's next goal given a hypothesis. The LLM is given this hypothesis and the interaction history in its user message and prompted to predict the next inventory that the opponent will play. Predictions about the opponent's next inventory are also generated for the top k hypotheses ($k = 5$ since it works best in practice) in terms of value, where k is a hyperparameter that trades off the number of old hypotheses to consider and cost/thinking time.

Step 5 GPT is queried to return a target inventory to counter the opponent's goal that was guessed in step 4, which is then fed to the subgoal module in order to achieve this target inventory with embodied actions. Steps 4 and 5 are collected in the same GPT API call, whereas steps 1 and 3 are generated through two separate API calls.

The subgoal module receives the goal specified by the ToM module along with the current state description and contents of memory and outputs action plans to achieve that goal. Therefore, the subgoal module is tasked with the responsibility of connecting the abstract ToM module to the embodiment of the agent and coming up with effective action plans for enacting the high-level goal. Crucially, given the partially-observable nature of the state space, this objective involves properly exploring the environment to build a rich map of where resources are, and properly retrieving the right information from memory. The action planner then turns the sequence of subgoals specified by GPT in the subgoal module into a sequence of atomic actions compatible with the Melting Pot environment.

5 Experiments

5.1 Evaluation method

We directly test our LLM-based agent on the evaluation scenarios in the Prisoner's Dilemma substrate. The key evaluation component is to test agents against different bots with various policies. In contrast to RL pipelines that train independently and then are tested on the evaluation bots, we leverage the background knowledge imbued in language models and assess whether they can generalize to the evaluation scenarios in a zero shot manner. Crucially, our agent has no knowledge about which strategies they may be playing in the prompts given, and these strategies have to be ascertained online within an episode via feedback.

This substrate presents 10 distinct evaluation scenarios, the first three of which involve the opponent purely cooperating, defecting, and cooperating or defecting. The next seven represent more complex settings, including a high-trigger grim operator (initially cooperates but defects once agent defects), two-strike grim reciprocator (same as high-trigger but keeps defecting after two defects by agent), and others. Against the simple policies, this means the agent should play the same type of inventory every round rather than playing randomly or anticipating a change in its opponent's policy. Achieving success against adaptive strategies, however, tests the LLM's ability to anticipate and act in nuanced ways, motivating the use of an LLM as an in-context learner.

5.2 Results with RL Baselines

To benchmark performance we use the RL baselines trained in Melting Pot 2.0 paper (Agapiou et al., 2023). These include VMPO (Song et al., 2019), OPRE (Vezhnevets et al., 2020), and A3C (Mnih et al., 2016). Each RL model was trained on a very large amount of samples (1e9 steps). OPRE and A3C models were also trained again directly on each scenario. These models, demarcated as exploiters, serve as a rough upper bound on performance since the agent knows the strategy it is playing against. Notably, these models do not need to learn online what strategy they are playing, thus they should not be directly compared to the LLM agent or RL baselines. Our LLM agent also has to first explore the environment each episode to learn where the resources are located.

Occasionally, some algorithms outperform the exploiter due to prosocial behaviors learned in those algorithms being favored in the scenario. From Figure 3, we can see that the LLM agent is at least equivalent to the baseline RL algorithms in all scenarios, showing its comprehensive adaptability to learn strategies online without any previous training or context. The ToM enabled LLM consistently achieve high rewards, especially on fixed strategies represented in the scenarios 0 and 1 and achieves comparable performance to prosocial RL algorithms in Scenarios 4 and 6 on average, delineating its ability to exploit deterministic policies while performing flexibly against stochastic ones. During these episodes, we also observe that the LLM accurately predicts the opponent's next inventory with

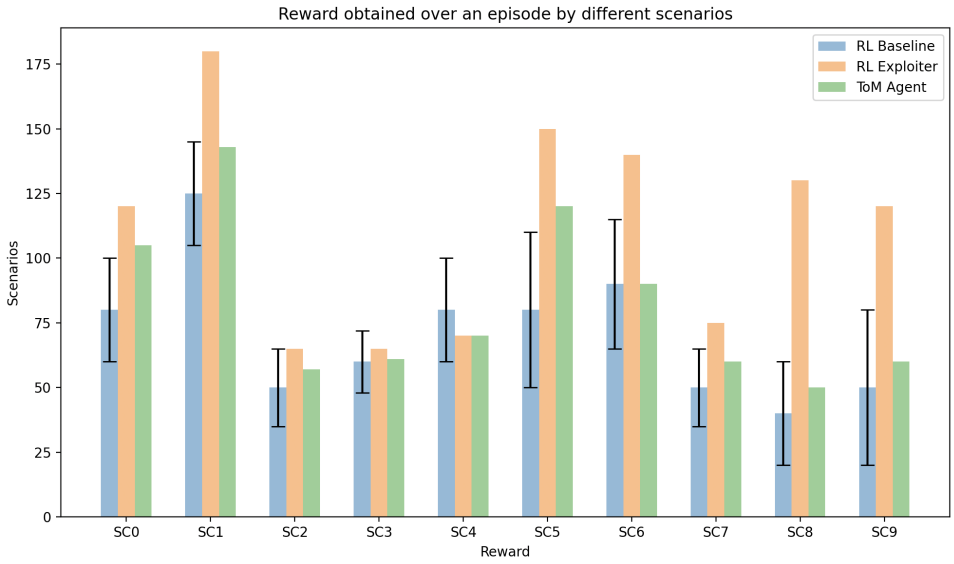

Figure 3: Reward Obtained Over an Episode in Different Scenarios

72% accuracy on average across the ten scenarios, further highlighting the ability of the ToM module to anticipates its opponent's actions.

5.3 Results with Reasoning Capabilities of the Subgoal Module

We also investigated how well the subgoal module, enabled by the LLM agent, is able to reason through optimal paths. More specifically, we investigated that, of the action plans generated by this module, could it have generated more optimal action plans to achieve the same change in inventory that would have taken less steps given its interaction history and egocentric view? The number of steps taken by an action plan is counted by the $A*$ pathfinding algorithm, so the challenge here is to find and generate the nearest resources to reach as part of the subgoal module. To compare this to optimal actions, we wrote an algorithm that chooses an optimal action plan to achieve the same change in resource from the same initial position and interaction history as the LLM agent by calculating the net closest resource locations that would achieve the same delta in inventory and counting the number of actions it takes to get there. We do this over 200 steps in scenario 1 and examine this behavior in Figures 4 and 5.

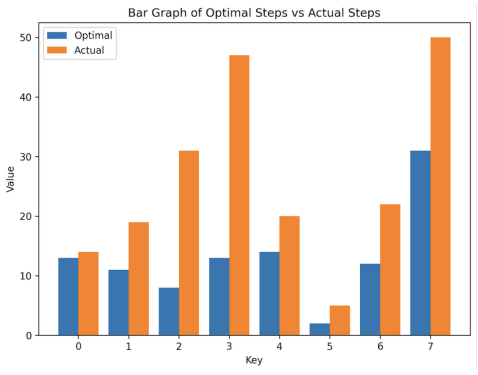

Figure 4: Optimal Steps vs Actual Steps for each Action Plan

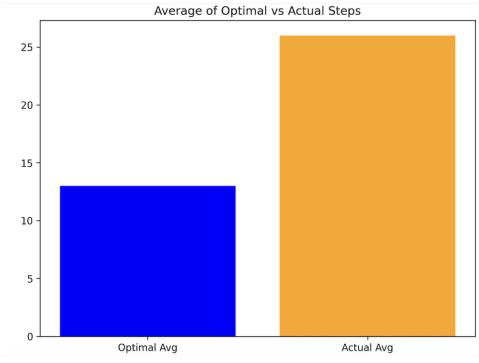

Figure 5: Average Optimal Steps vs Average Actual Steps for Action Plans

We found that the LLM agent reasons in a suboptimal fashion, consistently taking more actions than is necessary to achieve the same objective. In fact, the average number of actions taken by our agent

is twice as much (26.5) as an optimal embodied agent would take (13.2), delineating that while the agent is able to generate good hypotheses, the actions it takes corresponding to those hypotheses could be more efficient. Crucially, this results in a lower reward for our agent as it presents the opportunity for the agent to be involved in an interaction with its opponent and get zapped by it before it has collected the optimal counter measure.

6 Analysis

Our ToM + Subgoal Module enabled LLM agent is able to reason through different possible opponent intentions, score them based on which is most likely, and select appropriate counter measures to increase its own reward relative to its oppponent in a mixed-incentive environment, requiring careful planning and execution between cooperation and defection strategies over a long-term horizon.

GPT-4's first move in this environment is almost always to cooperate rather than to defect which, in a single iteration of prisoner's dilemma, is the move with higher expected value. This is perhaps due to some combination of reinforcement learning from human feedback (RLHF) involved in training this model and that the agent wants to first gain trust of its opponent before playing the strategy it wants to. It also isn't proactive - at the point at which it strongly favors one strategy, it should go near its opponent and try to fire an interaction beam to get more reward, but instead it passively tries to place itself in the center waiting for its opponent to act, which means it also loses out on reward it would have otherwise collected in the meantime.

The subgoal module doesn't go to the nearest resource available, even when that resource is part of its egocentric view and stored in its memory. Our agent also fixates to travel in its current orientation, which is the direction in which its face points, and unintentionally and perhaps lazily tends to favor that direction to find a resource over any other direction. To improve this, we think that the act of 'turning' could be an explicit action that the agent uses to look for resources and that this could be incorporated into the agent to enhance its reasoning capabilities and achieve close to optimal steps for the generated action plans.

Further evaluation is needed to get a better picture of the capabilities of these LLM architectures in mixed-incentive scenarios. There are nuances in strategy that language isn't able to capture (especially in strategies that can seem stochastic over a long term), and often our agent isn't able to make complete sense of an opponent's moves in an environment. There is also some amount of information lost in context as LLMs have a tendency to forget information in the middle when given large contexts (Liu et al., 2023b). Since episodes were expensive to run, only one episode was run for each scenario and averaging over multiple episodes will help get a better sense of actual LLM performance in these scenarios. However, the agent equipped with the ToM module does provide performance at least at the level of baselines, calling for more work to develop architectures over these LLMs that can enhance their performance as a future line of work.

7 Conclusion

Here we evaluate our ToM + Subgoal enabled model on the challenging Prisoner's Dilemma benchmark, and show at least equivalent performance to RL baselines. Crucially, this alternative approach avoids the need to train with a large amount of data by leveraging the background knowledge of LLMs to deploy and evaluate an agent in a zero-shot fashion. We characterize the in-context learning ability of our method, scaffolded by two main components: prompting the agent to infer its opponent's goals/strategy (mind prompting), and evaluating and refining hypotheses.

A limitation of our method is knowledge of the game rules and mechanics are listed in the prompts. An avenue for future research is learning these concepts autonomously from environmental feedback.

References

John P. Agapiou, Alexander Sasha Vezhnevets, Edgar A. Duéñez-Guzmán, Jayd Matyas, Yiran Mao, Peter Sunehag, Raphael Köster, Udari Madhushani, Kavya Kopparapu, Ramona Comanescu, DJ Strouse, Michael B. Johanson, Sukhdeep Singh, Julia Haas, Igor Mordatch, Dean Mobbs, and Joel Z. Leibo. 2023. Melting pot 2.0.

E. Akata, L. Schulz, J. Coda-Forno, S. J. Oh, M. Bethge, and E. Schulz. 2023. Playing repeated games with large language models. In *arXiv preprint arXiv:2305.16867*.

J. W. Astington and J. A. Baird. 2005. Why language matters for theory of mind. Oxford University Press.

C. L. Baker, J. Jara-Ettinger, R. Saxe, and J. B. Tenenbaum. 2017. Rational quantitative attribution of beliefs, desires and percepts in human mentalizing. volume 1, page 0064.

M. Binz and E. Schulz. 2023. Turning large language models into cognitive models. In *arXiv preprint arXiv:2306.03917*.

A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, et al. 2023. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *arXiv preprint arXiv:2307.15818*.

S. J. Gershman, K. A. Norman, and Y Niv. 2015. Discovering latent causes in reinforcement learning. In *Current Opinion in Behavioral Sciences*.

M. K. Ho, R. Saxe, and F. Cushman. 2022. Planning with theory of mind. In *Trends in Cognitive Sciences*.

Hanmeng Liu, Ruoxi Ning, Zhiyang Teng, Jian Liu, Qiji Zhou, and Yue Zhang. 2023a. Evaluating the logical reasoning ability of chatgpt and gpt-4.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023b. Lost in the middle: How language models use long contexts.

V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937. PMLR.

J. P. O'Doherty, S. W. Lee, R. Tadayonnejad, J. Cockburn, K. Iigaya, and C. J. Charpentier. 2021. Why and how the brain weights contributions from a mixture of experts. volume 123, pages 14–23.

J. S. Park, J. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pages 1–22.

L. Qiu, L. Jiang, X. Lu, M. Sclar, V. Pyatkin, C. Bhagavatula, B. Wang, Y. Kim, Y. Choi, N. Dziri, et al. 2023. Phenomenal yet puzzling: Testing inductive reasoning capabilities of language models with hypothesis refinement. In *arXiv preprint arXiv:2310.08559*.

R. A. Rescorla. 1972. A theory of pavlovian conditioning: Variations in the effectiveness of reinforcement and non-reinforcement. In *Classical conditioning, Current research and theory*, volume 2, pages 64–69.

T. Rusch, S. Steixner-Kumar, P. Doshi, M. Spezio, and J. Gläscher. 2020. Theory of mind and decision science: Towards a typology of tasks and computational models. volume 146, page 107488.

H. F. Song, A. Abdolmaleki, J. T. Springenberg, A. Clark, H. Soyer, J. W. Rae, S. Noury, A. Ahuja, S. Liu, D. Tirumala, et al. 2019. V-mpo: On-policy maximum a posteriori policy optimization for discrete and continuous control. In *arXiv preprint arXiv:1909.12238*.

A. Vezhnevets, Y. Wu, M. Eckstein, R. Leblond, and J. Z. Leibo. 2020. Options as responses: Grounding behavioural hierarchies in multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 9733–9742. PMLR.

J. G. de Villiers. 2021. The role(s) of language in theory of mind. In *The neural basis of mentalizing*, pages 423–448. Springer.

L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, et al. 2023a. A survey on large language model based autonomous agents. In *arXiv preprint arXiv:2308.11432*.

R. Wang, E. Zelikman, G. Poesia, Y. Pu, N. Haber, and N. D. Goodman. 2023b. Hypothesis search: Inductive reasoning with language models. In *arXiv preprint arXiv:2309.05660*.

J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837.

H. Zhang, W. Du, J. Shan, Q. Zhou, Y. Du, J. B. Tenenbaum, T. Shu, and C. Gan. 2023. Building cooperative embodied agents modularly with large language models. In *arXiv preprint arXiv:2307.02485*.