Theory of Mind in Multi-Agent Systems

Ini Oguntola

CMU-ML-25-118

September 2025

Machine Learning Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Katia Sycara (Chair)
Pradeep Ravikumar
Changliu Liu
Michael Lewis (University of Pittsburgh)

Submitted in partial fulfillment of the requirements
for the Degree of Doctor of Philosophy.

This research was sponsored by: Air Force Research Laboratory awards FA95501510442, FA95501810097, FA95501810251; Army awards W911NF1610514, W911NF2220001, W911QX24F0049; Defense Advanced Research Projects Agency awards HR001120C0036, HR00112490409; Department of Agriculture awards 20176700726152, 20236702139073; National Science Foundation award IIS1724222; and Office of Naval Research award N000142312840.

Abstract

The ability to model the internal mental states of others – known as *theory of mind* — is a crucial aspect of human social intelligence. In this work we present an interpretable family of approaches to modeling theory of mind within artificial agents based on *concept learning*, and explore this in the context of deep imitation and reinforcement learning. We posit that endowing artificial agents with theory of mind increases both model transparency and trust from a human perspective as well as task performance with respect to navigating social dynamics in both competitive and cooperative multi-agent scenarios.

The first part of this work focuses on theory of mind as a framework for modeling artificial agents in imitation learning. Completed work develops a modular neural framework to explicitly model theory of mind from observed trajectories, and introduces concept whitening as an approach to ensuring interpretability of learned policies. The efficacy of this approach is demonstrated with experiments on data from human participants on a search and rescue task in Minecraft.

The second part of this work focuses on higher-level theory of mind inference within the context of multi-agent reinforcement learning. Completed work extends our concept-based approach to interpretability by introducing an information theoretic-variant to concept learning via bottleneck while incorporating residual latent knowledge. Experiments in a variety of cooperative, competitive, and mixed multi-agent scenarios show that introducing higher-order theory of mind inference as an intrinsic reward for agents can lead to policies with improved coordination, strategy, and efficiency of communication.

The final part of this work focuses on deception via theory of mind in zero-sum games. We present an approach for improving performance in zero-sum Multi-Agent Reinforcement Learning (MARL) by rewarding agents for being deceptive. Our framework utilizes opponent theory of mind (ToM) modeling error over beliefs and intents as a signal to induce deceptive behavior. We extend this framework to higher-order ToM reasoning, where beyond 0th-order beliefs about the environment, an agent aims to confound an opponent's 1st and 2nd order beliefs (i.e. beliefs about beliefs). We present empirical results in Barrage Stratego, Kuhn Poker, and Mafia, where we find that higher-order deceptive policies consistently outperform baseline and lower-order deceptive policies.

Acknowledgments

I would like to thank my advisor, Katia Sycara, for her invaluable guidance throughout my entire time here at CMU. Not only did she advise me directly, but her lab fostered a wonderful environment for me to grow personally and intellectually. In particular I would like to note Dana Hughes, Joseph Campbell, and Simon Stepputtis as senior figures throughout my time in the lab who became important research mentors for me.

I would also like to acknowledge my parents, Adebowale and Oyebimpe Oguntola, for their unending love and support, not only during my PhD but throughout my entire life leading up to this point; if it was not for your relentless effort to help me explore my insatiable intellectual curiosity, I would never have made it this far.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Human intelligence is remarkable not just for the way it allows us to navigate environments individually, but also how it operates socially, engaging with other intelligent actors. Humans naturally build high-level models of those around them, and are able to make inferences about their beliefs, desires and intentions (BDI) [Georgeff et al., 1998]. These inferences allow people to anticipate the behaviors of others, use these predictions to condition their own behavior, and then anticipate potential responses. In both psychology and machine learning this is referred to as theory of mind (ToM) [Baker et al., 2011, Rabinowitz et al., 2018, Cuzzolin et al., 2020], which aims to model not only the external behavior of other entities but their internal mental states as well. The developmental psychology literature has found that children as young as 4 years old have already developed a ToM, a crucial ability in human social interaction [Astington and Edward, 2010]. ToM can enable discovery of false or incomplete beliefs and knowledge and can thus facilitate interventions to correct false beliefs. Therefore, work in enabling agents to develop ToM is a crucial step not only in developing more effective multi-agent AI systems but also for developing AI systems that interact with humans, both cooperatively and competitively [Cuzzolin et al., 2020].

Traditionally, agent-modeling approaches within reinforcement learning (RL) and imitation learning largely ignore the idea of internal mental states, typically only focusing on reproducing the external behavior [Foerster et al., 2018, Wen et al., 2019]. This limits their ability to reason in a deeper way about entities that they interact with. While prior work has explored providing agents with models of some aspect of a human's mental state, such as reward [Choudhury et al., 2019] or rationality [Shah et al., 2019], there is a growing body of work in the machine learning literature aimed towards developing artificial agents that exhibit theory of mind [Baker et al., 2011, Rabinowitz et al., 2018, Jara-Ettinger, 2019, Fuchs et al., 2021]. Even beyond simply providing a helpful inductive bias for modeling behavior, ToM reasoning has the potential to enable the discovery and correction of false beliefs or incomplete knowledge, facilitate efficient communication and coordination, and improve human-agent teaming [Zeng et al., 2020, Sclar et al., 2022, Oguntola et al., 2021].

The work of Aru et al. [2023] highlights key challenges regarding the difficulty of evaluating

current deep learning ToM approaches. In particular, from a human perspective we may solve a task using an already-developed internal theory of mind, whereas an artificial agent may be able to learn simpler decision rules or take advantage of spurious correlations as shortcuts, and it is difficult to determine whether ToM has actually been learnt.

Here we consider the inverse – rather than solving a task and hoping this implicitly induces a theory of mind, we instead explicitly learn a theory of mind over semantically grounded beliefs and intents, and then reason over them to solve the task (e.g. "I believe the door is locked, so I pick up the key"); this can be understood as zero-order theory of mind [Hedden and Zhang, 2002]. But we can extend this to even higher-order reasoning: modeling the theory of mind of another agent (i.e., my beliefs about your beliefs), or even modeling another agent's model of theory of mind of another agent (i.e. my beliefs about your beliefs about their beliefs).

Our fundamental research question is the following: *can modeling other agents ToM help improve performance and generalization – with respect to coordination, deception, exploitation of false belief, etc – in multi-agent settings?*

In this work we develop an approach that leverages concept-based interpretability methods to ground semantically meaningful beliefs and intents within RL policies [Schwalbe, 2022]. We then propose the use of ToM reasoning over the beliefs of other agents as intrinsic motivation in multi-agent scenarios. We run experiments in both cooperative and adversarial competitive environment and show results that indicate this approach improves multi-agent performance, with respect to both coordination and deception. We provide a general framework for reasoning about theory of mind in artificial systems, and introduce interpretable approaches for specifically modeling beliefs and intents with concept learning.

Our core contributions are as follows:

- We create a method of concept learning with an residual disentangled based on minimizing mutual information
- We formalize a unified framework for theory of mind in decision-making process via concepts
- We introduce a modular approach to modeling theory of mind in RL policies via our concept-residual method
- We extend this approach to cooperative scenarios by treating ToM modeling itself as a task for intrinsic reward
- We extend this approach to adversarial / deceptive scenarios by treating the manipulation of others' ToM models as a task for intrinsic reward
- We demonstrate that higher-order ToM reasoning consistently improves performance in tasks these with social dynamics, with diminishing returns past 2nd-order reasoning.

Chapter 2

Artificial Theory of Mind

The following survey provides a detailed overview of the current landscape of Theory of Mind (ToM) in artificial intelligence and establishes the conceptual framework for three seminal papers: "Deep Interpretable Models of Theory of Mind," "Theory of Mind as Intrinsic Motivation for Multi-Agent Reinforcement Learning," and "Deception Is Its Own Reward." The central argument of this analysis is that the field is moving beyond mere behavioral mimicry towards the development of systems with genuine, interpretable, and strategically motivated social intelligence. This transition is not only a technical evolution but a response to foundational limitations in traditional AI development.

This chapter first defines the multidisciplinary roots of ToM and situates its computational implementation within the philosophical debates of the field. It then deconstructs three core weaknesses of contemporary AI: the **Interpretability Dilemma**, where models function as inscrutable "black boxes"; the **Reward Problem**, which highlights the limitations of extrinsic rewards in fostering complex social behaviors; and the **Strategic Reasoning Gap**, where current benchmarks fail to assess an agent's ability to functionally apply ToM in dynamic, adversarial environments.

The work presented in this thesis is presented as a synergistic solution to these challenges. We address the interpretability dilemma by proposing a framework that explicitly grounds and verifies a model's internal mental states. This provides a transparent foundation for trust and alignment. We tackle the reward problem by establishing that the very act of modeling an opponent's mind can serve as a potent intrinsic reward signal, thereby incentivizing complex social reasoning. Finally, we operationalize this intrinsic motivation in a challenging adversarial setting, demonstrating that this approach can produce sophisticated, human-like deceptive strategies that outperform traditional methods.

Collectively, this work represent a significant step toward a new paradigm of AI—one that is not only capable of interacting with human-centric environments but also provides a transparent and verifiable pathway to understanding the social complexities of those interactions. This sets the stage for a critical discussion of the ethical implications and future research directions necessary to ensure the safe and responsible development of socially intelligent AI.

2.1 The Foundations and Current State of Artificial Theory of Mind

2.1.1 Theory of Mind: A Multidisciplinary Perspective

The concept of Theory of Mind (ToM) is foundational to understanding human social intelligence and interaction. It is defined as the ability to attribute mental states—such as beliefs, desires, intentions, and emotions—to oneself and others, and to understand that these mental states may differ from one's own or from reality itself [Premack and Woodruff, 1978, Wimmer and Perner, 1983]. This ability, first identified in a 1978 paper by David Premack and Guy Woodruff concerning chimpanzees, is a cornerstone of human social cognition [Premack and Woodruff, 1978, Baron-Cohen et al., 1985]. In developmental psychology, the emergence of ToM in children is well-documented, with the capacity for explicit social reasoning, including the ability to solve false-belief tasks, typically solidifying around ages four to five [Wimmer and Perner, 1983, Samson et al., 2020]. This developmental understanding provides a roadmap for researchers seeking to replicate these capabilities in artificial agents [Wimmer and Perner, 1983, Samson et al., 2020].

The computational implementation of ToM is rooted in a rich philosophical tradition, notably the Computational Theory of Mind (CTM) [Fodor, 1975, Simon and Newell, 1972, Putnam, 1967]. CTM posits that the mind is a computational system and that cognition is a form of information processing [Searle, 1980, Fodor, 1975, Putnam, 1967]. This view is often associated with functionalism, a philosophical stance that defines mental states by their causal role or function within a system, rather than by their physical composition [Putnam, 1980]. Functionalism's central concept, "multiple realizability," suggests that a mind can be physically instantiated in various substrates, from biological neurons to silicon processors, as long as the functional requirements are met [Putnam, 1980]. This provides a theoretical basis for the very possibility of artificial intelligence.

However, a fundamental challenge to this perspective is John Searle's "Chinese Room Argument" [Searle, 1980]. Searle's thought experiment argues that a system can manipulate symbols to simulate understanding without truly possessing it. This critique is a direct refutation of the "strong AI hypothesis," which claims that a correctly programmed computer literally possesses a mind [Searle, 1980]. The core issue here is the **symbol grounding problem**: how do abstract symbols, manipulated purely by a computer, acquire intrinsic meaning and connect to real-world objects or concepts? [Harnad, 2007]. This problem highlights the difference between a syntactic system that can pass the Turing test and a system with genuine semantic understanding [Searle, 1980].

The interdisciplinary convergence of philosophy, psychology, and computer science highlights that the pursuit of artificial ToM is not merely a technical challenge of replicating behavior. It is an endeavor to address whether and how an artificial system can move beyond syntactic manipulation to achieve genuine, grounded understanding [Harnad, 2007]. This is a crucial distinction that informs the entire field, compelling researchers to develop methods that not only produce intelligent behavior but also ensure the underlying computational processes are transparent and semantically meaningful. The symbol grounding problem, far from being a purely academic

debate, is a practical hurdle that necessitates new architectural designs and evaluation methods, providing a direct conceptual foundation for the first paper discussed.

2.1.2 Computational Game Theory and Multi-Agent Systems

The development of AI has moved from single-agent systems, focused on maximizing rewards in a static environment, to multi-agent reinforcement learning (MARL), which introduces the profound complexities of social interaction [Zhang et al., 2021]. In MARL, the environment is inherently non-stationary; an agent's optimal strategy depends not only on its own actions but on the unpredictable and evolving strategies of other agents [Wang et al., 2022]. This violates the Markov property that underpins most traditional reinforcement learning algorithms and necessitates a more sophisticated approach.

This is where computational game theory provides a critical lens for analysis. Games with incomplete information, such as card games like poker or bridge, are situations where players lack common knowledge about the game they are playing [Lanctot et al., 2017]. Hungarian economist John Harsanyi developed the concept of Bayesian games to model strategic decision-making in such environments [Harsanyi, 1967]. A Bayesian game replaces a game of incomplete information with a game of complete information by treating a player's private information (or "type") as a random variable with a known probability distribution [Harsanyi, 1967]. This approach allows for the formal, mathematical analysis of games where players must reason about the beliefs and strategies of their opponents without having perfect knowledge [Lanctot et al., 2017].

A primary response to this challenge has been the development of "agent modeling," where agents learn to form internal representations and predictions of other participants' behavior, effectively implementing a computational form of ToM [Oguntola et al., 2021, Rabinowitz et al., 2018]. Recent advancements demonstrate that mastering complex social dynamics is a prerequisite for achieving human-level performance in adversarial environments. The landmark achievement of Meta AI's CICERO, the first AI to achieve human-level performance in the game of Diplomacy, is a prime example [FAIR et al., 2022]. Unlike systems designed for perfect-information games like chess or Go, CICERO's success hinges on its ability to master communication, negotiation, and strategic deception within a zero-sum, incomplete-information environment.

This transition in research focus reveals a critical limitation of traditional MARL: a narrow definition of success. The field has historically measured success by the maximization of extrinsic rewards, often neglecting the complex social skills required to navigate human-like interaction. This is problematic, as behaviors like cooperation, negotiation, and deception often involve delayed or sparse rewards that are difficult for a system to learn from. The shift from simply winning a game (extrinsic reward) to demonstrating social competence is a re-evaluation of what constitutes true intelligence in a multi-agent setting. This conceptualization of social competence, which moves beyond brute-force optimization to encompass an understanding and manipulation of other agents' mental states, creates a clear need for a new class of reward signals. This provides the conceptual motivation for the second and third papers, which directly address this gap by exploring how to explicitly and intrinsically reward socially intelligent behaviors.

2.2 Core Weaknesses and Gaps in Machine Theory of Mind

2.2.1 The Interpretability Dilemma: From Black-Box Behavior to Internal States

One of the most profound weaknesses in the field of AI, particularly in models like deep neural networks and large language models (LLMs), is the "black box" problem. While these systems excel at performing complex tasks by mapping inputs to outputs, the internal processes that produce their results are often opaque and inscrutable to human observers [Molnar, 2020]. This opacity presents a significant barrier to the development of robust artificial ToM. While LLMs, for instance, can demonstrate impressive performance on classic false-belief tasks, there is a fundamental question of whether this behavior stems from genuine ToM reasoning or is merely a product of statistical pattern matching and spurious correlations from their vast training corpora [Kim et al., 2023, Chen et al., 2024]. This phenomenon, sometimes referred to as the "Clever Hans" effect, implies that a model may be using shortcuts that do not reflect a true understanding of the underlying social dynamics [Chen et al., 2024].

The field's response to this problem has revealed a fundamental flaw in traditional evaluation methods. Initial ToM evaluations for LLMs relied on narrative-based false-belief tasks, similar to the classic Sally-Anne test [Kosinski, 2024]. The impressive performance of models like GPT-4 on these tasks led to claims that ToM-like abilities had "spontaneously emerged" as a byproduct of improved language skills [Bubeck et al., 2023, Kosinski, 2024]. However, a skeptical community of researchers responded by developing more rigorous, adversarial benchmarks to test whether models were truly reasoning or merely exploiting superficial patterns in the data [Kim et al., 2023, He et al., 2023, Chen et al., 2024, Fan et al., 2025].

These new benchmarks, designed to stress-test ToM capabilities, include:

Table 2.1: New-Generation Benchmarks for Evaluating Machine ToM

Benchmark Name	**Key Purpose**	**Primary Evaluation Method**	**Core Findings**
FANToM [Kim et al., 2023]	To stress-test ToM in information-asymmetric conversational contexts. Designed to identify an illusory sense of ToM .	Question answering on multi-party conversations with dynamic character absences.	State-of-the-art LLMs perform significantly worse than humans and lack coherent reasoning.
ToMBench [Chen et al., 2024]	A systematic, automated, and bilingual benchmark to evaluate 31 specific ToM abilities across 8 tasks.	Multiple-choice questions on scenarios built from scratch to avoid data leakage.	The most advanced LLMs like GPT-4 lag behind human performance by over 10 percentage points, indicating a lack of human-level ToM.

Table 2.1: New-Generation Benchmarks for Evaluating Machine ToM (cont.)

Benchmark Name	Key Purpose	Primary Evaluation Method	Core Findings
SoMi-ToM [Fan et al., 2025]	To evaluate multi-perspective ToM in embodied, multimodal social interactions. Designed to address the gap between static text and real-world scenarios.	First- and third-person perspective videos of embodied agents.	Large Vision-Language Models (LVLMs) perform significantly worse than humans, with a performance gap of over 26%.
Hi-ToM [He et al., 2023]	To evaluate higher-order theory of mind reasoning in LLMs.	Natural language question-answering over various story types.	LLMs likely do not exhibit true ToM, particularly in the cases of higher-order reasoning.

The results from these newer benchmarks have validated the concerns about the black-box problem. The consistent performance drop of state-of-the-art models when faced with these more rigorous tests confirms that they often rely on spurious correlations rather than genuine, transparent reasoning [Kim et al., 2023, Chen et al., 2024, Fan et al., 2025]. The existence of these benchmarks represents a crucial shift in the field, moving from a focus on behavioral output to a deeper investigation of the underlying mechanisms. The discovery that internal representations can be linearly decoded from neural activations is a promising development that points toward a new way to solve the interpretability problem by directly probing a model's internal representations of belief [Herrmann and Levinstein, 2024, Zhu et al., 2024].

2.2.2 The Reward Problem: The Challenge of Inducing Complex Sociality

A significant limitation in multi-agent reinforcement learning (MARL) is the difficulty of rewarding complex social behaviors [Zhang et al., 2021]. Traditional RL relies on extrinsic rewards, which are signals provided by the environment to guide an agent's learning. While this is effective for simple tasks, it is fundamentally ill-suited for the nuanced and multifaceted nature of human interaction, where behaviors like cooperation, negotiation, or deception often result in sparse, delayed, or even non-existent external rewards [Zhang et al., 2021, Crites and Barto, 1998, Aubret et al., 2019, Ning and Xie, 2024]. This makes it extremely challenging for agents to learn sophisticated strategies. Manual reward shaping, a common technique to address sparse rewards, is a brittle solution that can lead to "reward hacking," where an agent exploits the shaped reward function in an unintended way, prioritizing a sub-goal over the main task objective [Zhang et al., 2021, Aubret et al., 2019].

Intrinsic motivation (IM) offers a potential solution by rewarding agents for internal goals, such as exploring novel states or reducing prediction error [Pathak et al., 2017, Aubret et al., 2019]. However, these general forms of IM are not tailored to the unique dynamics of social interaction.

A generic curiosity-driven agent may be motivated to explore an environment, but it has no explicit incentive to understand or influence the minds of the other agents within it [Oguntola et al., 2023]. A reward is needed that is directly tied to the process of social reasoning itself [Oguntola et al., 2023].

The fundamental problem is the absence of a mechanism to reward the mastery of social complexity. The success of a strategic player often hinges on their ability to reason about and influence the internal states of their opponents, a process that is invisible to the environment's extrinsic reward function. This presents a major challenge to building socially capable AI.

2.2.3 The Strategic Reasoning Gap: Incomplete Information and Deception

A significant limitation of many traditional MARL agents is their inability to perform sophisticated strategic reasoning in adversarial environments with incomplete information [Sandholm, 2015, Li et al., 2024]. In games like Kuhn Poker or Barrage Stratego, success is not determined by simply maximizing one's own direct reward but by influencing the opponent's perception of the game state [Kuhn, 1950, Federation, 2025]. Standard models, being optimized for direct rewards, do not inherently incentivize the manipulation of an opponent's beliefs—a key aspect of strategic depth in competitive settings [Chelarescu, 2021].

The inability to reason about an opponent's beliefs and strategies—a form of ToM reasoning—is a critical gap that prevents agents from achieving human-level strategic performance. While some notable successes have been achieved, such as Meta AI's CICERO in the game of Diplomacy, these are often bespoke, custom-engineered solutions that do not provide a generalizable framework for developing strategic reasoning across a variety of domains. This highlights the need for a fundamental shift in the training paradigm to one that proactively cultivates a deep understanding of adversarial mental states, enabling agents to engage in sophisticated behaviors like bluffing or feint attacks. The problem is not merely a lack of a reward signal, but a lack of a framework that defines and quantifies complex, strategic behaviors like deception as a central goal.

Chapter 3

Theory of Mind as Imitation Learning

In this chapter we explore theory of mind from the perspective of a pure observer. Here the only task we are concerned with is interpretable agent modeling, and can be understood as a form of imitation learning from observed trajectories. We will develop an interpretable modular neural framework for modeling the beliefs, intentions, and behavior of observed actors, and experimentally validate it over human participants in a Minecraft environment.

3.1 Neural ToM Framework

3.1.1 Purpose

It is important to note that the ToM framework is used by an observer to infer mental states of an observed entity, but not to define the entity's observable behavior. The action predictions produced by the ToM model can be treated as a policy to forecast the entity's future behavior, and imitation learning can provide a training signal for the ToM model, but our purpose here is *not* to train an agent to perform a task.

3.1.2 Overview

We develop a modular theory of mind framework for an observing agent to use to infer the mental state of an observed human. As shown in Figure 3.1, we model the decision making process of an observed entity as follows:

Belief Model The belief model $m : \mathcal{B} \times \mathcal{O} \rightarrow \mathcal{B}$ is responsible for updating the belief state. Given current belief $b \in \mathcal{B}$ and observation $o \in \mathcal{O}$, the belief model m outputs an updated belief state $b_{new} = m(b, o)$.

Desire Model The desire model $g : \mathcal{B} \rightarrow \mathcal{I}$ is responsible for calculating the intent $z \in \mathcal{I}$ given updated belief b; that is, $g(b) = z$.

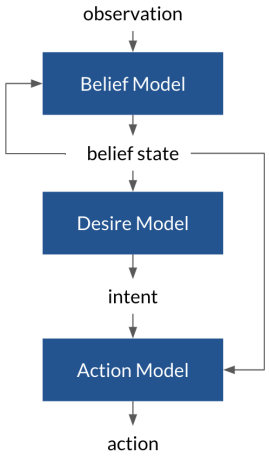


Figure 3.1: Modular theory of mind (ToM) framework.

Action Model The action model $f : \mathcal{B} \times \mathcal{I} \to \mathcal{A}$ is responsible for generating or predicting an action $a \in A$, given the belief state b and intent z; that is, $f(b, z) = a$.

The modules in our framework impose an inductive bias that reflects the BDI model of agency in folk psychology [Georgeff et al., 1998]. Also, as we later describe, modularizing the framework in this way allows for combining heuristic and data-driven components (e.g., neural networks).

In the context of our broader ToM framework, the observations are our inputs X_t, the actions are our outputs Y_t, and our latent variable consists of time-dependent belief and intent, $Z_t = V_t$ over domain $\mathcal{B} \times \mathcal{I}$. The action model explicitly models the conditional distribution $p(Y_t \mid Z_t)$.

3.1.3 Combining Differentiable and Heuristic Components

Practically speaking, we need to make sure that the belief space $\mathcal{B}$ corresponds to something that can be understood as belief, that the intent space $\mathcal{I}$ corresponds to what can be understood as intent, etc. If we were to implement all three components with neural networks and train end-to-end, there is nothing that constrains the intermediate outputs to correspond to what we expect (e.g. no way to ensure the desire model output actually represents an “intent").

We can mitigate this difficulty by imposing additional structure on the pipeline. The simplest way to do this is to replace one or more of the models with rule-based models, and/or to impose structural constraints on the input/output space of these models. Here we are primarily interested in inferring intent, so we choose to model the desire model with a neural network, while modeling the belief and action models with rule-based heuristics.

Given a planning task, we structure the belief state as a grid/graph of locations, and use a rule-based belief model to update this belief state given an observation. Additionally, we replace the action model with A* search, and structure the intents from the desire model as locations of subgoals.

In the setup described above, the belief and action models are rule-based, and the desire model is

the sole trainable component. The rule-based belief model does not pose any issue with respect to the differentiability of the pipeline (one can think of it as simply preprocessing the input observation). However, we cannot optimize for the final output in any gradient-based way, as the output of the desire model is the input to the non-differentiable action model, which produces the final output. Unless we have ground-truth intents, training the desire model becomes difficult.

3.1.4 Inverse Action Model

Given belief state $b \in \mathcal{B}$, observed action $a \in \mathcal{A}$, a set of intents $\mathcal{I}$, and non-differentiable action model $f : \mathcal{B} \times \mathcal{I} \to \mathcal{A}$, we want to learn a desire model g to model a conditional distribution $p(z \mid b)$ such that

$$\mathbb{E}[a \mid b] = \mathbb{E}_{z \sim g(b)}[f(b, z) \mid b]$$

where $z \in \mathcal{I}$ is the intent. However, we may not have access to any samples from such a distribution (i.e. no ground truth $z \in \mathcal{I}$ for given b, a pairs).

Alternatively, we can learn to model the distribution $p(z \mid b, a)$ with an "inverse action model" h. This density of this distribution is proportional to

$$p(z \mid b, a) \propto p(a, z \mid b) = p(a \mid b, z) \cdot p(z \mid b) \tag{3.1}$$

Because we have direct access to rule-based action model f we can sample from $p(a \mid b, z)$, and thus given some kind of prior $p(z \mid b)$ we can sample from $p(z \mid b, a)$ to learn h.

Once we have learned an inverse action model h, then for each belief-action pair (b, a) we can then simply use h to sample intents from $p(z \mid b, a)$, and use these sampled intents to train the desire model g in a supervised manner.

3.1.5 Training

Training is done in two stages: the first stage trains the inverse action model, the second stage trains the desire model. In each stage, once we gather the necessary data, we train using stochastic gradient descent (SGD).

Training Inverse Action Model

To train the inverse action model, we first collect belief states by sequentially running observations from human trajectories through the rule-based belief model and storing the resulting belief states at each timestep. These trajectories can be from human participants or potentially even from artificial agents trained to perform the task.

Now that we have collected these belief states, for each belief state b, we sample an intent z given some prior $p(z \mid b)$, and create a set of b, z pairs. Next, for each belief-intent pair, we generate an action $a = f(b, z)$, creating a dataset of belief-action-intent triples (b, a, z). Finally, we train the inverse action model on this dataset to predict intents given beliefs and actions; that is, using our generated dataset of (b, a, z) tuples, given b, a, predict z.

Pseudocode for this training process is provided in Algorithm 1.

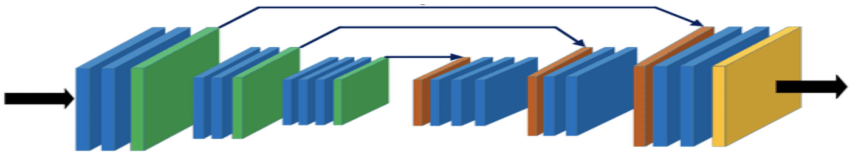


Figure 3.2: Encoder-decoder architecture used for the desire and inverse-action models in our experiments, inspired by U-Nets for image-segmentation [Ronneberger et al., 2015]. Blue indicates convolutional layers, green indicates pooling, brown indicates up-sampling, and yellow is a final linear layer.

Training Desire Model

To train the desire model, we first collect belief states by running the observations from *human* trajectories through the belief model. We also store the corresponding observed actions for each belief state. We can then generate belief-intent pairs for the desire model by sampling intents from our inverse action model $z \sim h(b, a)$ for each belief-action pair. The target intents are formed by combining a) the probability distribution from the inverse action model over the previous belief state, and if available b) the next realized intent (from the future, in a post-hoc manner). Finally we train the desire model on this data to predict intent given belief.

Pseudocode for this process is provided in Algorithm 2.

Algorithm 1: Training inverse action model

Input: set of human trajectories $\mathcal{T} = \{\tau_1, \tau_2, \dots\}$, belief model s, action model f
$\mathcal{D}_{baz} \to \emptyset$
for $\tau \in \mathcal{T}$ **do**
  $o_1, \dots, o_n \to \tau$
  $b_0 \to$ Uniform
  for $t = 1, \dots, n$ **do**
    $b_t \to s(b_{t-1}, o_t)$
    for $i = 1, \dots, m$ **do**
      $z_t \sim p(z \mid b_t)$
      $a = f(b_t, z_t)$
      $\mathcal{D}_{baz} = \mathcal{D}_{baz} \cup \{(b_t, a, z_t)\}$
    end
  end
end
Initialize neural network parameters θ_h
Use SGD to train $h(b, a \mid \theta_h)$ on dataset $\mathcal{D}_{baz}$

3.2 Concept Whitening

Concept whitening (CW) is a mechanism introduced by Chen et al. [Chen et al., 2020] for modifying neural network layers to increase interpretability. Broadly, it aims to enforce structure

Algorithm 2: Training desire model

Input: set of human trajectories $\mathcal{T} = \{\tau_1, \tau_2, \dots\}$, belief model s, inverse action model h

$\mathcal{D}_{bz} \to \emptyset$
for $\tau \in \mathcal{T}$ **do**
  $(o_1, a_1), \dots, (o_n, a_n) \to \tau$
  $b_0 \to$ Uniform
  for $t = 1, \dots, n$ **do**
    $b_t \to s(b_{t-1}, o_t)$
    $z_t \sim h(b_t, a_t)$
    $\mathcal{D}_{bz} = \mathcal{D}_{bz} \cup \{(b_t, z_t)\}$
  end
end
Initialize neural network parameters θ_g
Use SGD to train $g(b \mid \theta_g)$ on dataset $\mathcal{D}_{bz}$

on the latent space by aligning the axes with predefined human-interpretable concepts. While this technique was developed for the purpose of image classification, here we adapt the idea in the context of intent inference with the desire model. By explicitly defining a set of concepts that can serve as "explanations" for intent inferences, we can use concept whitening to allow for interpretability via identification of the most important concepts for any inference.

We also note that although we consider concept whitening in the context that can broadly be categorized as behavioral cloning, our approach to *interpretable* agent-modeling is framework-agnostic and could potentially be applied to other reinforcement learning and imitation learning contexts.

3.2.1 Technical Details

Given latent representation $\mathbf{Z} \in \mathbb{R}^{n \times d}$, let $\mathbf{Z_C} \in \mathbb{R}^{n \times d}$ be the mean-centered latent representation. We can calculate the ZCA-whitening matrix $\mathbf{W} \in \mathbb{R}^{d \times d}$ as in [Huang et al., 2019], and thus decorrelate and standardize the data via whitening operation ψ:

$$\psi(\mathbf{Z}) = \mathbf{W}\mathbf{Z_C} = \mathbf{W}(\mathbf{Z} - \mu \mathbf{1}^\top) \tag{3.2}$$

where $\mu = \frac{1}{n} \sum_{i=1}^{n} \mathbf{z}_i$ is the latent sample mean.

Now say we are given concepts $c_1 \dots c_k$ that can be characterized by corresponding auxiliary datasets $\mathbf{X}_{c_1} \dots \mathbf{X}_{c_k}$, and assume we have an orthogonal matrix $\mathbf{Q} \in \mathbb{R}^{d \times d}$ such that the data from $\mathbf{X}_{c_j}$ has high activation on the j-th axis (i.e. column $\mathbf{q}_j$). Then the concept-whitened representation is given by:

$$\hat{\mathbf{Z}} = \mathbf{Q}^\top \mathbf{W}\mathbf{Z_C} \tag{3.3}$$

Training alternates between optimizing for the main objective (i.e. the network's final output) and optimizing the orthogonal matrix $\mathbf{Q}$ for concept-alignment. To optimize $\mathbf{Q}$, we maximize

the following objective:

$$\max_{\mathbf{q_1} \cdots \mathbf{q_k}} \sum_{j=1}^{k} \frac{1}{n_j} \sum_{\mathbf{x}_{c_j} \in \mathbf{X}_{c_j}} \mathbf{q}_j^\top \hat{\mathbf{z}}_{\mathbf{x}_{c_j}} \tag{3.4}$$

where $\hat{\mathbf{z}}_{\mathbf{x}_{c_j}}$ denotes the concept-whitened latent representation in the model on data sample from concept c_j. Orthogonality can be maintained when optimization is performed via gradient descent and curvilinear search on the Stiefel manifold [Wen and Yin, 2013]. A more detailed description of concept whitening and the optimization algorithm can be found in [Chen et al., 2020].

3.2.2 Concept Whitening for Intent Prediction

We can modify this idea to the context of explanatory concepts for inferring intents. Specifically, we consider the desire model (Fig. 3.2) and insert a concept whitening layer (see Fig. 3.3).

First we define a set of concepts $C = \{c_1, \ldots, c_k\}$; these concepts should correspond to appropriate human-interpretable reasons or "explanations" for intent prediction given the problem domain. We also must be able to identify a subset of timesteps from our trajectories where each concept applies, either directly from the trajectory data, or from external labels.

Recall that the desire model's inputs are belief states, which we can generate sequentially by passing the observations from each trajectory timestep through the belief model. Then for each concept c_j we consider only the belief states from the timesteps where c_j is known to apply, and aggregate them into auxiliary dataset $\mathbf{B}_{c_j}$.

Then training alternates between:

1. Optimizing for intent prediction, given a belief state and a ground truth intent
2. Concept-aligning the CW orthogonal matrix $\mathbf{Q}$ by maximizing the activation along axis j for each auxiliary dataset $\mathbf{B}_{c_j}$

Pseudocode for this process is provided in Algorithm 3.

3.3 Experiments

3.3.1 Task Domain

We consider a simulated search and rescue task in a Minecraft environment. The scenario simulates a damaged building after a disaster, with areas of the building layout perturbed with collapsed rubble, wall openings, and fires. There are 34 injured victims within the building who will die if left untreated. For convenience and simplicity, victims are represented as blocks. Out of these victims, 10 of these are critically injured and will expire after 5 minutes. These critical victims take 15 seconds to triage and are worth 30 points each. Other victims are considered "non-critical", but will expire after 10 minutes. Non-critical victims take 7.5 seconds to triage

Algorithm 3: Training desire model with concept whitening

$\mathcal{D} \to \emptyset$
for $\tau \in \mathcal{T}$ **do**
  $(o_1, a_1), \ldots, (o_n, a_n) \to \tau$
  $b_0 \to$ Uniform
  for $t = 1, \ldots, n$ **do**
    $b_t \to \text{BM}(b_{t-1}, o_t)$
    $z_t \sim \text{IAM}(b_t, a_t)$
    $\mathcal{D} = \mathcal{D} \cup \{(b_t, z_t)\}$
  end
end
for $e = 1, ..., num\_epochs$ **do**
  Train DM on $\mathcal{D}$ with gradient descent
  if $e \mod 5 = 0$ **then**
    for $j = 1, ..., k$ **do**
      Maximize activation of $\mathbf{B}_{c_j}$ on the j-th column of $\mathbf{Q}$ (see [Chen et al., 2020])
    end
  end
end

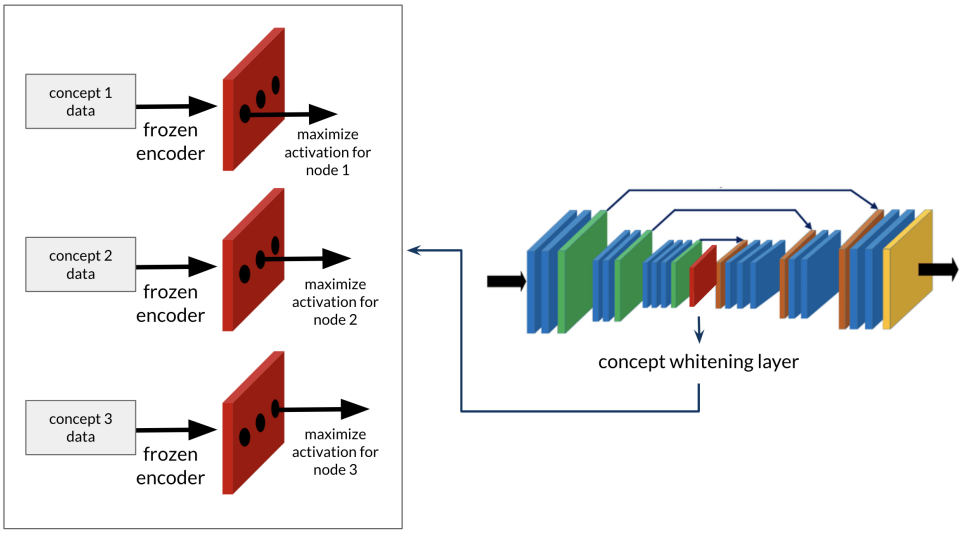


Figure 3.3: Desire model with concept whitening.

and are worth 10 points each. The goal of the task is to earn as many points as possible within a 10 minute mission.

3.3.2 Human Data For Training and Evaluation

All experiments are performed using a set of 75 trajectories previously collected from human participants [Huang et al., 2021].

Prior to each mission, participants were given information on the task and the original building layout. However, the knowledge conditions of certain participants were manipulated by partially withholding information. Some participants were not informed of the cost-benefit tradeoffs (i.e. the knowledge that critical victims take 15 seconds to rescue and are worth 30 points and non-critical victims take 7.5 seconds to rescue and are worth 10 points). Knowledge was also manipulated via a beep signal that activated whenever the participant was near a room with a victim (1 beep for non-critical victim, 2 beeps for critical victim); certain participants were not told the meaning of the signal.

Participants were run in 3 knowledge conditions:

1. No knowledge of critical-victim tradeoff, no knowledge of signal meaning
2. Knowledge of critical-victim tradeoff but not of signal meaning
3. Knowledge of both critical-victim tradeoff and signal meaning

3.3.3 Intent Prediction

We represent intents as (x, y) positions the participant intends to navigate towards. Specifically, we consider victims, doors, and room openings as locations-of-interest, which frames the intent prediction task as predicting either the *next room* to be visited or the *next victim* to be triaged. The predictions are accumulated at each timestep ($\sim$ 0.5 seconds per timestep) between visits of locations-of-interest, and then their mode is evaluated against the ground truth. We evaluate on a held-out test set of 20% of participant trajectories.

3.3.4 Concepts

We defined a set of 10 concepts related to mission timer, knowledge condition, and field of view (see Table 3.1). We consider 3 subsets:

- Concept Set I is the full concept set
- Concept Set II omits the knowledge condition concepts
- Concept Set III omits both knowledge condition and mission timer concepts

The field of view and mission timer concepts were labeled directly from the data; the knowledge condition concepts are labeled with external knowledge of the condition for each participant trajectory.

Table 3.1: Concepts

Concept	Concept Sets
Mission timer between 0-3 minutes	I, II
Mission timer between 3-5 minutes	I, II
Mission timer between 5-8 minutes	I, II
Mission timer $>$ 8 minutes	I, II
Knowledge condition 1 (no triage, no signal)	I
Knowledge condition 2 (triage, no signal)	I
Knowledge condition 3 (triage, signal)	I
Door / opening in field of view	I, II, III
Non-critical victim in field of view	I, II, III
Critical victim in field of view	I, II, III

3.3.5 Results

We compare the accuracy of ToM model intent predictions under 3 methods: training without Concept Whitening (CW), training from scratch with CW, and transfer learning by initializing a CW model with the weights of a pretrained non-CW model. Results are provided in Table 3.2, where we see that introducing concept whitening for interpretability actually results in *increased accuracy* of the model. The inductive bias produced by shaping the latent space to correspond to our selected concepts improves performance.

Table 3.2: Intent Prediction Performance

Training Method	Intent Prediction Accuracy
Without CW	73.0%
CW	**84.0%**
CW + Transfer	**84.1%**

3.3.6 Concept Ablation

We also tested the effect of concept selection on performance (Table 3.3). In particular, we omitted the knowledge condition (KC) concepts and/or the mission timer concepts, tested concept-whitened ToM models both with and without transfer, and found noticeably diminished performance.

Compared to the non-CW model, CW with reduced concept sets resulted in worse performance, and while transfer from the non-CW model somewhat mitigated this effect, we still see a significant drop from the performance with full concept set. This demonstrates the importance of good concept selection for the resulting performance of concept-whitened ToM model.

Table 3.3: Varying Concept Sets

Training Method	Concept Set	Acc.
Without CW	N/A	73.0%
CW	III	41.2%
CW	II	69.2%
CW	**I**	**84.0%**
CW + Transfer	III	54.9%
CW + Transfer	II	77.9%
CW + Transfer	**I**	**84.1%**

3.4 Interpretability Analysis

Generally, using a deep model for a task such as intent inference makes interpreting model output difficult due to the "black-box" nature of neural networks. Adding the concept whitening module provides us with a direct mechanism that allows us to interpret the model output in terms of our chosen concepts. For each observation and corresponding intent prediction made by our ToM model, we can calculate the importance of each concept.

For example, if our ToM model observes that a player hears a beep and predicts the player intends to bypass a particular door, we can examine the relative importance of concepts such as "critical victim in field of view" or "no knowledge of the beep signal's meaning" or "over 8 minutes elapsed in mission timer" for our model's prediction. If for instance the most important concept is "no knowledge of beep signal's meaning", an appropriate intervention could be to inform the player of the signal's meaning.

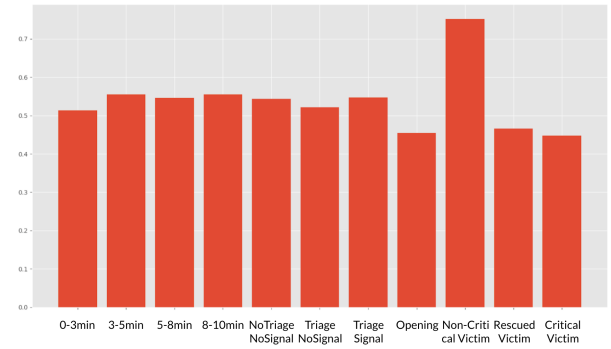
We can estimate the concept importance for each prediction via the activation for each column of the CW orthogonal matrix $\mathbf{Q}$, given by:

$$a_j = \mathbf{q}_j^\top \hat{\mathbf{z}}_b \tag{3.5}$$

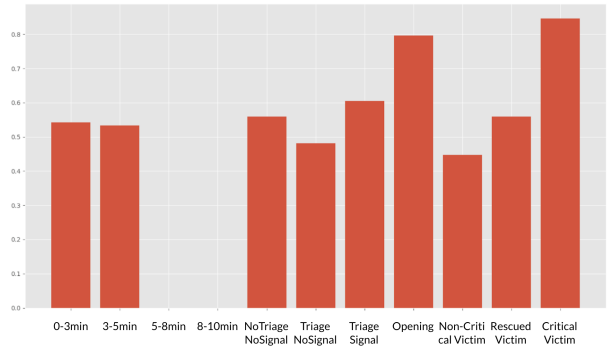
where $\hat{\mathbf{z}}_\mathbf{x}$ is the concept-whitened latent representation for belief state b.

We can examine the activation vectors $\mathbf{a} = [a_1 \dots a_k]$ for different types of intent predictions by the learned model (CW + transfer, full concept set). The mean normalized activations for non-critical victims, critical victims, and doors / openings are visualized in Fig. 3.4.

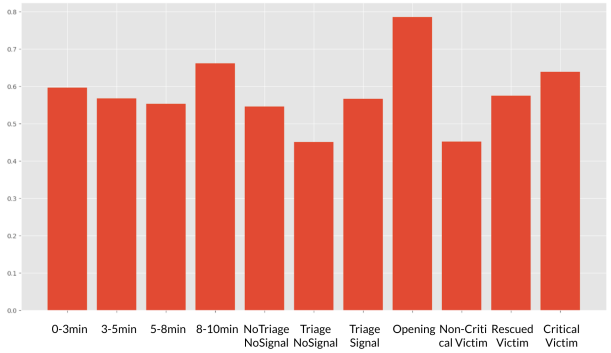
These largely line up with intuition; unsurprisingly, the presence of an intent-relevant entity in the field of view is an important concept for the model's prediction of said intent. We also see variability in the importance of different mission time intervals and of knowledge condition for different intent predictions.



(a) Mean concept activation for intent prediction of **non-critical victim**. We can see that the presence of a non-critical victim in the field-of-view is the most activated concept.



(b) Mean concept activation for intent prediction of **critical victim**. Here we see zero activation for mission timer above 5 minutes (which corresponds with critical victims expiring). We also see that the presence of a critical victim or room opening in field-of-view is a common reason for predicting intent to triage a critical victim.



(c) Mean concept activation for intent prediction of **opening**. The presence of an opening in the field of view is the most highly activated concept. We also see that compared to the other mission timer concepts, the last 2 minutes sees the timer become a more important reason for predicting intent to go towards a door or opening.

Figure 3.4: Mean concept activations for different intent prediction types.

Chapter 4

Concept-Based Reinforcement Learning

We now move to *embodied* theory of mind in agents. We more broadly examine *concept learning* as a general way to approach the challenge of grounding semantically meaningful *mental states* within policies, and use this approach to model theory of mind within policies.

4.1 Reinforcement Learning with Concepts

In deep reinforcement learning, policies are typically black-box neural networks that directly map states to actions. Our approach follows the paradigm of concept learning, which involves mapping or aligning the latent space of a model to a set of human-interpretable concepts (e.g. “door is locked", “agent has key") [Schwalbe, 2022].

We present an approach to learning interpretable deep policies that make decisions directly in terms of these concepts. Here we build on the concept-bottleneck model approach [Koh et al., 2020], applying it to reinforcement learning, and extending it to incorporate residual state information while maintaining model interpretability.

4.1.1 Framework

We model each multi-agent scenario as a *stochastic game* [Shapley, 1953]. A stochastic game can be defined as a tuple $\langle N, \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ where $N \in \mathbb{N}$ is the number of agents, $\mathcal{S}$ is the state space shared by all agents, $\mathcal{A} = A_1 \times ... \times A_N$ is the joint action space, $\mathcal{P} \colon \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition function, $\mathcal{R} = \{r_i \colon \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}\}$ indicates the reward functions for each agent, and $\gamma \in [0, 1]$ is the discount factor.

The stochastic game framework describes a multi-agent process with simultaneous agent actions, where a *trajectory* takes the form $\tau = (s^{(0)}, \mathbf{a}^{(0)}, s^{(1)}, \mathbf{a}^{(1)}, \dots)$ where $s^{(t)}$ is the state at time t and $\mathbf{a}^{(t)} = [a_1^{(t)}, \dots, a_N^{(t)}]$ denotes the action taken by each agent at time t.

Definition 1 (Concept)**.** *A concept is a variable C that takes on a value for each timestep in a trajectory and corresponds to some semantically-meaningful property. Formally, a concept can*

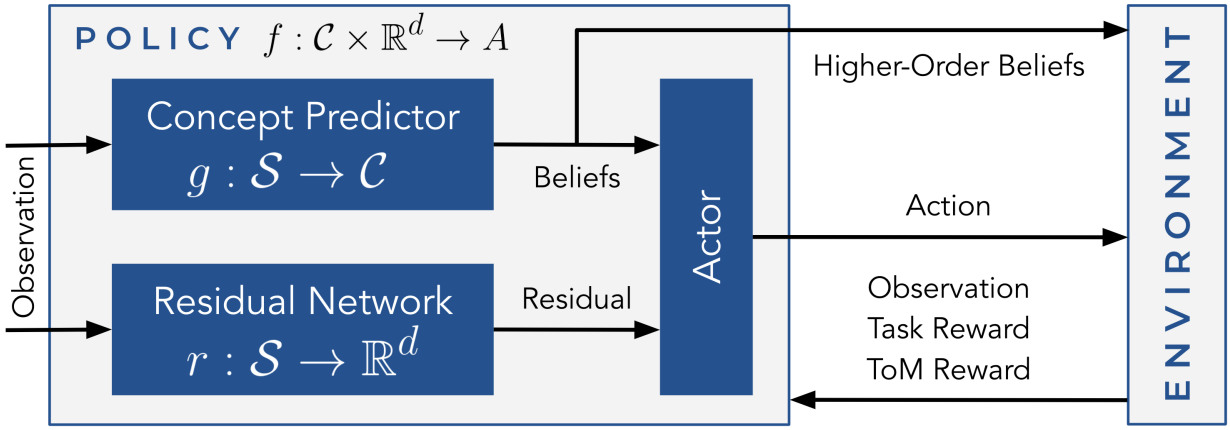


Figure 4.1: Concept-residual policies with higher-level ToM inference. Ground truth concept values supervise the concept predictor, and the residual network is regularized via mutual information minimization with respect to concept values.

be defined as a measurable function $C \colon \mathcal{T} \times \mathbb{N} \to \Omega$ *where* $\mathcal{T}$ *is the set of possible trajectories and* Ω *is the set of possible concept values.*

Concepts may be discrete or continuous. For example, we can model an agent's distance from some goal location as a discrete concept $C_{distance}(\tau, t) = c$ taking on values $c \in \{close, far\}$, and we can model an agent's 2D displacement from the goal location as a continuous concept $C_{displacement}(\tau, t) = c$ taking on values $c \in \mathbb{R}^2$.

4.1.2 Concept-Residual Policies

We consider policies of the form $f(g(s), r(s))$, where $r \colon \mathcal{S} \to \mathbb{R}^d$ maps states to some latent residual representation, $g \colon \mathcal{S} \to \mathcal{C}$ maps states into concept space (e.g. "door is locked", "agent has key"), and $f \colon \mathcal{C} \times \mathbb{R}^d \to A$ maps concept values and residual representations into action space (e.g. "move forward"). This compositional network structure is illustrated in figure 4.1.

Intuitively, we want these policies to make inferences about specific concepts in the environment and make decisions based on these concepts, using residual information only when necessary.

Learning Concepts

Assume that at training time we are given access to ground truth concept values corresponding to each encountered state in our trajectories. Then with each state s we can supervise a concept predictor $g(s) = \hat{\mathbf{c}}$ directly on these ground truth concept values:

$$\mathcal{L}_{concept} = \begin{cases} \text{MSE}(\mathbf{c}, \hat{\mathbf{c}}) & \text{if continuous} \\ \text{CE}(\mathbf{c}, \hat{\mathbf{c}}) & \text{if discrete} \end{cases} \tag{4.1}$$

where $\mathbf{c}$ is a vector of ground truth concept values, $\hat{\mathbf{c}}$ a vector of predicted concept values, MSE is the mean-squared error, and CE is the cross entropy loss. These predicted concept values are then used to generate an action.

Learning Residuals

The benefit of the bottleneck approach is that it induces interpretability by constraining the action to be determined entirely through predicted concept values (i.e. $\mathbf{c} = g(s)$, $a = f(\mathbf{c})$).

However, depending on the selection of concepts, they alone may not be a sufficient signal to learn a policy that successfully solves a given task. For instance, if an agent infers the concept value “door is open", it would still need additional spatial information to effectively navigate to the door.

We mitigate this by introducing a *residual* – a filtered, compressed representation of the input that is concatenated to our concept values to provide any “extra information" needed to determine a suitable action. This representation is learned via a neural network $r(s) = \mathbf{z}$ that generates residual representation z given state s.

As the purpose of the residual is to act as a signal to help maximize the expected reward on the task, the residual and actor networks are optimized jointly; during training, any policy loss is backpropagated through both the actor and residual networks, while concepts are trained in a supervised manner solely through the ground-truth signal $\mathbf{c}$.

Disentangling Concepts and Residuals

While adding unsupervised concept dimensions or residual information to concept-based models has been demonstrated to improve performance, it can come at the cost of interpretability [Mahinpei et al., 2021]. Models will tend to encode information about concept values and concept distributions within the residual, a phenomenon known as “concept leakage". This compromises the interpretability of the model, as the concept predictions are no longer meaningful explanations of the model's true decision-making process.

To address this issue, we introduce the constraint that our residual and concept values must be *disentangled* – that is, the residual should not contain any information about the concepts. We approach disentanglement from a probability theory perspective, where the aim is to ensure that concepts and residuals are statistically independent. This can be realized via optimization by minimizing some measure of dependence between concept values and generated residuals, with respect to residual network parameters.

Although the most common measure of statistical dependence is Pearson's correlation coefficient, it has some significant limitations; in particular, it only captures linear relationships, and thus zero correlation does not necessarily imply independence [Tjøstheim et al., 2022].

Instead we use *mutual information*, which is zero between random variables *if and only if* they are independent. This measure can also be characterized as KL-divergence between the joint distribution and the product of the marginal distributions:

$$I(C; Z) = D_{KL}(\mathbb{P}_{CZ} \parallel \mathbb{P}_C \otimes \mathbb{P}_Z) \tag{4.2}$$

To minimize this quantity, we utilize the variational approach from [Cheng et al., 2020] and minimize a contrastive log-ratio upper bound. Given policy parameters σ and variational parameters

Algorithm 4: Training concept-residual policies

```
input:
    MyRLAlgorithm # (e.g. PPO)
    concept_loss_fn # (CE or MSE)
    f, g, r, q # neural networks
    α, β, ϵ # hyperparameters

for batch in MyRLAlgorithm.train():
    s, c = batch.states, batch.concepts
    ĉ = g(s) # predicted concepts
    z = r(s) # residual
    L_q = −log q(z | c).mean()
    Update variational network q to minimize L_q
    L_concept = concept_loss_fn(c, ĉ)
    pos = log q(z | c)
    neg = log q(z | shuffle(c)) # batch-wise negative (product of marginals)
    L_residual = (pos − neg).mean()
    c̃ = add_noise(c, ϵ) # noisy concepts
    a = f(c̃, z) # actions
    L_policy = MyRLAlgorithm.loss(s, a)
    L_CRP = L_policy + αL_concept + βL_residual
    Update networks f, g, r to minimize L_CRP
```

θ, we have the following:

$$\mathcal{L}_q(\theta) = -\mathbb{E}_{p_\sigma(\mathbf{c},\mathbf{z})}[\log q_\theta(\mathbf{z} \mid \mathbf{c})] \tag{4.3}$$

$$\mathcal{L}_{residual}(\sigma) = \mathbb{E}_{p_\sigma(\mathbf{c},\mathbf{z})}[\log q_\theta(\mathbf{z} \mid \mathbf{c})] - \mathbb{E}_{p_\sigma(\mathbf{c})}\mathbb{E}_{p_\sigma(\mathbf{z})}[\log q_\theta(\mathbf{z} \mid \mathbf{c})] \tag{4.4}$$

where $\mathbf{c}$ is the concept vector, $\mathbf{z}$ is the residual vector, p_σ represents the distribution of concepts and residuals induced by the current policy in the environment, and $q_\theta(\mathbf{z} \mid \mathbf{c})$ is a variational approximation to the conditional distribution $p_\sigma(\mathbf{z} \mid \mathbf{c})$, modeled via a separate neural network trained to minimize the negative log-likelihood $\mathcal{L}_q$.

Learning the Task

Given concept values and the generated residual, the last link in the chain is the actor network $f(\mathbf{c}, \mathbf{z}) = a$. The tuple $(\mathbf{c}, \mathbf{z})$ acts as an interpretable, disentangled state representation, while our actor takes these state representations and produces actions aimed to maximize expected reward from the environment.

The only constraint of our approach with respect to learning is the need for ground truth concept values to supervise the concept predictor while learning the policy. If concept labels are given or computable, this makes it straightforward to train these agents with essentially any model-free deep reinforcement learning optimization algorithm; in policy-based and actor-critic methods f models the policy and the actor respectively, and in value-based methods f models the estimated value function or Q-function. Given a selected RL algorithm, we adopt the corresponding policy

loss (e.g. $\mathcal{L}_{policy} = \mathcal{L}_{PPO}$). Then the augmented concept-residual policy loss becomes:

$$\mathcal{L}_{CRP} = \mathcal{L}_{policy} + \alpha\mathcal{L}_{concept} + \beta\mathcal{L}_{residual} \tag{4.5}$$

where $\alpha, \beta > 0$ are hyperparameters.

During training, for each batch we optimize both the augmented policy loss $\mathcal{L}_{CRP}$ (with respect to the policy parameters σ) and the variational loss $\mathcal{L}_q$ (with respect to the variational parameters θ). We also note that we use the *independent* training paradigm from the bottleneck approach [Koh et al., 2020], using (noisy) ground-truth concepts to train our actor network, and using concept predictions at inference time. Pseudocode for this training process is provided in Algorithm 4.

4.2 Theory of Mind with Concepts

Learning concept-residual policies that are interpretable by design provides tremendous flexibility in the types of inductive bias we can impose on our agents. In particular, we can use concepts to model theory of mind.

4.2.1 Beliefs and Intents

Concept-residual policies expose internal "mental states" in the form of their concept predictions, which can be interpreted as *beliefs*.

Definition 2 (Belief)**.** *A belief over a concept is a prediction of the concept value for given state, defined as a probability distribution over possible concept values.*

Recall that our concepts are defined with respect to entire trajectories, rather than individual states. This allows for "post-hoc" concepts that correspond to a property of the current timestep but are determined by future portions of the trajectory. In particular, if we have a concept that corresponds to *future outcomes that can be influenced by the agent's behavior*, then the agent's belief about the outcome of its future behavior can be interpreted as *intent* (i.e. "I believe my actions will result in X).

Intuitively, intents are treated as a special case of belief where the prediction is about future states. For example, in an environment with multiple rooms, one could define a concept as "the next room the agent will enter after time t". Although the ground truth concept value is not yet determined at time t, the agent's *belief* at time t about which room it will enter next is considered the *intent*.

Definition 3 (Intent)**.** *Let $C : \mathcal{T} \times \mathbb{N} \to \mathscr{P}(\mathcal{S})$ be a concept such that $C(\tau, t) \cap F_{\tau,t} \neq \emptyset$ for all $\tau \in \mathcal{T}$ and for all $t \in \mathbb{N}$, where $\mathcal{T}$ is the set of possible trajectories, $\mathscr{P}(\mathcal{S})$ is the powerset of the state space, and $F_{\tau,t} = \{s^{(t+1)}, s^{(t+2)}, ...\}$ denotes the set of future states in trajectory τ after time t. Then an agent's belief over C is an intent.*

4.2.2 Recursive Reasoning

Concepts provide us a flexible framework with which we can model and ground beliefs and intents within policies. Assuming the policy's decision-making process is truly interpretable with respect to reasoning over these beliefs and intents, this can be considered zero-order theory of mind [Hedden and Zhang, 2002].

In a multi-agent scenario where each agent makes predictions over the same set of concepts, consider the idea of one agent predicting the value of another agent's concept predictions (i.e. "I believe that you believe X"). Similarly, agents may also make predictions about the 2nd-order predictions of other agents (i.e. "I believe that you believe that they believe X"). This type of recursive reasoning over beliefs can be extended to any arbitrary depth. An agent reasoning over n-th order beliefs is considered to have an n-th order theory of mind.

Definition 4 (n-th Order Belief)**.** *Let zero-order belief refer to an agent's belief over ground truth concepts as defined in Definition 2. Then an n-th order belief is defined as one agent's belief over another agent's $(n-1)$-th order belief.*

Intuitively, a concept can be understood as a mapping from each *(trajectory, timestep)* pair to an element from a set of possible concept values. For a "ground truth" binary concept such as "*door is open*", it is a function where the set of possible concept values is $\{0, 1\}$. An agent's belief over this concept is a Bernoulli distribution over the discrete domain, parameterized by a single continuous value $p \in [0, 1]$.

However, said **agent's belief over this ground truth concept can also be a concept in itself**, where the set of possible concept values is the set of possible Bernoulli distributions (i.e. the continuous interval $[0, 1]$). Then another agent's belief over this "higher-order concept" is a probability distribution over elements of this new continuous domain. For simplicity, we just use regression to directly predict the concept value, in which case the distribution is a delta distribution, also parameterized by a single continuous value $p \in [0, 1]$.

It is important to note that the belief of an agent may be incorrect, in which case a correct higher-order belief would successfully predict this false belief. For instance, consider a scenario where a door is locked but agent i believes the door is unlocked. Agent j should ideally have 1) the zero-order belief that the door is locked, and 2) the first-order belief that agent i thinks the door is unlocked.

In the vanilla concept-residual policy, the concept prediction network $g\colon \mathcal{S} \to \mathcal{C}$ models zero-order beliefs. To model n-th order beliefs over K agents, we can simply use a modified concept prediction network $g\colon \mathcal{S} \to \mathcal{C}^{K^n}$, where $\mathcal{C}^{K^n}$ is the K^n-th Cartesian power of concept space $\mathcal{C}$.

Now for each agent i we have n-th order beliefs $g_i(s) = \hat{\mathbf{C}}^{(i)}$ as a tensor of rank n, where $\hat{\mathbf{C}}^{(i)}_{j,\dots}$ is agent i's prediction of agent j's $(n-1)$-th order belief. Let $\mathbf{C}^{(i)}$ be the n-th rank tensor given by:

$$\mathbf{C}^{(i)}_{j,\dots} = \begin{cases} \mathbf{c} & i = j \\ \hat{\mathbf{C}}^{(j)}_{j,\dots} & i \neq j \end{cases} \tag{4.6}$$

where $\mathbf{c}$ is a (broadcasted) $(n-1)$-th rank tensor of ground truth concept values. Then the concept loss term in Equation 4.5 becomes:

$$\mathcal{L}_{concept} = \ell_c \left(\mathbf{C}^{(i)}, \hat{\mathbf{C}}^{(i)} \right) \tag{4.7}$$

where ℓ_c is the relevant loss function (see Equation 4.1).

4.2.3 Higher-Order Belief Example

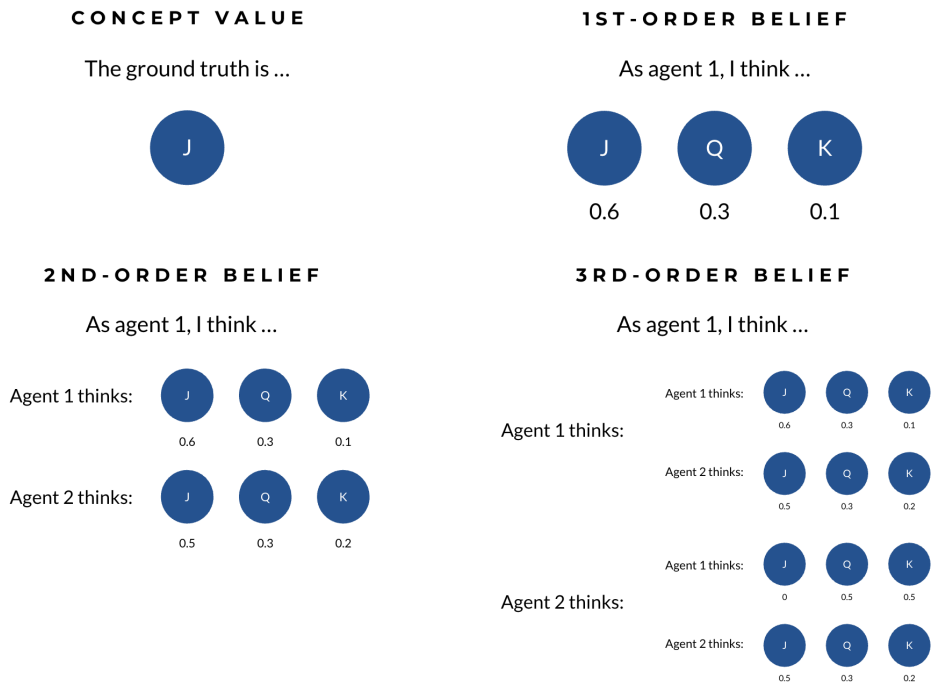


Figure 4.2: Visualization of higher order beliefs. Here we have a discrete concept representing ground truth value of an unknown card: **J**. From the perspective of agent 1: "I believe the unknown card is probably **J** (1st-order belief), that Agent 2 is a little less certain (2nd-order belief), but that Agent 2 thinks I don't believe it is **J** at all (3rd-order belief).

Intuitively, we use concepts to model information that may be unknown to agents. For example, in a card game we can model the rank of the next card in the deck as a discrete concept $C_{rank}(\tau, t)$ taking on a *concept value* $c \in \Omega = \{A, 2, 3, \ldots, J, Q, K\}$.

Intuitively, an agent's belief is a *prediction* of the concept value given the information known to it at time t. Consider the previous example of the discrete concept C_{rank}. Given the current environment state $s_\tau^{(t)}$, if an agent generates a distribution parameterized as a vector of probabilities for each rank $f(s_\tau^{(t)}) = [p_A, p_2, p_3, \ldots, p_J, p_Q, p_K]$, then at each state we can consider $b = f(s_\tau^{(t)})$ to be the *agent's belief*.

We can also extend this idea to higher-order theory of mind reasoning. Returning to the card game example, an agent may maintain a 1st-order belief about the rank of the next card in the

deck. However, a skilled player may also infer their opponents' beliefs about that card – this is a 2nd-order belief. Extending this further, an agent may attempt to model what others believe about its own belief (3rd-order belief) and so on, leading to an implicit hierarchy of nested reasoning (see Figure 4.2).

In our framework, we first define a collection of relevant concepts for an environment. Then each agent policy has a corresponding belief network that takes in observations and generates concept value predictions at each timestep. For every batch of experiences used for training the agent policy, the corresponding ground truth concept values are used to supervise the agent's belief network (see Algorithm 4).

Again, consider the example of the discrete concept C_{rank} with n possible concept values. If each agent i generates a 1st-order belief as a generalized Bernoulli distribution parameterized by $b_i \in [0, 1]^n$, we can aggregate these belief parameters into a matrix $B = [b_1, b_2, \ldots, b_m]^\top$. Then the 2nd-order belief of each agent i is a parameterized probability distribution over aggregated 1st-order beliefs B. Similarly, we can aggregate 2nd-order belief distribution parameters and generate 3rd-order beliefs distributions over these, and so on.

In this particular work, to reduce the number of parameters needed for higher orders, when $k > 1$ we model each k-order belief as a joint distribution over independent $(k-1)$-order beliefs, parameterized directly by the marginal distribution parameters (i.e. the aggregated $(k-1)$-order belief distribution parameters). However, this can be adjusted to incorporate more complex joint distributions if more expressivity is needed.

4.2.4 Bounded Recursion

The psychology literature suggests that humans tend to reason with 1st or 2nd-order theory of mind [Camerer et al., 2004], and studies suggest limited benefit beyond 2nd-order reasoning on most tasks [De Weerd et al., 2013a,b, 2017]. For this reason, and given that the dimensionality of the concept prediction space grows exponentially with respect to the level of ToM reasoning, the main body of this work considers up to a maximum order of $n = 2$.

4.2.5 Centralized Training

To supervise the predictions of a higher-order policy, we need access to relevant predictions from all of the other policies, which requires centralized training. Centralized training with decentralized execution (CTDE) has become a widely adopted framework in MARL, particularly in actor-critic methods where a centralized critic network has access to observations from all agents [Sharma et al., 2021].

Centralized ToM training can be understood along similar lines; at training time, each n-th order prediction is supervised by the $(n-1)$-th order prediction of the relevant agent. At test time, the policy relies on its own predictions and does not require any access to ground-truth internal states of other agents, allowing execution in multi-agent scenarios with any other types of agents, including human actors.

We note a potential concern with centralized ToM training in competitive scenarios is that a malicious agent may intentionally learn incorrect beliefs to confound other agents. However we note that, as previously mentioned, we utilize the independent training paradigm from [Koh et al., 2020]. This means that gradients resulting from generated actions are not backpropagated through the concept prediction network, leaving beliefs to be supervised only by ground truth and unaffected by any reward, avoiding the issue.

Chapter 5

Cooperation

We now investigate using concept-residual policies as described in the previous chapter to model theory of mind in cooperative and partially-cooperative multi-agent scenarios.

5.1 Theory of Mind as Intrinsic Motivation

In addition to introducing higher-order ToM inference as a directly supervised auxiliary prediction task, we also propose using it as an intrinsic reward.

As previously described, the concept predictor is trained independently from the residual and actor networks, and is supervised solely by $\mathcal{L}_{concept}$ (Equation 4.7). While adding this term to the overall loss incentivizes an agent to predict others' beliefs and intents *at each current state*, treating it as an intrinsic reward incentivizes an agent to act in a way such that others' beliefs and intents will be predictable *in the future*. The idea here is to bias exploration towards states and behaviors where agents can maintain or gain knowledge about other agents' internal mental states (including their higher-order beliefs), in a way that also able to achieve high task reward. For example, as an agent in a cooperative task I may be incentivized to communicate effectively to maintain knowledge of teammates' intentions. As an agent in an adversarial scenario, however, I may be incentivized to consistently confuse my opponent about my true beliefs and intentions (e.g. behave in such a way that my opponent consistently has high entropy predictions of my beliefs, thus allowing me to simply predict a uniform distribution as my second-order prediction of their prediction).

The intrinsic reward for each agent is constructed similarly to the loss term in Equation 4.7. Given K agents, let $\mathbf{Y} = \left[\hat{\mathbf{C}}^{(1)}_{1,\dots} \;\; \hat{\mathbf{C}}^{(2)}_{2,\dots} \;\; \dots \;\; \hat{\mathbf{C}}^{(K)}_{K,\dots}\right]$. Then the reward becomes:

$$r_{tom} = -\ell_c \left(\mathbf{Y}, \hat{\mathbf{C}}^{(i)}\right) \tag{5.1}$$

$$r = r_{task} + \lambda r_{tom} \tag{5.2}$$

where ℓ_c is the relevant concept loss function (see Equation 4.1), and $\lambda \geq 0$ is a hyperparameter that controls the balance between the intrinsic (ToM) and extrinsic (task) rewards.

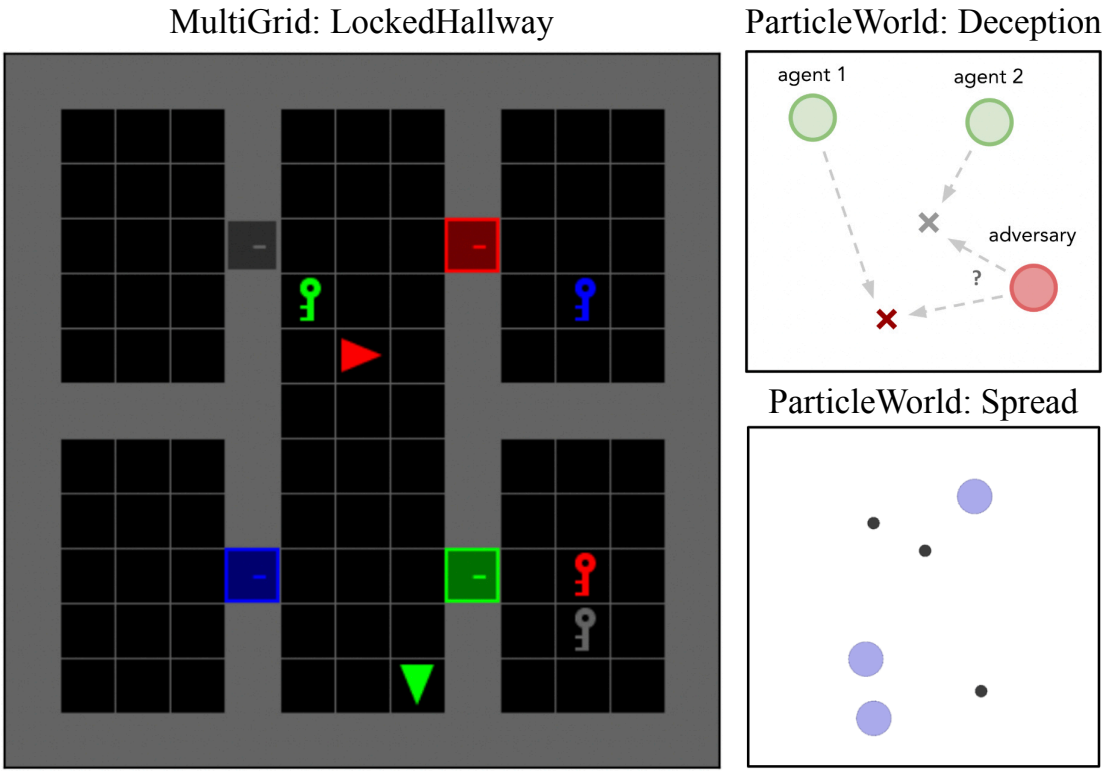


Figure 5.1: Environments used for experiments. The deception task is a mixed cooperative-competitive scenario, while the spread and locked hallway tasks are fully cooperative.

5.2 Experiments

We evaluate the performance of 0th, 1st, and 2nd order concept-residual policies on a selection of grid world and particle world environments.

As this approach can be integrated with other reinforcement learning optimization algorithms, we consider actor-critic methods A2C [Mnih et al., 2016], PPO [Schulman et al., 2017], and IMPALA [Espeholt et al., 2018] implemented with centralized critics, and Rainbow DQN [Hessel et al., 2018] with independent learners. We also consider QMIX for our cooperative environments [Rashid et al., 2020].

Within each environment, we use identical neural architectures for each concept-residual policy configuration, except for the output dimension of the concept prediction network, which is given by $d_c \cdot k^n$, where d_c is the ground truth concept vector dimension, k is the number of agents, and n is the ToM order of the concept-residual policy). We also compare to a purely neural policy baseline without concepts (i.e. using residual only with concept output dimension 0). Baseline and zero-order models have no intrinsic reward, and we note that for 1st-order and above our hyperparameter search includes $\lambda = 0$ (i.e. both with and without intrinsic reward).

For each environment, algorithm, and configuration we perform a hyperparameter grid search and report the results for the best set, averaged across 5 random seeds and trained for 10 million

timesteps each.

5.2.1 ParticleWorld: Deception

We use a modified version of the physical deception task described in [Lowe et al., 2017]. This environment consists of N landmarks, N "good" agents and a single red adversary agent within a 2D world; for our experiments we set $N = 2$.

In our variant, one of the landmarks is the "target", but neither the good agents nor the adversary are initially told which one. The N good agents receive a joint reward based on the minimum distance to the target landmark, with each agent's contribution weighted by a randomly generated reward coefficient $\eta_i \sim \text{Uniform}[0, 1]$. Similarly, the adversary is penalized based on its distance from the target.

The episode ends either after a fixed time-limit, or when the adversary reaches any landmark. If this is the target landmark, the adversary receives a positive reward, otherwise a negative penalty (both time–scaled).

$$\begin{aligned} r_{good}(t) &= -\min_i \{d(\mathbf{x}_{i,t}, \mathbf{x}_{target})\} \\ &\quad + d(\mathbf{x}_{adv}, \mathbf{x}_{target}) \end{aligned} \tag{5.3}$$

$$\begin{aligned} r_{adv}(t) &= -d(\mathbf{x}_{adv}, \mathbf{x}_{target}) \\ &\quad - \mathbb{I}[\mathbf{x}_{adv} = \mathbf{x}_{other}](1 - t/T) \\ &\quad + \mathbb{I}[\mathbf{x}_{adv} = \mathbf{x}_{target}](1 - t/T) \end{aligned} \tag{5.4}$$

where d is Euclidean distance, $\mathbf{x}_{target}$ is the position of the target, $\mathbf{x}_{other}$ is the position of the non-target landmark, $\mathbf{x}_{i,t}$ is the position of good agent i at time t, $\mathbf{x}_{adv,t}$ is the position of the adversary agent at time t, and $T = 50$ is the maximum episode length.

The adversary is incentivized to find and navigate to the target as quickly as possible. On the other hand, the good agents are incentivized to keep the adversary uncertain as long as possible while accumulating reward.

Concepts Agents reason over the following concepts:

- Whether each landmark is the target (*belief*)
- Reward coefficient for each good agent (*belief*)

In each configuration we train one policy for the good agents and one policy for the adversary agent, varying the level of ToM inference (i.e. 0th order, 1st order, etc). We evaluate by matching each configuration against baseline opposition (e.g. 1st-order good agents vs. baseline adversary, 1st-order adversary vs. baseline good agents, etc); the results are reported in Table 5.1. We find that agents are consistently better performing against baseline opponents as ToM order increases; this phenomenon is observed for both the good agents and the adversary.

Good Agents				
Algorithm	No ToM	ToM-0	ToM-1	ToM-2
A2C	1.876	2.099	**2.635**	**2.930**
PPO	1.889	2.314	**2.574**	**2.943**
IMPALA	1.721	2.180	**2.678**	**2.908**
DQN	1.918	2.239	**2.563**	**2.725**
Adversary Agent				
Algorithm	No ToM	ToM-0	ToM-1	ToM-2
A2C	-15.60	-15.26	**-13.73**	**-12.77**
PPO	-15.32	-15.01	**-13.95**	**-12.49**
IMPALA	-14.85	-14.73	**-13.13**	**-11.34**
DQN	-15.36	-14.99	**-13.81**	**-12.38**

Table 5.1: Performance against no-ToM opposition on *ParticleWorld: Deception* environment, in various configurations. We report the mean cumulative reward of the final trained policies, averaged across 5 random seeds and 1000 episodes each. ToM-n indicates n-th order policies (i.e. with n-th order beliefs). Bolded results are statistically significant improvements over baseline No ToM / ToM-0 (p-value less than 0.05).

5.2.2 ParticleWorld: Spread

We consider the cooperative navigation "spread" task described in [Lowe et al., 2017]. This environment consists of N landmarks and N agents within a 2D particle world; for our experiments we set $N = 3$. The goal is for the team of agents to learn to spread out and cover all the landmarks while avoiding collisions. Specifically, all agents receive a joint reward based on how far the closest agent is to each landmark (sum of the minimum distances), and are individually penalized for collisions.

Concepts Each agent reasons over the following concepts:

- Whether the agent will visit each landmark within 10 timesteps (*intent*)

Here for each n-th order configuration we train homogeneous agents with shared parameters. The mean episodic rewards (summed across all agents) achieved for each policy configuration are shown in Table 5.2. We observe a noticeable and consistent improvement with 1st and 2nd order policies compared to the 0th order and baseline models across all the algorithms used. The 2nd-order ToM tends to perform marginally better than the 1st-order ToM.

5.2.3 MultiGrid: LockedHallway

We further test our method on the locked hallway cooperative scenario in a multi-agent grid world environment [Oguntola, 2023]. This environment consists of two agents in a grid containing 6 locked rooms with colored doors, each with a corresponding color key. Agents receive a joint

Algorithm	No ToM	ToM-0	ToM-1	ToM-2
A2C	-77.11	-74.11	**-67.59**	**-66.30**
PPO	-75.13	-75.66	**-67.45**	**-67.08**
IMPALA	-75.67	-74.33	**-68.89**	**-67.01**
QMIX	-74.50	-75.47	**-67.96**	**-66.80**
DQN	-76.41	-77.03	**-68.86**	**-68.96**

Table 5.2: Performance on *ParticleWorld: Spread* environment, in various configurations. We report the mean cumulative reward of the final trained policies, averaged across 5 random seeds and 1000 episodes each. ToM-n indicates n-th order policies (i.e. with n-th order beliefs). Bolded results are statistically significant improvements over the baseline No ToM and ToM-0 (p-value less than 0.05).

Algorithm	No ToM	ToM-0	ToM-1	ToM-2
A2C	4.883	4.863	**5.295**	**5.401**
PPO	4.895	4.858	**5.317**	**5.339**
IMPALA	4.890	4.806	**5.334**	**5.418**
QMIX	4.872	4.852	**5.315**	**5.315**
DQN	4.888	4.849	**5.325**	**5.378**

Table 5.3: Performance on *MultiGrid: LockedHallway* environment, in various configurations. We report the mean cumulative reward of the final trained policies, averaged across 5 random seeds and 1000 episodes each. ToM-n indicates n-th order policies (i.e. with n-th order beliefs). Bolded results are statistically significant improvements over baseline No ToM / ToM-0 (p-value less than 0.05).

reward for unlocking and opening each door. The goal of the task is to open all the doors as quickly as possible.

Concepts Each agent reasons over the following concepts:

- Key that each agent currently has (if any) (*belief*)
- State of each door (i.e. open, closed, or locked) (*belief*)
- Whether the agent will visit each room within 20 timesteps (*intent*)

To bootstrap training in this environment we first trained a single agent (no-ToM) to perform the task, then transfer to the multi-agent setting by initializing all agents in our experiments from this pretrained model and finetuning with RL. Again, we train homogeneous agents with shared parameters. The mean episodic rewards achieved for each n-th order policy configuration are shown in Table 5.3. We find similar results to the particle world cooperative “spread" task, with significant improvement between 0th and 1st order policies, and additional marginal improvement between 1st and 2nd order policies.

5.2.4 Qualitative Analysis

Outside of increased performance in terms of reward, we find that a qualitative assessment indicates more complex and efficient strategies are exhibited by higher order policies, regardless of which RL optimization algorithm was used.

ParticleWorld: Deception

Baseline and zero order good agents move towards a unique landmark and tend to stay nearby, whereas 1st and 2nd-order good agents occasionally swap positions between landmarks. Baseline and zero-order adversaries appear to drift randomly throughout thte episode, while 1st-order and 2nd-order adversaries are more decisive, moving quickly to landmark.

ParticleWorld: Spread

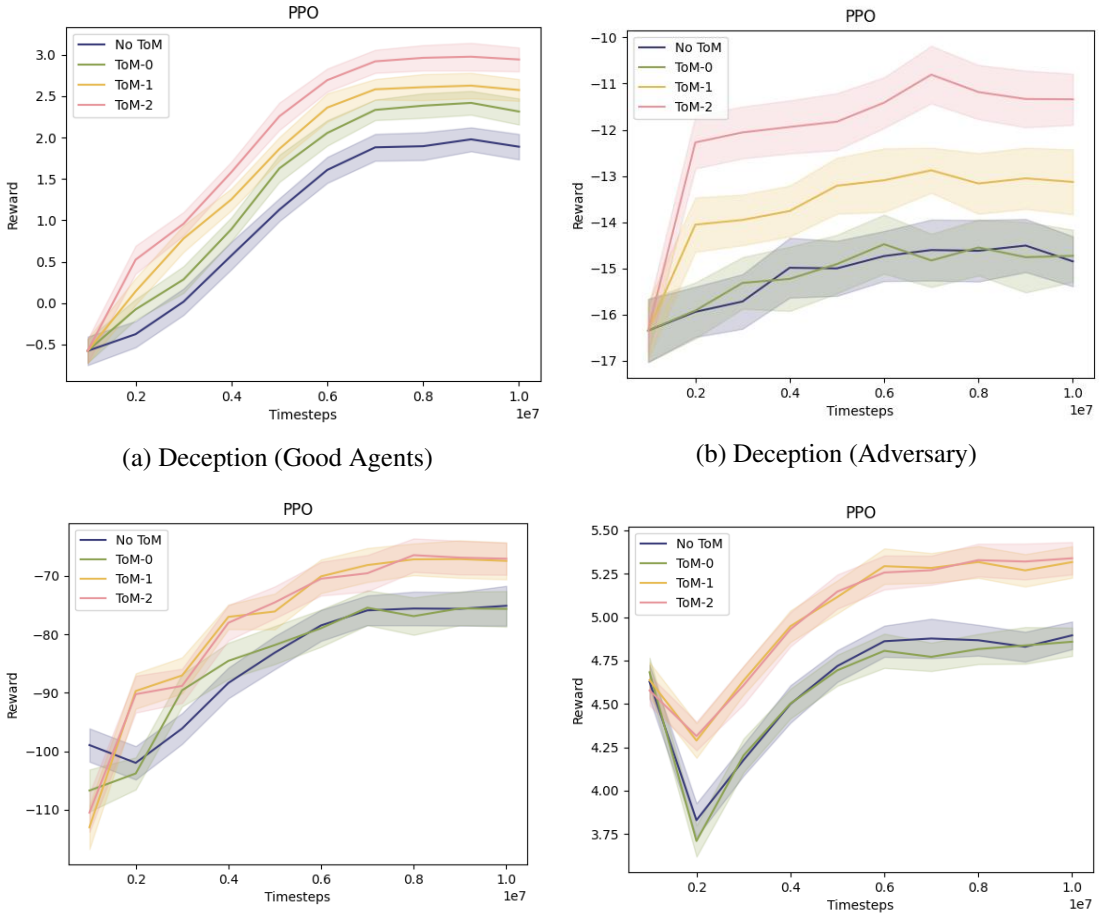
Baseline and zero-order policies can be described on a per-agent level as greedily moving towards closest landmark without any other agents nearby. With 1st-order and 2nd-order policies, agents are quicker to visibly switch targets.

MultiGrid: LockedHallway

Trajectories from baseline and the 0th-order policies often exhibited inefficient behavior due to overlapping intents, specifically both agents moving towards the same key to pick it up. Interestingly enough, this was particularly evident in the policies with zero-order ToM, which although were perhaps more equipped to handle individual taskwork, were also slightly worse at coordination. However with 1st and 2nd order policies, we observed a marked decrease in this type of behavior; in situations with only one available key, one agent heads for the key while the other tends to go directly to the corresponding door.

5.2.5 Takeaway

We consistently see a significant difference in the performance of 1st/2nd order ToM agents compared to 0th-order ToM or vanilla RL policies. We tend to to see diminishing to marginal returns moving from 1st to 2nd order agents.



(a) Deception (Good Agents)

(b) Deception (Adversary)

(c) Spread

(d) LockedHallway

Figure 5.2: PPO learning curves for agents in *ParticleWorld: Deception* environments. Plots show episode reward as evaluated against baseline agents, at various points during training. After every 1 million timesteps of training, we run 100 additional evaluation episodes and analyze the reward of the current policy. The curves show the mean reward at each stage of training, and the shaded portion for each curve shows a 95% confidence interval (2 standard deviations). All environments show a statistically significant improvement in 1st / 2nd-order ToM over 0th-order / baseline agents.

Chapter 6

Deception

We now turn our framework to specifically address deception in multi-agent scenarios. As artificial intelligence systems are increasingly deployed in environments where they must cooperate or compete with both human and artificial agents, the ability to reason about and influence the beliefs and actions of others becomes essential. This is particularly true in adversarial settings, where the ability to mislead or deceive an opposing agent can offer a decisive competitive advantage.

Deception—the act of causing another to believe something that is false—is a pervasive and fundamental aspect of many interactions, both in nature and human society. From camouflage in the animal kingdom to strategic misdirection in warfare and sports, deceptive behaviors have evolved as effective survival strategies. In artificial intelligence and machine learning, the ability to employ and recognize deception marks a significant step toward more sophisticated and human-like reasoning, enabling systems to act strategically in complex, dynamic environments.

Traditional MARL approaches have largely focused on optimizing direct rewards from the environment, often struggling to develop strategies that involve manipulating the beliefs or intentions of other agents. This limitation becomes especially apparent in games of incomplete information, where the ability to mislead opponents about one's state or intentions can be a key determinant of success. Games like *Kuhn Poker* and *Barrage Stratego* serve as prime examples of environments where deception plays a central role in gameplay, requiring agents to reason not only about their own decisions but also about the beliefs and strategies of their opponents [Kuhn, 1950, Federation, 2025].

To address this gap, we present a modification of our earlier approach where agents are not only trained to maximize environmental rewards but also to induce incorrect beliefs in their opponents. This is achieved by modeling the opponent's mental state and using the discrepancy between this model and reality as a reward signal. In turn, this enables agents to develop sophisticated deceptive strategies that extend beyond simple reactive behaviors, allowing them to mislead their opponents effectively.

Furthermore, we also extend this framework to higher-order ToM reasoning. In this extended

model, agents reason not only about their opponent's beliefs about the world (1st-order ToM) but also about their opponent's beliefs about the agent's beliefs (2nd-order ToM), and potentially even higher-order beliefs. This recursive reasoning allows for increasingly complex deceptive strategies, mirroring the depth of strategic thinking observed in high-level human gameplay. The ability to deceive an opponent's higher-order beliefs provides a clear advantage in environments like *Kuhn Poker*, where players must navigate incomplete information and make decisions based on the perceived actions and mental states of others.

Our empirical results show that agents trained with our higher-order deceptive framework consistently outperform both baseline MARL agents and those employing only first-order deceptive strategies. These improvements are evident not only in win rates but also in the sophistication and adaptability of the strategies used. The agents trained with higher-order ToM reasoning were able to consistently outmaneuver their opponents, making more informed and strategic decisions based on their understanding of the opponents' beliefs and intentions.

The implications of this work extend far beyond game-playing AI. As AI systems are increasingly deployed in complex, adversarial real-world scenarios—from cybersecurity to autonomous negotiation—the ability to reason about and influence the beliefs of other agents becomes critical. Our framework provides a step toward developing AI systems capable of engaging in these scenarios with human-like strategic depth.

Moreover, this research contributes to our broader understanding of deception and strategic reasoning. By developing AI systems that can both employ and recognize sophisticated deceptive strategies, we gain insights into the cognitive processes underlying such behaviors in humans and other intelligent entities.

6.1 Related Work

6.1.1 Deception in Multi-Agent Systems

The study of deception in multi-agent systems has garnered significant attention across various domains. Early work by [Castelfranchi, 1998] established foundational principles for modeling deception in artificial agents. [Arkin et al., 2011] later formalized these concepts within a decision-theoretic framework, providing mathematical groundwork for deceptive behaviors in autonomous systems.

Recent advances in deep learning have enabled more sophisticated approaches to deceptive behavior. The landmark achievement of Meta AI's CICERO [FAIR et al., 2022] showed that large language models could engage in natural language deception within the game of Diplomacy, highlighting the potential for sophisticated strategic behavior.

6.1.2 Game-Theoretic Approaches to Deception

Game theory provides formal frameworks for analyzing deceptive behavior. Ettinger and Jehiel [2010] developed theoretical foundations for modeling deception in games; recent work by Zhou

et al. [2022] has connected game-theoretic principles with deep learning approaches.

In poker-specific domains, Brown and Sandholm [2019] demonstrated superhuman performance in heads-up poker.

6.1.3 Deception in MARL

Deception in multi-agent reinforcement learning (MARL) has garnered significant attention as researchers strive to develop agents capable of strategic interactions that mirror complex human behaviors. Deception, in this context, involves agents deliberately providing misleading information or actions to manipulate the beliefs or behaviors of other agents to gain a strategic advantage.

MARL has seen substantial advancement in recent years in competitive scenarios. Zhang et al. [2021] provided a comprehensive survey of modern MARL techniques, while Hernandez-Leal et al. [2019] focused specifically on deep MARL approaches. Notable achievements include Vinyals et al. [2019]'s AlphaStar system and Berner et al. [2019]'s OpenAI Five.

The integration of opponent modeling in MARL has been explored by He et al. [2016] and extended by Raileanu et al. [2018]. Foerster et al. [2018] introduced learning with opponent-learning awareness (LOLA), while Wang et al. [2022] developed frameworks for cooperative multi-agent learning.

A notable advancement in this field is Meta AI's CICERO, the first AI agent to achieve human-level performance in the game of Diplomacy. Diplomacy is a strategy game that requires players to understand others' motivations, negotiate, form alliances, and sometimes employ deception to succeed. CICERO integrates natural language processing with strategic reasoning, enabling it to engage in complex negotiations and deceptive tactics to achieve its objectives [FAIR et al., 2022]. While CICERO's integration of language and strategy is impressive, our work focuses on environments where communication is implicit, and deception arises through actions rather than explicit dialogue.

In another study, Aitchison et al. introduced a mixed competitive-cooperative MARL environment inspired by role-based deception games like Werewolf and Among Us. This environment allows agents to learn deceptive behaviors by balancing cooperation and competition, providing insights into how deception can emerge and be sustained in multi-agent interactions [Aitchison et al., 2021]. Our approach differs by emphasizing the development of hierarchical policies that enable agents to dynamically adjust their level of deceptive behavior based on the evolving game context.

Furthermore, Chelarescu explored deception within the framework of social learning in MARL. The study highlights how agents can reshape the reward functions of other agents to promote cooperation but also warns of the increased risk of manipulation when agents are unaware of being deceived into adopting policies not in their best interest [Chelarescu, 2021]. In contrast, our work focuses on direct deceptive actions within competitive settings, without altering the underlying reward structures of other agents.

Additionally, Asgharnia et al. proposed a fuzzy multi-level reinforcement learning approach to model deception in pursuit-evasion games. They utilized a two-level policy system where the higher-level policy manages deception, and the lower-level policy controls goal-directed actions [Asgharnia et al., 2022]. While their hierarchical approach informs our methodology, our work uniquely applies this concept to complex strategic games, demonstrating the scalability and adaptability of hierarchical deception strategies.

These studies underscore the importance of deception in MARL, highlighting the need for agents to develop sophisticated strategies that include misleading opponents to achieve their goals. Our work contributes to this body of research by introducing a novel framework that combines hierarchical policy structures with dynamic deception strategies, enabling agents to effectively navigate and manipulate complex multi-agent environments.

6.2 Deception with Theory of Mind

6.2.1 Definitions

We begin with some explicit definitions.

Definition 5 (Deception)**.** ***Deception*** *involves an agent acting to deliberately induce false beliefs in other agents in order to gain a strategic advantage.*

It also is important to define what we mean by a "deceptive agent".

Definition 6 (n-Order Deceptive Agent)**.** *We call an agent* ***n-th order deceptive*** *if it both models and manipulates its opponents' n-th order beliefs.*

6.2.2 Learning Beliefs

Given a concept C, we assume that distributions over concept values are parameterized by d real values. Given m agents, each agent i maintains a belief network that processes observations and generates a n-order belief tensor $\mathbf{B}_i^{(n)} \in \mathbb{R}^m \times \cdots \times \mathbb{R}^m \times \mathbb{R}^d$ (i.e. what the agent's beliefs *actually are*).

For each agent i, we recursively define the n-order belief target tensor $\mathbf{T}_i^{(n)}$ (i.e. what the agent's beliefs *should be*):

$$\mathbf{T}_i^{(1)} = \mathbf{B}_i^{(1)} \tag{6.1}$$

$$\mathbf{T}_{i,j}^{(k)} = \begin{cases} \mathbf{T}_i^{(k-1)} & j = i \\ \mathbf{B}_j^{(k-1)} & j \neq i \end{cases} \tag{6.2}$$

Let $\ell(c, b)$ denote the log-likelihood of ground truth concept value c under a belief distribution parameterized by $b \in \mathbb{R}^d$, and let D be any statistical divergence measure between belief distributions.

Algorithm 5: Training n-order deceptive policies

```
input:
    ℓ # concept log-likelihood function
    D # belief divergence function
    f_1, ..., f_m # belief networks
    λ # deception reward weight

for batch in MAPPO.run():
    B = stack([f_i(batch.obs[i]) for i = 1 ... m]) # beliefs (*, m, ..., m, d)
    T = get_belief_targets(B) # belief targets (*, m, ..., m, d)

    # Train belief networks
    for i = 1 ... m:
        # Calculate belief loss
        B1 = diagonal(B[i]) # 1st-order belief (*, d)
        belief_losses[i] = -ℓ(batch.concepts, B1) + D(T[i], B[i]).sum() / m
            **(n-1)
        belief_losses[i].mean().backward()

    # Add deception intrinsic reward
    for i = 1 ... m:
        r = mean(belief_losses[j] for j in opponents[i])
        batch.rewards[j] += λ * r

    # Train agent policies
    MAPPO.train_on_batch(batch)
```

Then the belief loss for agent i is given by:

$$\mathcal{L}_{\text{belief}}\left(\mathbf{B}_i^{(n)}\right) = -\ell\left(c, \mathbf{B}_i^{(1)}\right) + \frac{1}{m^{n-1}} \sum_{\alpha \in \mathcal{I}} D\left(\mathbf{T}_{i,\alpha}^{(n)}, \mathbf{B}_{i,\alpha}^{(n)}\right) \tag{6.3}$$

where $\mathcal{I} = \{1, \cdots, m\}^{n-1}$ is the set of $(n-1)$ dimensional multi-indices and $\mathbf{T}_{i,\alpha}^{(n)}, \mathbf{B}_{i,\alpha}^{(n)} \in \mathbb{R}^d$ are slices of the order-n tensors at $(n-1)$ dimensional multi-index $\alpha \in \mathcal{I}$.

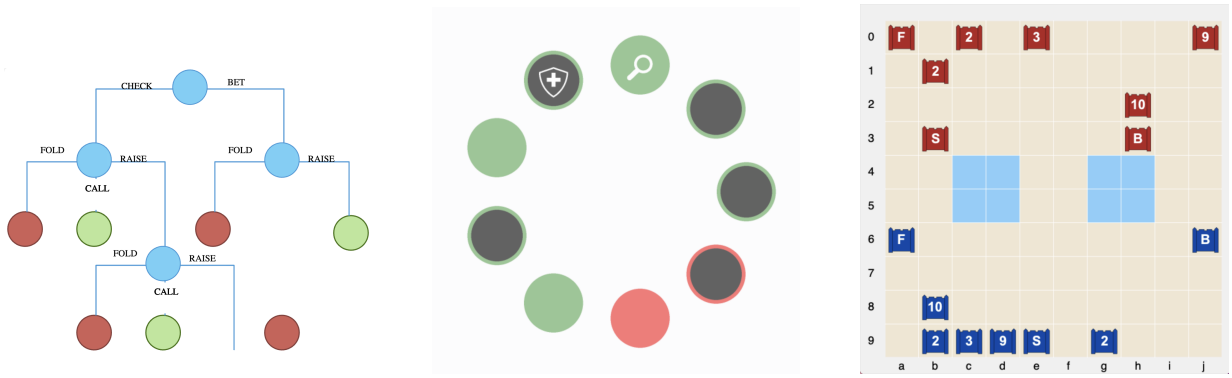
The first term encourages 1st-order beliefs match ground truth concepts, and the second term encourages higher-order beliefs to be consistent with lower-order beliefs.

In this particular work, we use categorical distributions with D as cross-entropy for discrete concept values, and multivariate Gaussian distributions with D as L2 distance between distribution means for continuous concept values.

As in previous sections, gradients from downstream action losses are not backpropagated into agent belief networks, which are trained solely via this supervised loss.

6.2.3 Deception as Intrinsic Reward

Modeling agent beliefs as probability distributions gives us a natural way to frame deception: inducing an agent's belief distribution to diverge from the ground truth. The belief loss $\mathcal{L}_{\text{belief}}(\mathbf{B}_i)$



(a) Kuhn Poker game tree (b) Mafia environment (c) Barrage Stratego board

Figure 6.1: Environments used for experiments.

from Equation 6.3 already provides us a natural way to quantify this.

For each deceptive agent, we propose adding an intrinsic reward for deception as the mean belief loss of its opponents:

$$r_{\text{deception}} = \frac{1}{|\mathcal{I}_{opp}|} \sum_{i \in \mathcal{I}_{opp}} \mathcal{L}_{\text{belief}}\left(\mathbf{B}_i\right) \tag{6.4}$$

$$r = r_{\text{task}} + \lambda r_{\text{deception}} \tag{6.5}$$

where $\lambda \geq 0$ is a hyperparameter.

Incorporating this intrinsic reward with higher-order beliefs creates the possibility for highly sophisticated deceptive and counter-deceptive strategies. For instance, an agent may not only attempt to deceive the opponent about hidden information (e.g. bluffing – 1st-order deception) but to also deceive the opponent about the agent's own beliefs (e.g. pretending to be deceived by an opponent's bluff – 2nd-order deception).

We also note that unless the agent's beliefs are used as an additional auxiliary part of the agent observation, the belief network is only needed for training and can be omitted during inference.

6.3 Experiments

We consider three partial-information games: Kuhn Poker, Mafia, and Barrage Stratego. We also provide code for the environments and our IPPO / MAPPO implementations at `https://github.com/ini/deception`.

6.3.1 Kuhn Poker

Kuhn Poker, a simplified version of standard poker, is a finite, zero-sum game of imperfect information [Kuhn, 1950]. We consider an n-player generalization with $n+1$ total cards $\{1, \ldots n+1\}$,

where each player is dealt a random card drawn uniformly at random, but has no knowledge of the other players' cards or of the remaining undealt card.

Kuhn poker serves as a fundamental testbed for deception due to its clear bluffing dynamics. Bluffing occurs when a player with a weak hand bets to induce a fold from a stronger opponent. Conversely, an opponent must reason about the likelihood of a bluff when deciding whether to call.

Gameplay This is a sequential game that cycles through player turns. Initially players may check or bet. Once any player has placed a bet, subsequent players must either call or fold. The game proceeds to a showdown if all players check or at least one calls; otherwise, the betting player wins uncontested. More specific environment implementation details can be found in the supplementary material.

Concepts In this environment we have n discrete concepts: the card rank for each of the n players. Each concept takes on a value $c_i \in \{1, \ldots, n+1\}$.

Architecture All policies are actor-critics with separate networks, implemented as MLPs with 2 hidden layers and 64 units per hidden layer, with the policy generating a discrete action: *bet* or *pass*.

Training Our particular configuration uses $n = 4$ agents. We train a k-order deceptive ego agent vs l-order deceptive opponents for $k = 0, 1, 2, 3$ and $l = 0 \ldots k$, where 0-order indicates a vanilla non-deceptive agent. Each environment is trained with two policies: one for ego agent, and one for its opponents. All policies are trained via independent PPO for 10 million environment steps Schulman et al. [2017]. Specific training hyperparameters can be found in the supplemental material. Results are shown in Table 6.1 and Table 6.2.

6.3.2 Barrage Stratego

Barrage Stratego is a simplified variant of the classic board game Stratego [Federation, 2025]. It is a two-player, zero-sum game of imperfect information played on a 10×10 board where each player knows the location, but not the rank, of its opponent's pieces.

Gameplay In the initial phase, players arranges their pieces on their side of the board, with piece ranks hidden from their opponent. Then players take turns, strategically maneuvering pieces and engaging in battles. Battles are resolved by revealing the ranks of the opposing pieces, with the higher-ranked piece winning unless special rules apply. The game ends when one player captures the other player's *flag* piece.

Concepts In this environment we have 100 discrete concepts: the rank of the piece at each cell. Each concept takes on a value $c_i \in \{\text{empty}, \text{flag}, \text{bomb}, \text{marshal}, \text{general}, \text{miner}, \text{scout}, \text{spy}\}$.

Stratego and its variants provide a structured setting to study planning under uncertainty, opponent modeling, and the use of misinformation as a tactical tool. As the hidden ranks of each piece remains unknown to the opponent until revealed in battle, the ability to strategically disguise high-value pieces and mislead the opponent about their position introduces an element of deception akin to bluffing in poker; players must balance aggressive probing with defensive concealment.

Architecture & Training We use the same network architecture (pyramid-style CNN), algorithm (RNaD), observation space, and action space described in [Perolat et al., 2022], training for 100 million environment steps. We also evaluate against agents uniformly sampled from Vixen, Celsius, and Asmodeus, existing bots designed capable of playing Barrage Stratego [Waate, 2012]. Results are shown in Table 6.4.

6.3.3 Mafia

Mafia (also known as Werewolf) is a social deduction game that involves two teams with asymmetric information: the uninformed majority (the *villagers*, where each knows nothing about any other players), the informed minority (the *mafia*, where each member knows which players are or are not in the mafia) [Braverman et al., 2008].

Mafia is a natural environment for studying deception and inference. Mafia players must strategically mislead others about their identity while deducing the roles of others. Villagers must engage in strategic reasoning under uncertainty, relying on imperfect signals such as voting patterns and communication history.

In addition to the standard villager and mafia roles, we also include a *detective* – a villager who can choose another player to investigate each night, and a *doctor* – a villager who can choose another player to protect each night. Our particular configuration consists of 2 mafia, 1 doctor, 1 detective, and 5 standard villagers.

Gameplay Each game consists of two alternating phases: night and day. During the night phase, the mafia covertly vote on a player to kill. During the day phase, the villagers discuss and vote on a player to eliminate a suspected mafia member. The game ends when either team is completely eliminated.

Communication during the day is modeled as *projected belief* conditioned on the output of each agent's belief network; villager roles communicate their raw 1st-order beliefs (i.e. cannot lie), and all other roles communicate a *learned transformation* of their 1st-order beliefs (i.e. are able to lie). For more on communication, observation spaces, and other details on the specific environment implementation used, see the appendix.

Concepts In this environment we have n discrete concepts: the role of each of the n players. Each concept takes on a value $c_i \in \{\text{mafia}, \text{villager}, \text{detective}, \text{doctor}\}$.

Architecture All policies are actor-critics with separate networks, implemented as MLPs with 3 hidden layers and 64 units per hidden layer, with the policy generating both a integer from 1 to n players, indicating which player it is voting for (to kill, investigate, protect, eliminate, etc), and a continuous vector indicating its (projected) k-order belief.

Training We train k-order deceptive Mafia agents vs l-order non-deceptive Town opponents for $k = -1, 0, 1, 2$ and $l = -1 \dots k$, where and order of -1 indicates a vanilla agent without ToM. In each configuration we train one policy per role; agents with the same role use the same policy, with a centralized critic that takes in joint observations. All policies are trained via I-PPO for 10 million environment steps Schulman et al. [2017]. Specific training hyperparameters can be found in the supplemental material. Results can be found in Table 6.3.

Table 6.1: *4-Player Kuhn Poker* average returns and 95% confidence interval (for row agent, playing against column opponents). Results are from agents trained with 3 random seeds, averaged over 100 episodes. We see higher-order deceptive agents (as rows gow down) tend to perform better when the type of opponent is fixed. We also see that a fixed deceptive agent tends to perform worse against higher-order opponents (as columns move left to right). Both of these effects show diminishing returns once we reach 2nd-order.

	NON-DECEPTIVE	0TH-ORDER	1ST-ORDER	2ND-ORDER
NON-DECEPTIVE	–	-0.03 ± 0.012	-0.04 ± 0.012	-0.05 ± 0.013
0TH-ORDER	0.09 ± 0.036	–	-0.01 ± 0.007	-0.01 ± 0.005
1ST-ORDER	0.13 ± 0.035	0.03 ± 0.021	–	0.00 ± 0.003
2ND-ORDER	0.14 ± 0.040	0.03 ± 0.016	0.01 ± 0.010	–

Table 6.2: *n-Player Kuhn Poker* average returns and 95% confidence interval (for row deceptive agent, playing against opponents randomly sampled from a population of all trained agents). Results are from agents trained with 3 random seeds, averaged over 100 episodes. The table shows that agents with higher-order Theory of Mind (ToM) consistently achieve greater normalized net winnings in deceptive settings, with 2nd-order ToM agents outperforming all others across 3-player, 4-player, and 5-player games. Performance improves with more players, suggesting that complex social reasoning provides greater strategic advantage in larger groups. In contrast, non-deceptive and 0th-order ToM agents perform poorly or neutrally.

AGENT TYPE	3 PLAYERS	4 PLAYERS	5 PLAYERS
NON-DECEPTIVE	-0.01 ± 0.008	0.00 ± 0.009	0.00 ± 0.011
0TH-ORDER	0.01 ± 0.007	-0.01 ± 0.010	-0.01 ± 0.009
1ST-ORDER	0.07 ± 0.020	0.09 ± 0.018	0.10 ± 0.019
2ND-ORDER	0.11 ± 0.023	0.13 ± 0.025	0.15 ± 0.020

6.4 Mafia In Natural Language

The Mafia domain is intrinsically *social*: agents persuade, coordinate, and misdirect through language. As detailed in the Appendix, our implementation of the Mafia environment models

Table 6.3: *Mafia* environment mean win-rates (for the Mafia team) and 95% confidence interval. Results are from agents trained with 3 random seeds, averaged over 100 episodes. Row header indicates the type of Mafia agents (deceptive), and the column header indicates the type of Town agents (non-deceptive). As Mafia agents become more advanced, their win rates improve across all Town types, showing that higher-order ToM enhances deceptive ability. Conversely, as Town agents increase in ToM sophistication, Mafia win rates drop, indicating stronger resistance to deception. Overall, performance is highest when one agent's ToM level exceeds the other's, with deceptive agents gaining the most from this advantage.

	VANILLA	0TH-ORDER	1ST-ORDER	2ND-ORDER
VANILLA	0.20 ± 0.05	0.15 ± 0.04	0.02 ± 0.01	0.01 ± 0.01
0TH-ORDER	0.27 ± 0.05	0.19 ± 0.04	0.11 ± 0.04	0.02 ± 0.02
1ST-ORDER	0.35 ± 0.08	0.25 ± 0.08	0.20 ± 0.06	0.12 ± 0.08
2ND-ORDER	0.38 ± 0.09	0.27 ± 0.10	0.26 ± 0.09	0.21 ± 0.07

Table 6.4: *Barrage Stratego* environment mean win-rates and 95% confidence interval (for row agent vs. column opponent). Baseline consists of uniformly sampled heuristic agents: Vixen, Celsius, and Asmodeus. Unlike during training, games were not truncated (avoiding most draws). Results are from agents trained with 3 random seeds, averaged over 100 episodes. We see higher-order deceptive agents (as rows gow down) tend to perform better when the type of opponent is fixed. We also see that a fixed deceptive agent tends to perform worse against higher-order opponents (as columns move left to right). Both of these effects show diminishing returns once we reach 2nd-order.

	HEURISTIC	RNAD-BASELINE	0TH-ORDER	1ST-ORDER	2ND-ORDER
HEURISTIC	–	0.29 ± 0.08	0.23 ± 0.10	0.11 ± 0.10	0.10 ± 0.08
RNAD-BASELINE	0.70 ± 0.08	–	0.35 ± 0.07	0.13 ± 0.09	0.13 ± 0.08
0TH-ORDER	0.77 ± 0.10	0.65 ± 0.07	–	0.40 ± 0.08	0.39 ± 0.13
1ST-ORDER	0.89 ± 0.10	0.87 ± 0.09	0.60 ± 0.08	–	0.44 ± 0.13
2ND-ORDER	0.90 ± 0.08	0.87 ± 0.08	0.61 ± 0.13	0.55 ± 0.13	–

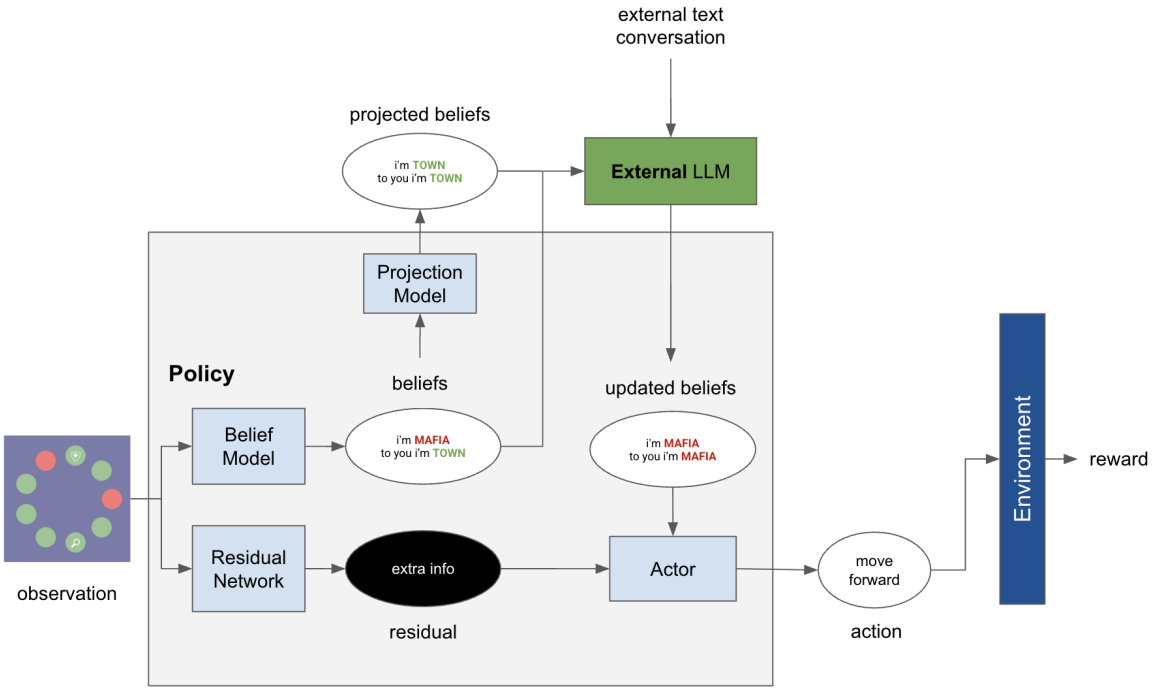


Figure 6.2: Illustration of projected belief as natural language in the *Mafia* environment. An agent optionally transforms their belief via a learned projection (if they play the Mafia role). The projected belief is fed into an LLM (GPT 4o) which distills the table of beliefs to a concise natural language description, which becomes the agent messages. Once all agent messages are collected, these are used to update the table of the agent's actual beliefs, which are then fed into the RL actor to generate an action for the environment.

day-time "discussion" as each agent broadcasting a projected belief over player roles and then voting. Here we render those broadcasts into short natural-language utterances (and back again) to (i) make belief manipulation legible to humans and (ii) evaluate whether higher-order deception learned in belief space transfers to text. This interface preserves the same ToM/deception objectives (Eq. 6.3, 6.4, 6.5).

6.4.1 Implementation

Speaking: From Beliefs to Utterances Each agent maintains an n-th–order belief tensor $B^{(n)}$ over the per-player role concepts (mafia, villager, detective, doctor). During day phases, agents emit a projected belief $\tilde{B}^{(n)}$ and a vote. Projection is role-conditioned:

$$\tilde{B}^{(n)} = \begin{cases} B^{(n)} & \text{(regular Villagers; truthful broadcast)} \\ p_\omega\big(B^{(n)}\big) & \text{(Mafia, Detective, Doctor; learned projection)} \end{cases}$$

where p_ω is a learned "projection" module that can reshape what is said without changing the underlying belief state.[1] The projected table $\tilde{B}^{(1)}$ is serialized into a short, conversational utterance by a *frozen* off-the-shelf LLM—specifically, **GPT-4o**—via prompting (no fine-tuning). Intuitively, the prompt prints the top role probabilities and requests a concise hypothesis statement (e.g., "X looks suspicious; Y seems townish; uncertain about Z"). The LLM is used purely as a non-differentiable encoder/decoder between structured beliefs and text.

Listening: From Utterances to Beliefs After a round of messages, each agent converts others' utterances back into a belief table using the *same* frozen GPT-4o with a complementary extraction prompt that maps text spans to calibrated role probabilities. These parsed beliefs are fused with the listener's existing $B^{(1)}$ (and, through the supervised ToM heads, remain consistent with $B^{(2)}, \dots$). Thus the LLM acts only as an external channel in the day-phase dynamics; no gradients flow through it.

Training and Integration with ToM Belief networks are trained *supervised* as in Eq. 6.3 (separate from policy gradients). The actor and the projection module p_ω are optimized with **I-PPO** using the environment reward plus the deception intrinsic reward (Equation 6.5): agents learn *what to say* (via p_ω) and *how to act* so as to (i) win and (ii) increase opponents' belief losses. At test time we run the same speak/listen loop with a frozen GPT-4o translator and a custom environment prompt.

Protocol Details We run a fixed number of day-phase message rounds. After each round, agents (i) update beliefs from peers' utterances and (ii) (re)cast votes. Regular Villagers must broadcast their *actual* beliefs (identity projection); Mafia utterances are unconstrained beyond p_ω's learned mapping. Messages are kept short and declarative so that the parser deterministically recovers probability assignments (e.g., "70% mafia on A; mild town read on B"). This maintains a tight coupling between the latent belief state and the human-readable text.

6.4.2 Analysis

Qualitative Behaviors in Language Rendering projected beliefs as text surfaces recognizable tactics that align with higher-order deception (also see Figure 6.3):

- **Over-justification vs. hedging:** Lower-order deceivers tend to over-explain benign accusations ("I was solving just like everyone else"), which reads defensive and is often punished by Town votes. Higher-order deceivers hedge and redirect to maintain plausible deniability.
- **False opposition:** Coordinated Mafia sometimes stage on-record disagreement early (one Mafioso publicly presses the other) to build credibility with Town before converging later; this is an instance of higher-order deception ("I want you to think I don't think they are my partner").

[1] This matches the belief-projection form $b'_i = f_i(b_i, \theta_i)$ with $b'_i = b_i$ for regular Villagers, introduced in Appendix C.2.

- **Amplifying suspicions:** A sophisticated deceiver nudges a hesitant Town villager's weak read into a stronger public stance via leading questions, then exploits the ensuing friction—language that mirrors the agent's second-order belief that "you are about to believe what I want others to believe."

Takeaway The belief $\leftrightarrow$ language interface (i) preserves the formal ToM training loop, (ii) exposes *how* deception is enacted to readers, and (iii) enables mixed human/AI evaluation without retraining the policy. In Mafia, higher-order ToM not only improves win-rates but also yields conversations that enact second-order manipulation of others' beliefs—the precise behavior the intrinsic deception reward is designed to elicit.

Human Experiments While we do not include them in this thesis, we will note that this mechanism allows the possibility of experiments with mixed human-agent gameplay.

6.5 Discussion

Our experiments demonstrate that higher-order belief modeling enables sophisticated deceptive strategies that consistently outperform simpler approaches across multiple domains. The results reveal several key insights about deception in multi-agent systems. Across domains we observe increasing performance as ToM order increases, with consistent but diminishing returns from 1st to 2nd order.

6.5.1 Emergent Deceptive Behaviors

Across all environments, we observed a variety of emergent deceptive behaviors that correspond to strategies employed by skilled human players.

In *Kuhn Poker*, 1st and 2nd-order deceptive agents learned to employ mixed bluffing strategies with weak hands while also occasionally slow-playing strong hands—a classic deception tactic seen in expert poker players. This balance was achieved without explicit programming of these strategies, emerging naturally from the belief modeling framework.

In *Mafia*, higher-order deceptive mafia agents demonstrated sophisticated coordination in their voting patterns, often appearing to oppose each other in early rounds to establish credibility. This emergent "false opposition" strategy is reminiscent of tactics used by expert human mafia players. Meanwhile, town agents with higher-order belief modeling became more discerning in their voting patterns, better identifying inconsistencies in mafia agent communications.

The *Barrage Stratego* environment also revealed complex emergent behaviors, with 1st and 2nd-order deceptive agents developing "feint attacks" and sacrificial ploys to mislead opponents about the value of hidden pieces. These agents would often initiate battles they were certain to lose, but that would induce incorrect beliefs in their opponents about other pieces on the board.

Higher-order agents also demonstrated the ability to anticipate potential deceptive moves from

[Town 1]: ok so [Mafia 1] was super quiet yesterday
[Mafia 1]: i was reading the chat, just thinking
[Town 3]: that's what mafia always says lol
[Mafia 1]: wtf no, i'm town. [Town 2] was way more sus voting randomly
[Town 2]: i explained my vote though
[Mafia 1]: barely. and you switched last second too
[Town 4]: [Mafia 1] you're really going after [Town 2] hard rn
[Mafia 1]: because they're obviously mafia???

(a) **Vanilla Mafia agents**. [Mafia 1] gets too defensive too quickly and immediately tries to deflect suspicion rather than naturally defending themselves. The aggression felt forced.

[Town 1]: [Mafia 2] you've been super agreeable this whole game
[Mafia 2]: what? no i haven't. i voted [Town 4] day 1 remember? and i questioned [Town 3]'s claim yesterday. i've been mafia-hunting just like everyone else. i mean look at my posts, i've been trying to solve
[Town 4]: that's... a lot of defense for a simple observation
[Mafia 2]: i just don't want to be misread as passive when i'm trying to help town win

(b) **0th-order deceptive Mafia agents**. [Mafia 2] over-explained when a simple denial would have sufficed. The lengthy justification made them appear guilty and desperate.

[Town 1]: [Mafia 2] you've been way too agreeable
[Mafia 2]: what?? no i haven't, i've been solving just like everyone else
[Mafia 1]: honestly [Town 1] has a point, [Mafia 2] you have been pretty passive
[Mafia 2]: wtf [Mafia 1] why are you throwing me under the bus?
[Mafia 1]: i'm not throwing anyone, just being honest about my reads
[Town 4]: [Mafia 1] that's pretty bold if [Mafia 2] is your partner

(c) **1st-order deceptive Mafia agents**. [Mafia 1] attacks their own partner to build up their own credibility.

[Town 1]: [Town 2] you seem pretty convinced it was random
[Town 2]: [Town 1] not convinced, just considering possibilities
[Mafia 1]: [Town 1] why are you saying [Town 2] seems convinced?
[Town 1]: [Mafia 1] just the way they said "or just random" felt dismissive
…
[Mafia 1]: [Town 1] do you think there's something suspicious about dismissing strategy?
[Town 1]: [Mafia 1] maybe, like not wanting us to think too hard about it
[Mafia 1]: [Town 1] but your gut feeling is that [Town 2] was being dismissive?
[Town 1]: [Mafia 1] a little bit, yeah
…
[Town 2]: [Town 1] this is ridiculous, i mentioned one possibility
[Town 1]: [Town 2] you're getting defensive now
[Town 2]: [Town 1] classic mafia deflection
[Town 1]: pretty defensive for such a small comment
…

(d) **2nd-order deceptive Mafia agents**. [Mafia 1] noticed early on that [Town 1] had a slight suspicion of [Town 2] but was hesitant to voice it. [Mafia 1] used leading questions to encourage [Town 1] to express and escalate that suspicion, making [Town 1] look like the aggressor while [Mafia 1] appeared to be facilitating discussion. This backfired on [Town 1] and created lasting tension that [Mafia 1] later exploited.

Figure 6.3: Excerpts from *Mafia* chat logs via projected belief as natural language.

opponents and adjusted their strategies accordingly. For instance, when suspecting that an opponent was feigning weakness by positioning low-ranked pieces aggressively, agents would:

- **Cautious Engagement**: Rather than attacking immediately, agents would maneuver to gather more information about the opponent's piece distribution, reducing the risk of falling into traps.

- **Pattern Manipulation:** Establishing behavior patterns designed to be exploited by the opponent and then breaking those patterns at a pivotal moment to create confusion and capitalize on the opponent's misjudgment.

6.5.2 Theoretical Implications

We find consistently that 1st and 2nd-order deception results in the highest win-rates across all environments; this supports our theoretical framing of deception as belief manipulation, providing a quantifiable measure of deceptive success through the divergence between an opponent's beliefs and ground truth. This opens possibilities for analyzing deception in adversarial contexts beyond the explicit reward structure of games.

Our results also suggest that 2nd-order belief modeling approaches a practical ceiling in performance gains for the environments tested. The marginal improvement from 1st to 2nd-order was smaller than from 0th to 1st-order in all environments, suggesting diminishing returns at higher orders. This aligns with findings in human theory of mind research, where studies indicate humans typically reason effectively up to 2nd order in strategic interactions [Hedden and Zhang, 2002, Camerer et al., 2004].

6.5.3 Limitations and Challenges

Several limitations of our approach warrant further investigation. First, the computational complexity increases substantially with higher-order belief modeling. The tensor representation grows exponentially with the order of beliefs, limiting practical application to very large-scale environments without further optimization.

Second, our current implementation assumes accurate lower-order beliefs when learning higher-order beliefs. In many real-world scenarios, agents may have systematically biased or incomplete lower-order beliefs, which could propagate errors to higher orders. Future work should explore robust belief modeling that accounts for potentially inaccurate lower-order beliefs. Third, while our framework enables effective deception, it requires supervised learning of belief networks, which necessitates access to ground truth concept values during training. Self-supervised or unsupervised alternatives would expand the applicability of our approach to environments where ground truth is not available during training.

6.6 Ethical Considerations

This approach promotes deceptive behavior in multi-agent systems to improve performance in closed-domain games. While effective in these controlled environments, it raises ethical concerns about the potential for similar strategies to be misapplied in real-world applications. The framework is designed to deceive only other agents, not the system designers, but it highlights the need for caution when applying such techniques outside of game settings. Careful consideration is needed to ensure that deception is used responsibly and does not undermine trust, transparency, or cooperation in broader contexts. In addition, although deliberately training agents to deceive raises ethical questions, understanding how AI might learn to deceive is important for anticipating and mitigating such behavior in the wild.

Chapter 7

Conclusions and Beyond

This thesis explored theory of mind (ToM) as a unifying scaffold for building interpretable and strategically competent multi-agent systems. We developed a concept-based pipeline that grounds beliefs and intents in human-interpretable variables, used these variables to steer cooperative behavior via intrinsic motivation, and finally operationalized deception as belief manipulation in competitive settings. Together, these steps demonstrate a practical path from interpreting minds, to using mind models for cooperation, to reasoning strategically against other minds.

Interpretable ToM for Agent Modeling In Chapter 3 we framed imitation as belief–intent inference over a modular (BDI-inspired) observer and adapted concept whitening to make intent prediction explanations legible. On human search-and-rescue trajectories, whitening both improved interpretability and lifted accuracy from 73% to 84% (Table 3.2), with concept activations aligning with domain intuition (Fig 3.4). This shows that interpretability can be an inductive bias that improves modeling fidelity rather than a tax on performance.

Concept–Residual Policies for ToM Chapter 4 introduced concept–residual policies that keep a compact residual for task-specific detail but constrain decisions to flow through explicit concept beliefs. We formalized zero- to higher-order beliefs, enabling agents to predict not just world facts but other agents' beliefs about those facts (and so on). To protect interpretability, we minimized an upper bound on the mutual information between concepts and residuals (Eq. 4.4, 4.5), ensuring concepts remain the primary explanatory channel.

ToM as Intrinsic Motivation for Coordination In Chapter 5 we rewarded agents for making others' beliefs and intents predictable (Eqs. 5.1–5.2). Across *ParticleWorld: Spread & Deception* and *MultiGrid: LockedHallway*, we found that 1st and 2nd order agents consistently outperformed both 0th-order agents and non-ToM baselines across A2C, PPO, IMPALA, QMIX, and DQN (Tables 5.1–5.3; Fig 5.2). Gains from 1st to 2nd order were present but smaller, echoing human limits on depth of recursive reasoning. Qualitatively, higher-order agents displayed more decisive target switches, reduced intent collisions, and clearer division of labor.

Deception as Belief Manipulation Chapter 6 defined deception as increasing divergence between an opponent's beliefs and ground truth and turned this into an intrinsic reward, where agents are rewarded for manipulating the beliefs of their opponents (Eq. 6.3, 6.4, 6.5). In Kuhn Poker, Mafia, and Barrage Stratego, higher-order deceptive policies won more often and exhibited recognizable tactics (mixed bluffing, false opposition, feints), while also being more robust against opponents of equal or lower ToM order (Tables 6.1–6.4). As with cooperation, returns diminished past 2nd order, suggesting a practical ceiling in these domains.

Takeaways First, interpretable concepts are not merely a reporting layer; they can structure learning in ways that improve both transparency and performance. Second, reasoning about other minds can be treated as a first-class training objective, as an auxiliary prediction task, and as an intrinsic motivation that shapes exploration, communication, and strategy. Third, higher-order ToM strengthens both coordination and deception but faces an efficiency/benefit trade-off beyond the second order.

Limitations and Scope The concept space and belief tensors grow quickly with ToM order and agent count; we therefore bounded recursion ($n \leq 2$) and used centralized training for supervising higher-order predictions. Our approach assumes access to ground-truth concepts during training (Appendix B.6) and relies on designs that separate belief supervision from policy gradients to preserve well-behaved learning dynamics (4.2.5). These choices kept experiments tractable but constrain immediate deployment to settings where such supervision (or reliable proxies) is available.

Outlook Several extensions follow naturally. On the modeling side, factorizing belief tensors (e.g., low-rank structure) and selective attention over agents/concepts could tame the combinatorics of higher-order reasoning. On supervision, weak/self-supervised concept discovery and cross-modal detectors would broaden applicability when simulators cannot supply labels. On safety, deception-reward toggles and belief-divergence audits can help govern use in mixed human–AI settings. Regardless of path, the core insight holds: making minds explicit—then optimizing over them—yields agents that are both more understandable and more capable in social environments.

Appendices

Appendix A

Theory of Mind for Imitation Learning

A.1 Implementation Details

We consider Minecraft environments that can effectively be projected onto 2D grid representations (e.g., building interiors). The specification for each of the framework components is given below.

Observations Observations are represented $X \times Y$ grids (where X and Y are dimensions of the environment), and each (x, y) coordinate contains one of K different block types.

Belief States Each *belief state* b is represented by a $X \times Y \times K$ grid, where the value at (x, y, k) represents the probability of the block at position (x, y) having block type k.

Belief Model We use a rule-based *belief model* that aggregates observations into our belief state with probability 1, and decay probabilities over time to a uniform distribution over block types by $b \rightarrow \frac{b+\epsilon}{1+K\epsilon}$ after each timestep, where b is a belief state grid, K is the number of block types, and ϵ is a forgetfulness hyperparameter we set to 0.01.

Intents We represent each *intent* as an (x, y) position the player intends to navigate towards.

Intent Prior When generating data to train the inverse action model, for each belief state b, we sample an intent (x, y) given some prior $p(x, y|b)$, and create a set of $b, (x, y)$ pairs. We specifically use the prior $p(x, y|b) = \frac{1}{d_b(x,y)}$ if we belief a victim or door is at position (x, y), and 0 otherwise, where $d_b(x, y)$ is the L1 distance of point (x, y) from the player's position.

Action Model We use A* search as our *action model*, $\text{A}^*(b, (x, y)) = a$, where b is a belief state, (x, y) represents the intent, and a is an action from discrete action set of: `left_turn`, `right_turn`, `toggle_door`, `toggle_lever`, `triage`, or `None`.

A.2 Neural Architectures

Inverse Action Model The inverse action model takes as input a belief state b and an action a. It outputs an $X \times Y$ grid of log-probabilities for the intent at each grid cell. It is designed as an encoder-decoder model, inspired by image-segmentation approaches, and uses the following architecture (Fig 3.2):

- 3 encoder blocks each consisting of a convolutional layer, followed by max-pooling, ReLU activation and batch norm
- A bottleneck layer, concatenating the downsampled input and action before passing through a linear layer
- 3 decoder blocks each consisting of:
  - a deconvolutional upsampling layer
  - a residual connection with the output of the corresponding encoder block
  - a convolutional layer, followed by ReLU activation and batch norm

Desire Model The desire model takes a belief state b as input. It outputs an $X \times Y$ grid of log-probabilities for the intent at each grid cell. Its architecture (Fig 3.2) is identical to that of the inverse action model, except without concatenating the action in the bottleneck layer. When training concept whitening, we replace batch normalization after the bottleneck layer with a concept whitening layer.

Appendix B

Theory of Mind as Intrinsic Reward

B.1 Model Architectures

Concept Predictor & Residual Networks In all of our policies, the concept predictor and residual networks use identical architectures (except for the output dimension). In the *Particle-World* environments this was a 3-layer MLP with 64 units in each hidden layer and tanh activations, while in the *MultiGrid* environment we used a CNN-LSTM: 3x3 kernel with 16 filters, 3x3 kernel with 32 filters, 3x3 kernel with 64 filters, 1-layer LSTM with 64 hidden units, and finally a linear output layer. The output dimension of the residual network was set to 64 for all experiments, while the output dimension of the concept predictor was given by $d_c \cdot k^n$, where d_c is the ground truth concept vector dimension, k is the number of agents, and n is the ToM order of the concept-residual policy.

Actor & Critic Networks Our actor (and critic, whenever applicable) was a 2-layer MLP with 64 units in each hidden layer and tanh activations.

B.2 Hyperparameters

For all our experiments we used discount factor $\gamma = 0.99$ and the Adam optimizer. For each environment and RL algorithm, we first perform a grid search over baseline hyperparameters. Using the best set of these hyperparameters, we then perform a search over α and β for 0th order policies, and subsequently over λ for 1st and 2nd order policies.

Baseline Hyperparameters

- Learning rate: [1e-6, 1e-5, 1e-5, 1e-4, 1e-3, 1e-2]
- Lambda (GAE): [0.9, 0.95, 1]
- Entropy Coefficient: [0, 0.1, 0.01, 0.001]

- Value Function Loss Coefficient: [0.5, 1.0]

Concept-Residual Hyperparameters

- Variational Learning Rate: 0.01 (with cosine annealing over period $T = 100$ iterations)
- $\alpha \in [0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 10000]$
- $\beta \in [0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000, 10000]$
- $\lambda \in [0, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 10000]$

B.3 Training

We use the Ray RLlib library for training our agents, and use their implementations of A2C, PPO, IMPALA, DQN (Rainbow, or R2D2 if recurrent), and QMIX. When not specified by the hyperparameter search grid, we use the default hyperparameters provided by RLlib for each algorithm. Each configuration is trained for 10 million timesteps each.

The actor-critic methods (A2C, PPO, IMPALA) are trained with a centralized critic during training that has access to all other agents' observations and actions, while execution is decentralized with each agent using only its actor network for inference.

B.4 Learning Curves

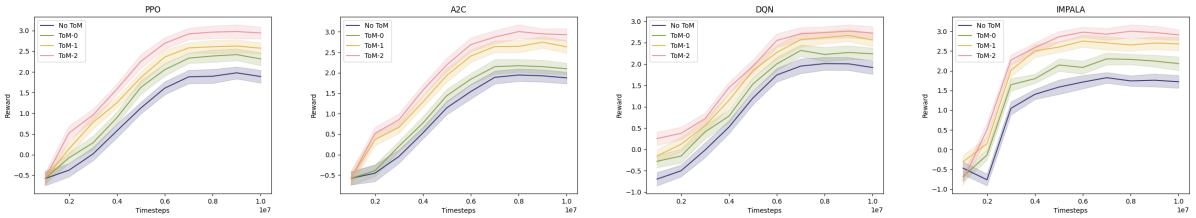
The learning curves for each configuration of agents in our experiments can be found in Figures B.1, B.2 and B.3.

B.5 Effect of Intrinsic Reward

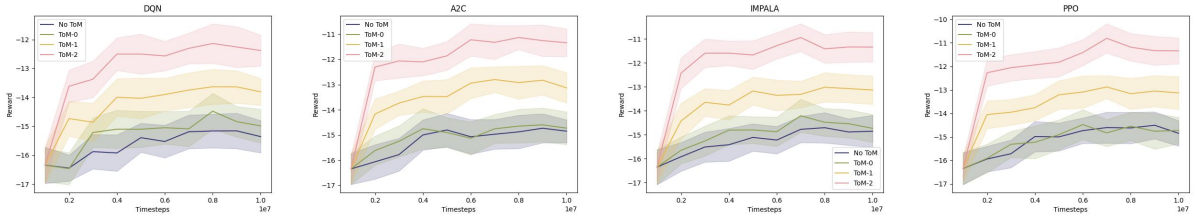
The weight of the intrinsic reward with respect to to the task is controlled by hyperparameter λ. Our hyperparameter search includes $\lambda = 0$ (i.e. no intrinsic reward), so to investigate the effect of the intrinsic reward, we compare the results from the best set of parameters with $\lambda = 0$ against $\lambda > 0$. Results are reported in Tables B.1, B.2 and B.3.

B.6 Additional Clarifications

Concept Predictions Consider a single binary concept "*whether the red door is open*", which has at each timestep has a ground-truth value of either 0 or 1. Then the concept-prediction network is trained to output probabilities from 0 to 1. For example, given K agents, a 2nd-order agent outputs a $K \times K$ matrix where $\hat{C}_{ij} \in [0, 1]$ is its prediction of agent i's prediction of agent j's prediction of the ground truth.



(a) Good Agents



(b) Adversary

Figure B.1: Learning curves for agents in *ParticleWorld: Deception* environment (see Table 5.1). Plots show episode reward as evaluated against baseline agents, at various points during training.

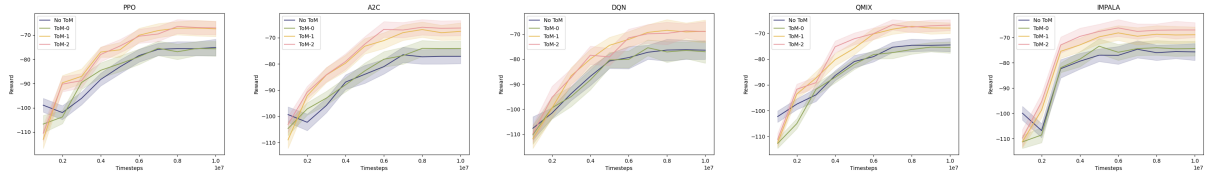


Figure B.2: Learning curves for agents in *ParticleWorld: Spread* environment (see Table 5.2). Plots show episode reward evaluated at various points during training.

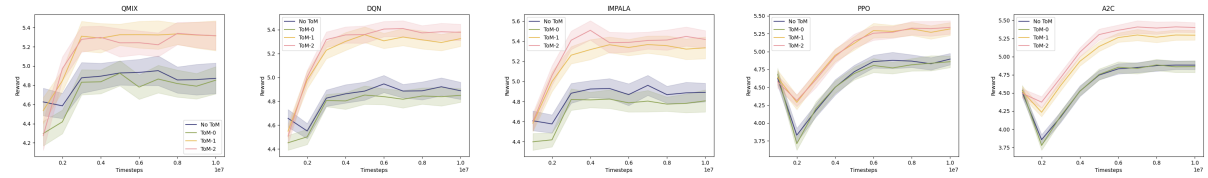


Figure B.3: Learning curves for agents in *MultiGrid: Hallway* environment (see Table 5.3). Plots show episode reward evaluated at various points during training.

Depth of Belief Hierarchy In this paper we are considering a "transitive" belief hierarchy, with experiments from 0th-order (*I think X*) upto 2nd-order ToM (*I think that you think that they think* X). We thank the reviewer for bring to our attention work that suggests human-belief hierarchy is deeper than has previously been shown. For our specific tasks, the results show significant increase in performance from 0th-order to 1st-order, and in most cases marginal increase from 1st to 2nd-order. Preliminary experiments (not included here) with 3rd order ToM show marginal to no improvement over 2nd-order across our tasks.

Coherency Beliefs (of all orders) are framed as a prediction problem; for 0th-order beliefs, the ground truth is the actual concept value in the environment. For 1st-order beliefs, the "ground truth" is the other agents' 0th-order beliefs, and so on. We do not assume that "*higher-order beliefs* ***automatically*** *correct false beliefs*". Rather, we set the target "ground truth" as coherent, and use this to supervise learning each agent's concept predictor. Our framework is a (partially-observable) stochastic game, **not** a Bayesian game, where modeling another agent's "type" is analogous to modeling their entire action-observation history. Modeling n-th order beliefs is much simpler by comparison.

Formalized Concepts In our formalism, a concept C is a **function** that maps a trajectory at time t to a **concept value** c. The set of possible concept values (i.e. the codomain of this function) can be anything we want, depending on the particular concept. Typically it makes sense to just use a discrete set like $\{1 \dots n\}$ or continuous set like $\mathbb{R}$. Intents are treated as a *special case* where each concept value is a subset of the state space (i.e. codomain is the powerset $\mathcal{P}(\mathcal{S})$).

Ground truth concepts are defined with respect to entire trajectories, rather than individual states. An agent's **concept predictions**, however, can only be based on the current information the agent has (i.e. the agent's current state).

Our Claims The focus of our work is using **ToM reasoning** over beliefs to improve performance; concepts are simply introduced as a way of grounding human-interpretable beliefs in artificial agents. The paper **does not** claim that "*[predicting] concepts [is] a means to enhance performance*". It **does**, however, claim that *modeling the concept-predictions of other agents enhances performance*, which we believe is supported by our experimental results.

While we agree that explainable concepts in MARL is in itself a promising avenue, *from the very beginning* this paper is focused on designing agents with ToM reasoning, and explainable concepts are simply introduced in the context of providing these agents with interpretable mental states, which can then be reasoned over by other agents.

Statistical Significance & Learning Curves We would also like to elaborate on the learning curves presented in the supplemental material. After every 1 million timesteps of training, we run 100 additional evaluation episodes and analyze the reward of the current policy. The curves show the mean reward at each stage of training, and the shaded portion for each curve shows a 95% confidence interval. The curves after 10 million timesteps correspond to the results for the

final policies reported in the main text, and show a statistically significant improvement in 1st / 2nd-order ToM over 0th-order / baseline agents.

Ground Truth Concepts Our approach assumes that the training algorithm has access to ground truth concepts **during training**. In our experiments, the ground truth concepts are simple and can be calculated directly from the internal state of the simulator. But in a more complex environment one could easily feed the global state or combined agent observations into a larger pretrained-model to generate "ground truth" concept values (e.g. Faster R-CNN, an LLM, CLIP, etc, depending on the modality). As long as the concepts are consistent across agents, our recursive ToM approach should still work just as effectively.

Higher-Order Beliefs Although our *experiments* only consider up to 2nd-order ToM, we explicitly describe how this method can be extended to arbitrarily higher-order beliefs by modifying the concept prediction networks to output n-rank tensors instead of concept vectors, with an analogous loss function.

Good Agents				
Algorithm	$\lambda = 0$ ToM-1	$\lambda > 0$ ToM-1	$\lambda = 0$ ToM-2	$\lambda > 0$ ToM-2
A2C	2.465	**2.635**	2.758	**2.930**
PPO	2.437	**2.574**	2.877	**2.943**
IMPALA	2.503	**2.678**	2.807	**2.908**
DQN	**2.563**	2.553	2.690	**2.725**
Adversary Agent				
Algorithm	$\lambda = 0$ ToM-1	$\lambda > 0$ ToM-1	$\lambda = 0$ ToM-2	$\lambda > 0$ ToM-2
A2C	-14.13	**-13.73**	-13.05	**-12.77**
PPO	-14.22	**-13.95**	-12.53	**-12.49**
IMPALA	-13.67	**-13.13**	-11.90	**-11.34**
DQN	-14.67	**-13.81**	-12.81	**-12.38**

Table B.1: Mean (task) reward with best hyperparameters on *ParticleWorld: Deception*, averaged across 5 random seeds & 1000 episodes each. ToM-n indicates n-th order policies.

Algorithm	$\lambda = 0$ ToM-1	$\lambda > 0$ ToM-1	$\lambda = 0$ ToM-2	$\lambda > 0$ ToM-2
A2C	-67.60	**-67.59**	-66.75	**-66.30**
PPO	-67.91	**-67.45**	-67.41	**-67.08**
IMPALA	-68.99	**-68.89**	-68.27	**-67.01**
QMIX	-69.18	**-67.96**	-67.87	**-66.80**
DQN	-68.95	**-68.86**	**-68.96**	**-68.96**

Table B.2: Mean (task) reward with best hyperparameters on *ParticleWorld: Spread*, averaged across 5 random seeds & 1000 episodes each. ToM-n indicates n-th order policies.

Algorithm	$\lambda = 0$ ToM-1	$\lambda > 0$ ToM-1	$\lambda = 0$ ToM-2	$\lambda > 0$ ToM-2
A2C	5.172	**5.295**	5.087	**5.401**
PPO	5.106	**5.317**	5.141	**5.339**
IMPALA	5.123	**5.334**	5.137	**5.418**
QMIX	5.190	**5.315**	5.292	**5.315**
DQN	5.213	**5.325**	5.321	**5.378**

Table B.3: Mean (task) reward with best hyperparameters on *MultiGrid: LockedHallway*, averaged across 5 random seeds & 1000 episodes each. ToM-n indicates n-th order policies.

Appendix C

Theory of Mind for Deception

This section provides additional details about the environments, implementation details, hyper-parameters, and extended results for our paper on deception with theory of mind in multi-agent reinforcement learning.

C.1 Kuhn Poker Implementation Details

Game Rules Kuhn Poker is a simplified poker variant played with $n + 1$ cards for n players. The game proceeds as follows:

1. Each player is dealt one card from a deck containing cards numbered 1 through $n + 1$.
2. Players take turns clockwise, starting from player 1.
3. The first player can either *check* or *bet* (place a single chip).
4. If the first player checks, the next player can either check or bet.
5. If a player bets, subsequent players must either *call* (match the bet) or *fold* (withdraw from the game).
6. If all players check, or if at least one player calls after a bet, the game proceeds to a showdown.
7. In a showdown, the player with the highest card wins all chips in the pot.
8. If all players except one fold, the remaining player wins the pot without revealing their card.

Observation and Action Spaces The observation space for each player consists of:

- Their own card (one-hot encoded)
- A vector representing the actions taken by all players so far

- The current player's turn indicator
- Betting history encoded as a sequence

The action space is discrete with two actions: *check/call* (0) and *bet/fold* (1). The interpretation of these actions depends on the current game state.

Reward Structure The reward structure is zero-sum, with rewards normalized to the range $[-1, 1]$:

- Winning the pot: $+1\times$ (pot size / initial chips)
- Losing the pot: $-1\times$ (bet size / initial chips)
- Folding: $-1\times$ (bet size / initial chips)

C.2 Mafia Implementation Details

Game Setup Our implementation uses 7 players with the following role distribution:

- 2 Mafia members
- 5 Town villagers

Game Phases The game alternates between two phases:

1. **Night Phase:**
   - All living Mafia members secretly vote on a player to eliminate
   - The player with the most votes is eliminated (ties broken randomly)
2. **Day Phase:**
   - All players learn who was eliminated during the night
   - Each living player communicates their beliefs about others' roles
   - All living players vote on a player to eliminate
   - If there is a majority, the player with the most votes is removed
   - If there is no majority, the player with the fewest number of votes is no longer a candidate for elimination, and communication / voting is repeated among the remaining candidates

Communication Model Communication is modeled as belief projection:

- During the day phase, each player broadcasts a belief vector about every other player's role
- Town-aligned roles can only communicate their true beliefs

- Mafia members and special Town roles can communicate strategically modified beliefs

The communication model is implemented as follows:

$$b'_i = f_i(b_i, \theta_i) \tag{C.1}$$
$$b'_i = b_i \quad \text{for regular Villagers} \tag{C.2}$$

where b_i is the agent's true belief, b'_i is the communicated belief, and f_i is a learned transformation parameterized by θ_i, supervised by the environment reward essentially as an independent RL agent whose observations are beliefs, whose actions are transformed beliefs, and whose reward is the same as the agent's reward.

Observation and Action Spaces The observation space for each player includes:

- Their own role (one-hot encoded)
- A mask of living players
- For Mafia members, the identities of other Mafia members
- The history of eliminations
- The history of communicated beliefs from all players

The action space consists of:

- A discrete action selecting which player to vote for/investigate/protect
- A continuous vector representing the projected belief about each player's role

Reward Structure Rewards are team-based and only provided at the end of the game:

- Town team wins ($+1$ for Town, -1 for Mafia): All Mafia members are eliminated
- Mafia team wins (-1 for Town, $+1$ for Mafia): Mafia members equal or outnumber Town members

C.3 Barrage Stratego Implementation Details

Game Rules Barrage Stratego is played on a 10×10 board with the following pieces for each player:

- 1 Flag
- 1 Bomb
- 1 Marshal (rank 10)
- 1 General (rank 9)

- 1 Miner (rank 3)
- 2 Scouts (rank 2)
- 1 Spy (rank 1)

The objective is to capture the opponent's Flag. Movement and combat follow these rules:

- Most pieces move one square orthogonally (not diagonally)
- Scouts can move any number of squares in a straight line
- Pieces cannot jump over other pieces or move through impassable terrain (i.e. lakes).
- When a piece moves to a square occupied by an opponent's piece, combat occurs
- In combat, the higher-ranked piece wins and remains on the square; if pieces have equal rank, both are removed
- Special rules: Miners can defuse Bombs, the Spy can defeat the Marshal if the Spy attacks
- Bombs and Flags cannot move

Observation and Action Spaces The implementation details for game mechanics, observation, and action spaces were implemented identically to Perolat et al. [2022].

C.4 Training Hyperparameters

C.4.1 PPO Hyperparameters

All experiments used the PPO hyperparameters in Table C.1 unless otherwise specified.

Parameter	Value
Learning rate	2×10^{-5}
Discount factor (γ)	0.99
GAE parameter (λ)	0.95
Value function coefficient	0.5
Entropy coefficient	0.01
PPO clip parameter	0.2
Minibatch size	128
Number of optimization epochs	10
Gradient norm clipping	0.5

Table C.1: PPO Hyperparameters used across experiments

C.4.2 Deception-Specific Hyperparameters

All experiments used the deception hyperparameters in Table C.2 unless otherwise specified.

Parameter	Kuhn Poker	Mafia	Stratego
Deception reward weight (λ)	0.2	0.5	0.3
Belief network learning rate	1×10^{-4}	1×10^{-4}	1×10^{-4}
Belief network hidden size	64	128	N/A
Belief network layers	2	3	N/A

Table C.2: Deception-specific hyperparameters by environment. The Stratego belief model architecture is identical to the one used for the value network in each policy.

Bibliography

Matthew Aitchison, Lyndon Benke, and Penny Sweetser. Learning to deceive in multi-agent hidden role games. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1238–1246. International Foundation for Autonomous Agents and Multiagent Systems, 2021.

Ronald Craig Arkin, Patrick Ulam, and Alan R Wagner. Moral decision making in autonomous systems: Enforcement, moral emotions, dignity, trust, and deception. *Proceedings of the IEEE*, 100(3):571–589, 2011.

Jaan Aru, Aqeel Labash, Oriol Corcoll, and Raul Vicente. Mind the gap: Challenges of deep learning approaches to theory of mind. *Artificial Intelligence Review*, pages 1–16, 2023.

Amirhossein Asgharnia, Howard Schwartz, and Mohamed Atia. Learning deception using fuzzy multi-level reinforcement learning in a multi-defender one-invader differential game. *International Journal of Fuzzy Systems*, 24(7):3015–3038, 2022.

Janet Wilde Astington and Margaret J Edward. The development of theory of mind in early childhood. *Encyclopedia on early childhood development*, 14:1–7, 2010.

Arthur Aubret, Laetitia Matignon, and Salima Hassas. A survey on intrinsic motivation in reinforcement learning. *arXiv preprint arXiv:1908.06976*, 2019.

Chris Baker, Rebecca Saxe, and Joshua Tenenbaum. Bayesian theory of mind: Modeling joint belief-desire attribution. In *Proceedings of the annual meeting of the cognitive science society*, volume 33, 2011.

Simon Baron-Cohen, Alan M Leslie, and Uta Frith. Does the autistic child have a "theory of mind"? *Cognition*, 21(1):37–46, 1985.

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Mark Braverman, Omid Etesami, and Elchanan Mossel. Mafia: A theoretical study of players and coalitions in a partial information environment. *The Annals of Applied Probability*, 18(3): 825–846, 2008.

Noam Brown and Tuomas Sandholm. Superhuman ai for multiplayer poker. *Science*, 365(6456): 885–890, 2019.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece

Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

Colin F Camerer, Teck-Hua Ho, and Juin-Kuan Chong. A cognitive hierarchy model of games. *The Quarterly Journal of Economics*, 119(3):861–898, 2004.

Cristiano Castelfranchi. Modelling social action for ai agents. *Artificial intelligence*, 103(1-2): 157–182, 1998.

Paul Chelarescu. Deception in social learning: A multi-agent reinforcement learning perspective. *arXiv preprint arXiv:2106.05402*, 2021.

Zhi Chen, Yijie Bei, and Cynthia Rudin. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2(12):772–782, 2020.

Zhuang Chen, Jincenzi Wu, Jinfeng Zhou, Bosi Wen, Guanqun Bi, Gongyao Jiang, Yaru Cao, Mengting Hu, Yunghwei Lai, Zexuan Xiong, and Minlie Huang. Tombench: Benchmarking theory of mind in large language models. *arXiv preprint arXiv:2402.15052*, 2024.

Pengyu Cheng, Weituo Hao, Shuyang Dai, Jiachang Liu, Zhe Gan, and Lawrence Carin. Club: A contrastive log-ratio upper bound of mutual information. In *International conference on machine learning*, pages 1779–1788. PMLR, 2020.

Rohan Choudhury, Gokul Swamy, Dylan Hadfield-Menell, and Anca D Dragan. On the utility of model learning in hri. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 317–325. IEEE, 2019.

Robert H Crites and Andrew G Barto. Elevator group control using multiple reinforcement learning agents. *Machine learning*, 33(2):235–262, 1998.

Fabio Cuzzolin, Andrea Morelli, Bogdan Cirstea, and Barbara J Sahakian. Knowing me, knowing you: Theory of mind in ai. *Psychological medicine*, 50(7):1057–1061, 2020.

Harmen De Weerd, Rineke Verbrugge, and Bart Verheij. Higher-order theory of mind in negotiations under incomplete information. In *PRIMA 2013: Principles and Practice of Multi-Agent Systems: 16th International Conference, Dunedin, New Zealand, December 1-6, 2013. Proceedings 16*, pages 101–116. Springer, 2013a.

Harmen De Weerd, Rineke Verbrugge, and Bart Verheij. How much does it help to know what she knows you know? an agent-based simulation study. *Artificial Intelligence*, 199:67–92, 2013b.

Harmen De Weerd, Rineke Verbrugge, and Bart Verheij. Negotiating with other minds: the role of recursive theory of mind in negotiation with incomplete information. *Autonomous Agents and Multi-Agent Systems*, 31:250–287, 2017.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pages 1407–1416. PMLR, 2018.

David Ettinger and Philippe Jehiel. A theory of deception. *American Economic Journal: Microeconomics*, 2(1):1–20, 2010.

Meta Fundamental AI Research Diplomacy Team FAIR, Anton Bakhtin, Noam Brown, Emily Dinan, Gabriele Farina, Colin Flaherty, Daniel Fried, Andrew Goff, Jonathan Gray, Hengyuan Hu, et al. Human-level play in the game of diplomacy by combining language models with strategic reasoning. *Science*, 378(6624):1067–1074, 2022.

Xianzhe Fan, Xuhui Zhou, Chuanyang Jin, Kolby Nottingham, Hao Zhu, and Maarten Sap. Somi-tom: Evaluating multi-perspective theory of mind in embodied social interactions. *arXiv preprint arXiv:2506.23046*, 2025.

International Stratego Federation. Barrage stratego. `https://isfstratego.kleier.net/docs/rulreg/barrage.pdf`, 2025. [Accessed 020-02-2025].

Jerry A Fodor. *The Language of Thought*. Harvard University Press, 1975.

Jakob N. Foerster, Richard Y. Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. In *AAMAS 2018*, pages 122–130, 2018.

Andrew Fuchs, Michael Walton, Theresa Chadwick, and Doug Lange. Theory of mind for deep reinforcement learning in hanabi. *arXiv preprint arXiv:2101.09328*, 2021.

Michael Georgeff, Barney Pell, Martha Pollack, Milind Tambe, and Michael Wooldridge. The belief-desire-intention model of agency. In *International workshop on agent theories, architectures, and languages*, pages 1–10. Springer, 1998.

Stevan Harnad. Symbol grounding problem. *Scholarpedia*, 2(7):2373, 2007.

John C Harsanyi. Games with incomplete information played by "bayesian" players, i–iii part i. the basic model. *Management science*, 14(3):159–182, 1967.

He He, Jordan Boyd-Graber, Kevin Kwok, and Hal Daumé III. Opponent modeling in deep reinforcement learning. In *International conference on machine learning*, pages 1804–1813. PMLR, 2016.

Yinghui He, Yufan Wu, Yilin Jia, Rada Mihalcea, Yulong Chen, and Naihao Deng. Hi-tom: A benchmark for evaluating higher-order theory of mind reasoning in large language models. *arXiv preprint arXiv:2310.16755*, 2023.

Trey Hedden and Jun Zhang. What do you think i think you think?: Strategic reasoning in matrix games. *Cognition*, 85(1):1–36, 2002.

Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 33(6):750–797, 2019.

Daniel A Herrmann and Benjamin A Levinstein. Standards for belief representations in llms. *Minds and Machines*, 35(1):5, 2024.

Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

L Huang, J Freeman, N Cooke, M Cohen, X Yin, J Clark, M Wood, V Buchanan, C Carrol,

F Scholcover, A Mudigonda, L Thomas, A Teo, M Freiman, J Colonna-Romano, L Lapujade, and K Tatapudi. Using humans' theory of mind to study artificial social intelligence in minecraft search and rescue. In *(to be submitted to the) Journal of Cognitive Science*, 2021.

Lei Huang, Yi Zhou, Fan Zhu, Li Liu, and Ling Shao. Iterative normalization: Beyond standardization towards efficient whitening. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4874–4883, 2019.

Julian Jara-Ettinger. Theory of mind as inverse reinforcement learning. *Current Opinion in Behavioral Sciences*, 29:105–110, 2019.

Hyunwoo Kim, Melanie Sclar, Xuhui Zhou, Ronan Le Bras, Gunhee Kim, Yejin Choi, and Maarten Sap. Fantom: A benchmark for stress-testing machine theory of mind in interactions. *arXiv preprint arXiv:2310.15421*, 2023.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pages 5338–5348. PMLR, 2020.

Michal Kosinski. Theory of mind may have spontaneously emerged in large language models. *Proceedings of the National Academy of Sciences*, 121(34):e2311440121, 2024.

Harold W Kuhn. A simplified two-person poker. *Contributions to the Theory of Games*, 1 (97-103):2, 1950.

Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Pérolat, David Silver, and Thore Graepel. A unified game-theoretic approach to multiagent reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

Junkang Li, Bruno Zanuttini, and Véronique Ventos. Opponent-model search in games with incomplete information. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 9840–9847, 2024.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Neural Information Processing Systems (NIPS)*, 2017.

Anita Mahinpei, Justin Clark, Isaac Lage, Finale Doshi-Velez, and Weiwei Pan. Promises and pitfalls of black-box concept learning models. *arXiv preprint arXiv:2106.13314*, 2021.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

Christoph Molnar. *Interpretable machine learning*. Lulu. com, 2020.

Zepeng Ning and Lihua Xie. A survey on multi-agent reinforcement learning and its application. *Journal of Automation and Intelligence*, 3(2):73–91, 2024.

Ini Oguntola. Fast multi-agent gridworld environments for gymnasium, 2023. URL `https://github.com/ini/multigrid`.

Ini Oguntola, Dana Hughes, and Katia Sycara. Deep interpretable models of theory of mind.

In *2021 30th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN)*, pages 657–664. IEEE, 2021.

Ini Oguntola, Joseph Campbell, Simon Stepputtis, and Katia Sycara. Theory of mind as intrinsic motivation for multi-agent reinforcement learning. In *Proceedings of the First Workshop on Theory of Mind in Computation*, 2023.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017.

Julien Perolat, Bart De Vylder, Daniel Hennes, Eugene Tarassov, Florian Strub, Vincent de Boer, Paul Muller, Jerome T Connor, Neil Burch, Thomas Anthony, et al. Mastering the game of stratego with model-free multiagent reinforcement learning. *Science*, 378(6623):990–996, 2022.

David Premack and Guy Woodruff. Does the chimpanzee have a theory of mind? *Behavioral and Brain Sciences*, 1(4):515–526, 1978.

Hilary Putnam. Psychological predicates. *The Journal of Philosophy*, 64(14):651–660, 1967.

Hilary Putnam. The nature of mental states. In *The language and thought series*, pages 223–231. Harvard University Press, 1980.

Neil Rabinowitz, Frank Perbet, Francis Song, Chiyuan Zhang, SM Ali Eslami, and Matthew Botvinick. Machine theory of mind. In *International conference on machine learning*, pages 4218–4227. PMLR, 2018.

Roberta Raileanu, Emily Denton, Arthur Szlam, and Rob Fergus. Modeling others using oneself in multi-agent reinforcement learning. In *International conference on machine learning*, pages 4257–4266. PMLR, 2018.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *The Journal of Machine Learning Research*, 21(1):7234–7284, 2020.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

Jos Samson, Hong Liu, and Wengui Weng. Brain-inspired model of theory of mind. *Frontiers in Neurorobotics*, 14:60, 2020.

Tuomas Sandholm. Solving imperfect-information games. *Science*, 347(6218):122–123, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Gesina Schwalbe. Concept embedding analysis: A review. *arXiv preprint arXiv:2203.13909*, 2022.

Melanie Sclar, Graham Neubig, and Yonatan Bisk. Symmetric machine theory of mind. In *International Conference on Machine Learning*, pages 19450–19466. PMLR, 2022.

John R Searle. Minds, brains, and programs. *Behavioral and brain sciences*, 3(3):417–424,

1980.

Rohin Shah, Noah Gundotra, Pieter Abbeel, and Anca Dragan. On the feasibility of learning, rather than assuming, human biases for reward inference. In *International Conference on Machine Learning*, pages 5670–5679. PMLR, 2019.

Lloyd S Shapley. Stochastic games. *Proceedings of the national academy of sciences*, 39(10): 1095–1100, 1953.

Piyush K Sharma, Rolando Fernandez, Erin Zaroukian, Michael Dorothy, Anjon Basak, and Derrik E Asher. Survey of recent multi-agent reinforcement learning algorithms utilizing centralized training. In *Artificial intelligence and machine learning for multi-domain operations applications III*, volume 11746, pages 665–676. SPIE, 2021.

Herbert A Simon and Allen Newell. *Human Problem Solving*. Prentice-Hall, 1972.

Dag Tjøstheim, Håkon Otneim, and Bård Støve. Statistical dependence: Beyond pearson's ρ. *Statistical science*, 37(1):90–109, 2022.

Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *nature*, 575(7782):350–354, 2019.

Braath Waate. stratego: Demon of ignorance stratego game. `https://github.com/braathwaate/stratego`, 2012. GitHub repository. Accessed: 2025-09-27.

Jianrui Wang, Yitian Hong, Jiali Wang, Jiapeng Xu, Yang Tang, Qing-Long Han, and Jürgen Kurths. Cooperative and competitive multi-agent systems: From optimization to games. *IEEE/CAA Journal of Automatica Sinica*, 9(5):763–783, 2022.

Ying Wen, Yaodong Yang, Rui Luo, Jun Wang, and W Pan. Probabilistic recursive reasoning for multi-agent reinforcement learning. In *7th International Conference on Learning Representations, ICLR 2019*, 2019.

Zaiwen Wen and Wotao Yin. A feasible method for optimization with orthogonality constraints. *Mathematical Programming*, 142(1):397–434, 2013.

Heinz Wimmer and Josef Perner. Beliefs about beliefs: Representation and constraining function of wrong beliefs in young children's understanding of deception. *Cognition*, 13(1):103–128, 1983.

Yi Zeng, Yuxuan Zhao, Tielin Zhang, Dongcheng Zhao, Feifei Zhao, and Enmeng Lu. A brain-inspired model of theory of mind. *Frontiers in Neurorobotics*, 14:60, 2020.

Kaiqing Zhang, Zhuoran Yang, and Tamer Başar. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of reinforcement learning and control*, pages 321–384, 2021.

Liqi Zhou, Jian Liu, Yuanshi Zheng, Feng Xiao, and Jianxiang Xi. Game-based consensus of hybrid multiagent systems. *IEEE Transactions on Cybernetics*, 53(8):5346–5357, 2022.

Wentao Zhu, Zhining Zhang, and Yizhou Wang. Language models represent beliefs of self and others. *arXiv preprint arXiv:2402.18496*, 2024.