ToMCAT: Theory-of-Mind for Cooperative Agents in Teams via Multiagent Diffusion Policies

Pedro Sequeira
SRI International
Menlo Park, USA
pedro.sequeira@sri.com

Vidyasagar Sadhu
SRI International
Menlo Park, USA
srikanthvidyasagar.sadhu@sri.com

Melinda Gervasio
SRI International
Menlo Park, USA
melinda.gervasio@sri.com

ABSTRACT

In this paper we present ToMCAT (Theory-of-Mind for Cooperative Agents in Teams), a new framework for generating ToM-conditioned trajectories. It combines a meta-learning mechanism, that performs ToM reasoning over teammates' underlying goals and future behavior, with a multiagent denoising-diffusion model, that generates plans for an agent and its teammates conditioned on both the agent's goals and its teammates' characteristics, as computed via ToM. We implemented an online planning system that dynamically samples new trajectories (replans) from the diffusion model whenever it detects a divergence between a previously generated plan and the current state of the world. We conducted several experiments using ToMCAT in a simulated cooking domain. Our results highlight the importance of the dynamic replanning mechanism in reducing the usage of resources without sacrificing team performance. We also show that recent observations about the world and teammates' behavior collected by an agent over the course of an episode combined with ToM inferences are crucial to generate team-aware plans for dynamic adaptation to teammates, especially when no prior information is provided about them.

KEYWORDS

Multiagent Learning, Meta-learning, Theory-of-Mind, Diffusion Policies, Adaptive Agents, Team Modeling

1 INTRODUCTION

Recent advances in Artificial Intelligence (AI) present an opportunity to greatly improve collaboration between humans and computers to address complex tasks. Namely, AI has the ability to easily generate long courses of action that maximize joint gains, reason about uncertainty, and learn from observation. However, effective human-machine collaboration can only be achieved if artificial agents are endowed with cognitive and social mechanisms, including understanding and predicting the behaviors and preferences of others, communicating intentions, developing shared mental models and coordinating behavior accordingly [4]. In humans, such capability is referred to as forming a *Theory-of-Mind* (ToM) [2].

In this paper, we focus on developing mechanisms that allow understanding other agents'—human or artificial—underlying motivations and predict and adapt to changes in their behavior in order to maximize individual rewards in mixed-motivation settings, all *while* interacting with them. To address this problem, we propose a novel framework called ToMCAT (Theory-of-Mind for Cooperative Agents in Teams).[1] Our approach is delineated in Fig. 1 and comprises two modules: a ToM-predictive network (ToMnet, left) and a multiagent denoising-diffusion policy model (MADiff, right).

The ToMnet, based on the work of Rabinowitz et al. [20], is a neural network that learns a strong prior over agents' preferences and behavior from data generated by some known family of agents. After training, it can be used to make predictions about a particular agent given only limited data about its behavior—a process referred to as meta-learning. We extend the ToMnet framework in [20] in various ways. First, we extend it to multiagent settings, where the observer performing the ToM reasoning is itself an active member of a team; therefore predictions are made about multiple teammates, not only over a single agent, and need to be conditioned on the observer's characteristics. In addition, we assume that predictions about teammates are made solely based on the observer's own (partial) current and past observations rather than an external observation function as in [20], and therefore observations include information about all teammates, including the observer itself. Moreover, our ToMnet's outputs represent the observer's beliefs about the mental state of its teammates, and thus *all* predictions are represented as probability distributions.

The MADiff component is based on the work of Zhu et al. [31] and corresponds to a probabilistic denoising-diffusion model that generates multiagent plans. We extend MADiff by guiding the generation of plans based on aspects of both the observer agent and its teammates as encoded in latent representations from the ToMnet, allowing for adaptation via *ToM-conditioned* planning. We also provide a new online mechanism that determines *when* to replan based on a measure of misalignment between planned and real world states, allowing for efficient reuse of MADiff plans.

By combining ToMnet and MADiff, ToMCAT allows an agent not only to plan for itself but also to predict the plans of its teammates from its own observations, leveraging ToM reasoning to compensate for the lack of communication and full observability of teammates' states. Moreover, the conditionally generated behavior through MADiff allows for recursive ToM and nested beliefs, *e.g.*, allowing an agent to reason about how the different team members model each other, including the agent itself. Furthermore, conditioning plan generation on ToMnet embeddings that are trained to predict behavior characteristics of different teammates allows interacting with unknown teammates (*i.e.*, generalizability) and combining characteristics of teammates from different teams,

Proc. of the Adaptive and Learning Agents Workshop (ALA 2025), Avalos, Aydeniz, Müller, Mohammedalamen (eds.), May 19 – 20, 2025, Detroit, Michigan, USA, ala-workshop.github.io. 2025.

[1] Throughout the paper we use "teams" to refer to *groups of agents* in general. While in our experiments we focus on collaboration, ToMCAT's mechanisms are agnostic to the alignment of the agents' self interests, *i.e.*, can be used in collaborative or competitive settings. Similarly, we use "teammate" to refer to any other agent in the group.

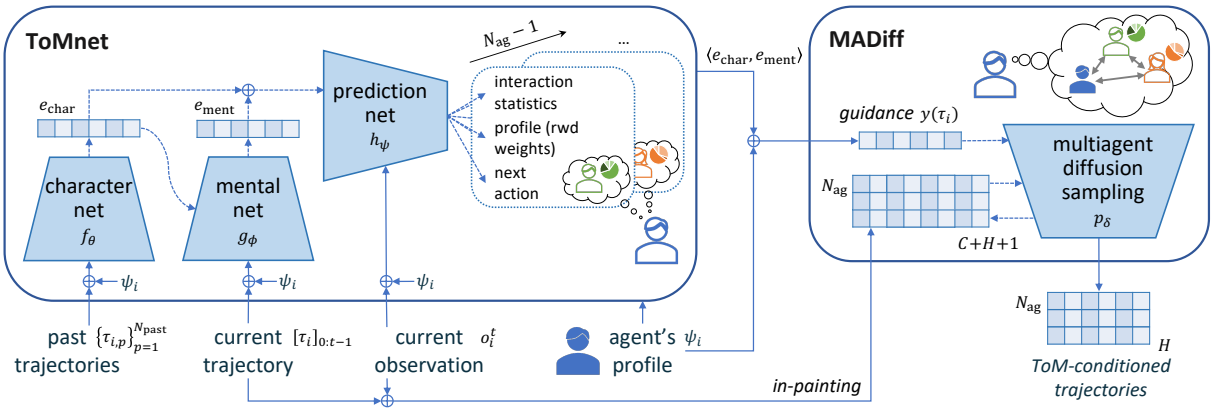


Figure 1: The ToMCAT architecture. Left: a Theory-of-Mind network (ToMnet) makes predictions about the behavior and underlying motivations of the various teammates. Right: a Multiagent Diffusion model (MADiff) generates trajectories conditioned on the ToM reasoning. ⊕ indicates a concatenation operation.

without the need to retrain either model (*i.e.*, compositionality). In summary, our contributions are as follows:

(1) A meta-learning approach to probabilistic multiagent ToM that takes into account others' preferences and behavior and the observer's own characteristics.
(2) A team behavior predictive denoising-diffusion model that allows generating multiagent plans conditioned on ToM inferences, enabling fast understanding of and adaptation to teammates' preferences and behavior, even in the absence of strong priors.
(3) A dynamic conditional replanning mechanism that allows for efficient online agent adaptation—*i.e.*, while interacting with others—by computing discrepancies in planned world states.

2 RELATED WORK

To create agents that can adapt to a diverse set of teammates, one option is to use Reinforcement Learning (RL) to train policies that maximize a reward function while treating teammates as an element of the dynamic environment. *E.g.*, in independent Q-learning (IQL) [17] an agent relies only on its observations without considering the actions of others, resulting in non-stationarity and suboptimal performance. To address this issue, one can instead use multiagent RL (MARL). In particular, centralized training and decentralized execution (CDTE) approaches have been proposed [18] where an agent has access to states and actions of other agents during training, resulting in a centralized value function from which individual policies are derived via a policy gradient algorithm [28]. However, such approaches do not support cases where agents cannot directly observe the internal states of others.

One approach to overcome that limitation is to use self-play (SP) [24], where an agent is trained to play against itself. However, this is mostly applicable to zero-sum settings or homogeneous teams, being hard to adapt to different styles of play or roles in a task. Another approach is to train single-agent policies against a known set of agent models. Namely, in population-based training (PBT) [13] one first builds a population of diverse agent models and then trains a policy against agents sampled from the population, improving the population during training, *e.g.*, using genetic algorithms. While ToMCAT follows a similar approach to PBT by learning from a diverse set of teammates, we train a single model to learn the distribution of possible team behaviors in a task such that an agent can be assigned any role and adapt to the behavior of arbitrary teammates, including sub-optimal ones w.r.t. the task.

Another related body of work is that of ad hoc teamwork [27], where an agent needs to collaborate effectively with other agents without prior coordination or explicit pre-training. Some approaches assume that the task is known beforehand, *e.g.*, the ODITS [6] framework trains a single-agent RL policy by considering all teammates as part of the environment, learning a teamwork situation encoder whose latent representation conditions the policy for online adaptation, akin to the way we condition diffusion policies based on ToM embeddings. Other approaches try to infer the underlying collaborative task itself, assumed unknown to the ad hoc agent. *E.g.*, the ATPO framework [21] casts ad hoc teamwork as identifying the task/teammate from a set of known tasks from histories of observations, then selecting the appropriate action from the corresponding pre-computed task policy. One disadvantage of RL-based approaches is that if an agent's objectives need to change during a task, *e.g.*, for role reassignment, we need to re-train the policy, which can be computationally expensive. In contrast, ToMCAT does not train (a set of) RL policies explicitly optimizing for a single reward function—rather, it uses denoising-diffusion policies to generate plans guided by both the agent's own preferences, which might not be aligned with those of its teammates, and teammates' perceived behavior characteristics. In addition, our models make predictions about the underlying attributes of teammates, *i.e.*, from a family of parameterized agents, rather than trying to identify a teammate from a discrete set.

There is also a wealth of works trying to imbue ToM capabilities in autonomous agents (see, *e.g.*, [7, 22]). Here we mention those that are mostly related to our approach on conditioning agent behavior on predictions of others' underlying motivations and observed behavior. Some, like ours, adopt the framework of I-POMDPs [5, 8] (an introduction to which is provided in Sec. 3.1), where models of others are explicitly added as part of the decision-making space

of an agent. *E.g.*, the I-POMDP-Net [9] integrates a planning algorithm to approximately solve POMDPs called QMDP [16] and beliefs about models of others into a single neural network for ToM based multi-agent decision making. The PsychSim framework [19] is a multiagent-based simulation tool based on I-POMDPs for modeling social interaction, where agents form beliefs about the world and maintain recursive models of other agents (ToM). It uses exact Bayesian inference to update the beliefs upon new observations, allowing an agent to compute optimal actions conditioned on the ToM models through forward planning. Other approaches employ inverse RL (IRL) to model the motivations of others from observed behavior, which can be seen as a form of ToM [15]. *E.g.*, the MIRL-ToM approach in [29] allows recovering the reward functions used by different agents given trajectories of their behavior by explicitly updating posterior distributions over a set of known models via Bayesian ToM, which allows modeling adaptation to others without requiring a joint equilibrium solution. Overall, these approaches only work with fixed agent profiles and/or a fixed number of teammates, and the (explicit) Bayesian belief update can become computationally intractable as the number of teammates grows. In contrast, ToMCAT does not use an explicit planning module to solve for the I-POMDP and instead learns a distribution of team behaviors from a rich dataset and uses diffusion policies for fast online adaptation to different teammates.

3 TOMCAT

As depicted in Fig. 1, our approach uses two models to generate ToM-conditioned behavior for online prediction and adaptation in the context of teams. The Theory-of-Mind network (ToMnet, left) is responsible for making predictions about teammates' behavior conditioned on their observed behavior, while a Multiagent Diffusion network (MADiff, right) generates agent trajectories for all team members conditioned on the ToM reasoning.

3.1 Preliminaries

We model the problem of an agent modeling and adapting to the behavior of others using the formalism of interactive partially observable Markov decision processes (I-POMDPs) [9]. I-POMDP allows unifying the ToMnet [20] and MADiff [31] approaches—it supports modeling different types or families of agents in the same task as in the ToMnet approach, and characterizing the decision-making problem of individuals in multiagent settings as the MADiff approach. However, it departs from the latter in that it models self-interested agents and is thus applicable to general-sum tasks. Formally, an I-POMDP for an agent i is defined as the tuple $\mathcal{M} = \left(\check{S}_i, A, T_i, R_i, \Omega_i, O_i\right)$, where $\check{S}_i = S \times_{j=1}^{N_{\text{ag}}} M_j$ is the set of interactive states, where S is the physical environment state space and M_j is the set of possible models for agent j representing its internal state, A is the joint action space, $R_i : \check{S}_i \times A \to \mathbb{R}$ is agent i's reward function, $T_i : S \times A \to S$ the transition function, Ω_i the observation space, and $O_i : S \times A_i \to \Omega_i$ the conditional observation function.

For a given task, we assume a family of agents $\mathcal{A} = \bigcup_i \mathcal{A}_i$, akin to player *types* in Bayesian game theory [10], where each agent type is characterized by some private attribute vector, *i.e.*, unobservable by others, containing all information relevant to its decision making. Without loss of generality, here we consider individual reward functions of the form $R_i(s^t) = \boldsymbol{\phi}(s^t)^\top \boldsymbol{\theta}_i$, consisting of linear combinations of (physical) state features, $\phi_k : S \to \mathbb{R}, k = 1, \ldots, |\boldsymbol{\phi}|$, weighted by some weight vector, $\boldsymbol{\theta}_i$. Therefore, we consider that agent types can be succinctly expressed via a *profile* function, $\psi : \mathcal{A} \to \mathbb{R}^n$, and in our experiments set $\psi_i(\mathcal{A}_i) \coloneqq \boldsymbol{\theta}_i$, *i.e.*, each agent type is motivated to optimize over different aspects of the same general task. For simplicity, we write ψ_i to denote agent i's profile. As for the models $m_j \in M_j$ of others, here we allow any representation of teammate j's private information, including predictions over its profile, beliefs or future behavior.

For training the different modules, we consider observed trajectories of the form $\tau_i = \{(o_i^t, a_i^t)\}_{t=0}^T$, where $o_i^t = O_i(a_i^{t-1}, s^t)$ and $a_i^t \sim \pi_i(o_i^t)$ are respectively agent i's observation over the global state, s^t, and its action, taken at trajectory step t. We assume that an agent's observations collected through O_i include partial information about its teammates' overt behavior that will be used to make predictions about them during task performance.[2]

3.2 Theory-of-Mind Reasoning

As mentioned earlier, our first goal is to create a mechanism capable of inferring the motivations and intent of teammates from their observed behavior. To achieve that, we follow the approach in [20], which aims to develop a machine ToM by training a neural network called a *ToMnet* to model the mental states and behaviors of other agents through meta-learning. The ToMnet learns to predict agents' future actions and infer their goals and beliefs, including false beliefs, by observing their behavior in an environment of interest.

As outlined in Fig. 1 (left), the ToMnet consists of three modules:

Character Net: processes trajectories from past episodes of an observer agent, i, acting in the context of a team, with the goal of *characterizing* the general behavioral tendencies of teammates $j \in N^{-i}$. Given past trajectories, $\{\tau_{i,p}\}_{p=1}^{N_{\text{past}}}$, and the observer's profile, ψ_i, it outputs a *character embedding*, as $e_{\text{char},i} \coloneqq \frac{1}{N_{\text{past}}} \sum_{p=1}^{N_{\text{past}}} f_\theta(\psi_i, \tau_{i,p})$,[3] where f_θ is a learned neural network. The length of past trajectories is denoted by T_{past}.

Mental Net: focuses on the current episode's trajectory so far to provide the necessary context about teammates' behavior to infer their current *mental* state. It computes a *mental embedding* as $e_{\text{ment},i} \coloneqq g_\phi(\psi_i, [\tau_i]_{t-T_{\text{cur}}:t-1}, e_{\text{char},i})$, where $[\tau_i]_{T_{\text{cur}}:t-1}$ comprises agent i's last T_{cur} timesteps in the current trajectory and g_ϕ is another learned neural network.

Prediction Net: uses the outputs of the Character and Mental Nets to update the observer's beliefs about the teammates' behavior and their underlying characteristics. In particular, the Prediction Net is capable of inferring each teammate's next-step action distribution, $\widehat{\pi}_j \coloneqq \Delta(A_j)$, and the distribution over the sign of its profile elements, $\widehat{\psi}_j \coloneqq \Delta(\{-1, 0, 1\}^n)$, given the output of $d_\gamma(\psi_i, o_i^t, e_{\text{char},i}, e_{\text{ment},i})$, where d_γ is another learned neural network serving as the shared torso for the different prediction heads. The Prediction Net can also estimate other statistics of

[2] Such information *does not* include teammates' reward functions, policies or identifiers.
[3] In contrast to [20], we average the embedding for each past trajectory when computing e_{char} to allow for variable number of past trajectories.

teammates' future behavior, such as the likelihood of picking up objects in the environment or state occupancies.

The ToMnet is parametrized by θ, ϕ and γ and is trained end-to-end. It leverages meta-learning to construct a *general ToM*—the model's parameters encode a prior for modeling diverse agent types—while dynamically updating an *agent-specific ToM* — the model's predictions update a posterior over the characteristics of a particular teammate from recently observed behavior [20]. Our extension to multiagent settings results in a single model capable of forming predictions about multiple agent types from the perspective of different types of observer agents. Since others' behavior is contingent on an observer's own behavior, the observer's profile is used as input to condition the predictions of the different modules.

3.3 Multiagent Diffusion Policies

The second component of our architecture depicted in Fig. 1 (right) allows for learning multiagent policies conditioned on the ToM information provided by the ToMnet. The idea is to learn policies that can be used by an agent to adapt to its teammates online, while interacting with them to perform some task. To achieve that, we adopted *Multiagent Diffusion* (MADiff) [31], a framework that extends single-agent diffusion policies [1, 14] for flexible behavior planning in teams. MADiff trains a conditional generative behavior model from diverse data and treats multiagent planning as sampling from that model. Furthermore, we adopt a *decentralized* execution paradigm, where we assume that each agent makes its own decision without any communication with other agents, based only on local information from its own observations.

MADiff is defined as a multiagent offline learning problem, where we have access to a static dataset, $\mathcal{D}$, containing joint trajectories of the form $\boldsymbol{\tau} := [\tau_i]_{i=0}^{N_{\text{ag}}}$. Since action sequences tend not to be smooth [1], we only model observed state sequences as in [31]. Namely, given a joint trajectory $\boldsymbol{\tau} \in \mathcal{D}$, we sample sequences of joint observations, $\hat{\boldsymbol{\tau}} := \left[(o_i^t, o_i^{t+1}, \ldots, o_i^{t+H-1})\right]_{i=0}^{N_{\text{ag}}}$, where $t \sim \mathcal{U}(1, T)$ is the time at which an observation was made by agent i, sampled uniformly at random, and H is the diffusion horizon. To retrieve actions from generated plans we use $a_i^t := I_\xi(o_i^t, o_i^{t+1})$, where I_ξ is an inverse dynamics model shared amongst agents that is independent from but simultaneously trained with p_δ.

To condition trajectory generation on ToM reasoning, we sample from perturbed distributions of the form $\hat{p_\delta}(\hat{\boldsymbol{\tau}}) \propto p_\delta(\hat{\boldsymbol{\tau}})y(\boldsymbol{\tau})$, where $y(\boldsymbol{\tau})$ is an additional input to the diffusion model encoding particular properties of joint trajectory $\boldsymbol{\tau}$. Following the decentralized approach, in ToMCAT perturbations are made from the perspective of an observer agent i, which we denote using $y(\tau_i)$, and can include any of the following information:

Profile: the observer agent's profile, ψ_i for which the joint trajectories are being generated.

Character: the embedding provided by the Character Net characterizing teammates $j \in N^{-i}$, *i.e.*, $e_{\text{char},i}\left(\{\tau_{i,p}\}_{p=1}^{N_{\text{past}}}\right)$, where $\tau_{i,p} \in \mathcal{D}$ are *other* trajectories sampled from the dataset in which agent i interacted with the *same* (types of) teammates.

Mental: the Mental embedding characterizing teammates' mental states, *i.e.*, $e_{\text{ment},i}([\tau_i]_{0:t-1})$.

Algorithm 1: Online Dynamic Conditional Replanning

Input: N, i, ψ_i, $\mathcal{D}$, N_{past}, f_θ, g_ϕ, η, λ, ϵ_δ, I_ξ, ω, C, H

Initialize $e_{\text{char}} \leftarrow e_{\text{char},i}\left(\{\tau_{i,p} \sim \mathcal{D}\}_{p=1}^{N_{\text{past}}}\right)$, $t \leftarrow 0$, $\hat{o_i} \leftarrow \varnothing$, $h \leftarrow \texttt{Queue}(C, \varnothing)$, $\tau_{\text{plan}} \leftarrow \texttt{Queue}(H, \varnothing)$;
while not done **do**
  Agent i observes o_i; $\texttt{enqueue}(h, o_i)$;
  if $\text{empty}(\tau_i)$ || $\delta_{obs}(o_i, \hat{o_i}) > \lambda$ **then**
    $e_{\text{ment}} \leftarrow e_{\text{ment},i}(h)$; $y(\tau_i) \leftarrow [e_{\text{char}}, e_{\text{ment}}, \psi_i]$;
    Initialize $\hat{\boldsymbol{\tau}}^K \sim \mathcal{N}(\mathbf{0}, \alpha \boldsymbol{I})$;
    for $k = K \ldots 1$ **do**
      $\hat{\boldsymbol{\tau}}_i^k[: C+1] \leftarrow h$;
      $\hat{\epsilon} \leftarrow \epsilon_\delta(\hat{\boldsymbol{\tau}}^k, k) + \omega\left(\epsilon_\delta(\hat{\boldsymbol{\tau}}^k, y(\tau_i), k) - \epsilon_\delta(\hat{\boldsymbol{\tau}}^k, k)\right)$;
      $\hat{\boldsymbol{\tau}}^{k-1} \leftarrow \texttt{denoise}(\hat{\boldsymbol{\tau}}^k, \hat{\epsilon}, k)$;
    $\hat{\tau}_i \leftarrow \hat{\boldsymbol{\tau}}_i^0[C+1:]$; $\tau_{\text{plan}} \leftarrow \texttt{Queue}(H, \hat{\tau}_i)$; $\hat{o}_i \leftarrow o_i$;
  $\hat{o}_i' \leftarrow \texttt{dequeue}(\tau_i)$; $a_i = I_\xi(\hat{o}_i, \hat{o}_i')$; $\hat{o}_i \leftarrow \hat{o}_i'$;
  Execute a_i, let teammates $j \in N^{-i}$ act;

Returns: any measure of behavior optimality that we wish to condition trajectory generation on, *e.g.*, the agent's individual reward, $R_i(\tau_i)$, or the task reward common to all agents.

In situations where we want to condition trajectory generation on multiple factors, we concatenate them to form $y(\tau_i)$. In addition, we follow [1, 14, 31] and constrain trajectory generation on the agent's current observation, o_i^t, and on the C previous steps, in a process akin to pixel *in-painting* in image generation [25]. This leads to an augmented trajectory $\hat{\boldsymbol{\tau}} := \left[(o_i^{t-C}, \ldots, o_i^t, o_i^{t+1}, \ldots, o_i^{t+H})\right]_{i=0}^{N_{\text{ag}}}$.

The MADiff model is parametrized by δ and ξ which are simultaneously trained by sampling trajectories from dataset $\mathcal{D}$ and randomly selecting an agent to be the "observer", while other agents are the teammates. Since we follow a decentralized paradigm, the aforementioned in-painting information for the teammates is zeroed-out since the observer does not have access to their observations. Notwithstanding, MADiff is trained to predict the future observations of *all* agents given its local observations.

3.4 Dynamic Replanning

To generate plans (trajectories), we use classifier-free guidance [12] with low temperature sampling, where we assume an online setting where an agent i is both observing and interacting with teammates in the context of some task. Generating plans using diffusion models can be computationally intensive [1, 14], so Janner et al. [14] proposed a warm-start solution where a plan generated for previous timesteps is reused by first diffusing it for a small number of steps—a fraction of the original diffusion steps—and then denoising the resulting noise for the same number of steps, thereby generating a new plan. However, this approach assumes that plans taken at consecutive timesteps are consistent with one another. Because ToMCAT generates multiagent plans in a decentralized manner, the likelihood that plans remain consistent in consecutive steps is much lower since it would require teammates' actions to be consistent with the predicted trajectory.

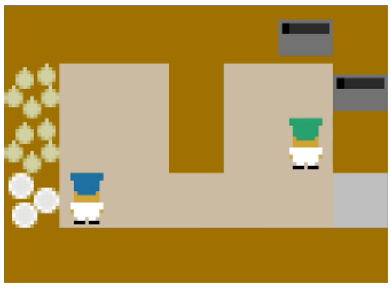


Figure 2: The cooking domain used in the experiments.

As such, we propose a *dynamic replanning* approach where an agent follows the actions in a generated plan while its predictions, including about its teammates, remain consistent, and replans whenever they are inconsistent or the plan is depleted. Moreover, to reduce the number of denoising steps, we use DDIM sampling [26]. This results in the online planning (sampling) approach delineated in Algorithm 1, corresponding to a modified version of the algorithm in [1]. In addition to computing and using e_{char}, e_{ment} and ψ_i as conditioning variables, the other novelty resides in line 4, where we compute the difference between the current observation and the observation predicted in the plan, using a suitable metric, δ_{obs}. If the observation difference is higher than a predefined threshold, λ, we generate a new plan, otherwise we continue using the actions from the previously computed one. See Sec. A.1 of the Appendix for a detailed description of the algorithm.

4 EXPERIMENTS & RESULTS

In this section, we detail the experiments carried out to assess the usefulness of ToMCAT in generating dynamic, ToM-conditioned strategies for online adaptation to diverse teammates in collaborative tasks. We detail our experimental scenario, how we trained various RL agents to generate ground-truth behavioral data, and how we trained ToMnet and MADiff models therefrom.[4] We then detail the different experiments and analyze the main results.

4.1 Data Collection

To test our approach, we designed a scenario in the *Overcooked-AI* simulation platform [3],[5] which is a benchmark domain for research in human-AI collaboration. The custom scenario used in our experiments is illustrated in Fig. 2. Briefly, the task objective is for a team of $N_{\text{ag}} = 2$ agents to collaborate in order to maximize the number of soups served within a time limit. Agents need to collect onions, place them in pots, wait for soups to cook, then serve them in the corresponding station. Each agent has $|A_i| = 6$ available actions at each timestep: they can move in four cardinal directions, perform a "no-op" action, or "interact" with an object in the environment (*e.g.*, pickup or place items). Agents' initial locations in the environment are randomized. Our custom scenario, inspired by the environment in [30], adds a bottleneck (middle counter) to increase the usefulness of specialized roles in the task and decrease the likelihood that agents can perform the task on their own, requiring effective coordination for success.

[4]Implementation and training details are provided in the Appendix.
[5]https://github.com/HumanCompatibleAI/overcooked_ai

As mentioned earlier, here we consider reward functions consisting of linear combinations of state features, and set the profile for an agent i as $\psi_i := \boldsymbol{\theta}_i$, where $\boldsymbol{\theta}_i$ is the vector of weights associated with each feature. We designed a set of representative profiles capturing varied sub-goals in the Overcooked task (see Table A.1 in the Appendix the corresponding weights, $\boldsymbol{\theta}_i$, and in Sec. A.2 a description of each reward feature). The profiles range from specialized roles, *i.e.*, Cook and Server, to agents that are not task-oriented, *i.e.*, Random and Follower, where the latter can result in antagonistic behavior, *e.g.*, they can unintentionally block the other agent. Our goal was to design profiles that lead to distinct behaviors in the task and require different types of collaboration.[6]

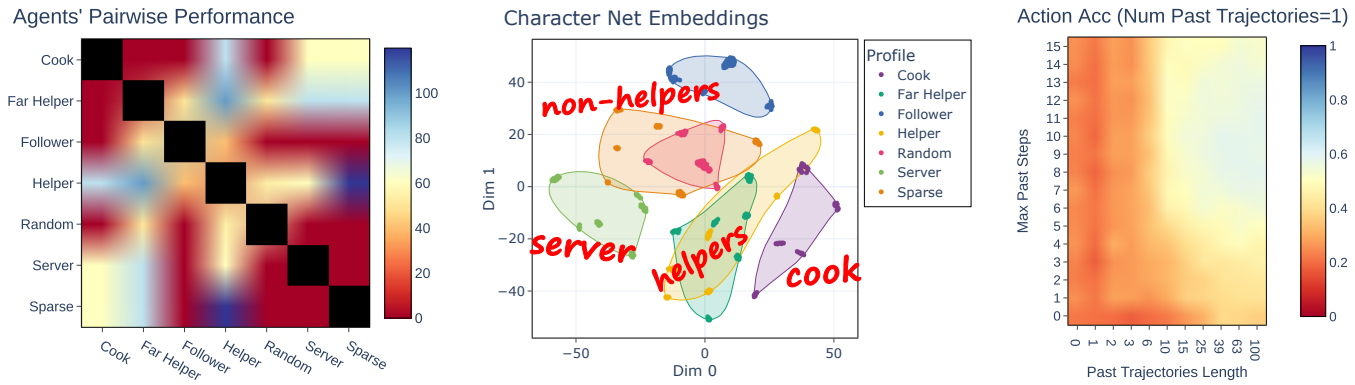
We wanted the dataset $\mathcal{D}$ used to train our ToM and planning modules to be diverse, showing different ways of satisfying combinations of conditions, $y(\tau_i)$, so we trained decentralized MARL policies for each pair of profiles in Table A.1 using PPO [23] (see details in Sec. A.3 of the Appendix), resulting in 21 teams. We then used the trained policies to generated the dataset, $\mathcal{D}$, by rolling out trajectories 1 000 trajectories for each agent pair, with a maximum trajectory length of $T = 200$ timesteps and a policy exploration (softmax) temperature of 0.3 to ensure some stochasticity in agents' action selection, resulting in $|\mathcal{D}| = 21\,000$ joint trajectories. Fig. 3a shows the pairwise performance of the different agent pairs in the dataset, as measured by the mean task reward, R_{task}, corresponding to a reward of $R_{\text{task}} = 20$ when a soup delivered, and 0 otherwise. As we can see, the dataset represents a wide range of different team behavior, as desired for training our predictive models. It also shows the impact of pairing an agent trained to optimize for some reward function (*i.e.*, profile) with different types of teammates, thus showing inter-dependencies in the task leading to varying degrees of performance (see, *e.g.*, results for the Sparse agent).

4.2 Model Training

We trained the ToMnet in a supervised manner, where we first sampled a profile, $\psi_i \sim \Psi$, where Ψ is the set of possible profiles listed in Table A.1. We set agent i to be the observer, and sampled a joint trajectory, $\tau \sim \mathcal{D}$ where the team, $N := \{i, j\}$, is composed of the observer and a teammate, j. We then selected a timestep, $t \sim \mathcal{U}(1, T)$, select the "current" observation, o_i^t, and set the "current" trajectory to $[\tau_i]_{t-T_{\text{cur}}:t-1}$, where $T_{\text{cur}} = 10$. We further selected $N_{\text{past}} = 4$ past trajectories, $\{\tau_{i,p} \sim \mathcal{D}\}_{p=1}^{N_{\text{past}}}$, that capture prior information about the teammate, *i.e.*, where each $\tau_{i,p}$ contains data of the same team, N. All these data were used as inputs to the ToMnet. Different statistics were then computed from $[\tau_i]_{t+1:T}$, which along with the sign of each of the teammate's profile weights, $\boldsymbol{\theta}_j$, and its next action, a_j^t, formed the targets of the Prediction net. See Sec. A.4 of the Appendix for details.

To train the MADiff component, we sampled joint trajectories $\tau \sim \mathcal{D}$ similarly to how we trained the ToMnet, but augmented trajectories by computing the ToM embeddings, $e_{\text{char},i}$ and $e_{\text{ment},i}$, for each timestep using the trained ToMnet, so they could be used as conditioning variables in $y(\tau_i)$. In addition, when sampling a trajectory for training, we used $C = 16$ steps to constrain trajectory generation using in-painting, and the subsequent $H = 64$ steps

[6]Although we design a small set of representative profiles, the features form a space of possible profiles, corresponding to the admissible family of agents in the task, $\mathcal{A}$.



(a) MARL Pairwise Performance **(b) Character Embeddings** **(c) ToMnet Action Accuracy**

Figure 3: MARL and ToMnet training results.

as the planning horizon. As explained earlier, here we consider decentralized execution, where observers have access to their local observations, but generate predictions for the future behavior of the whole team. To ensure that, we masked out teammate observations when using history in-painting. After training, we used DDIM sampling for $k = 15$ diffusion steps to generate plans in all experiments. See Sec. A.5 of the Appendix for more details.

4.3 Model Analysis

We now provide some insights from training the ToMnet and MADiff models. Regarding the ToMnet, Fig. 3b plots the 2D t-SNE representations of the character embeddings, e_{char}, computed for the trajectories in $\mathcal{D}$. We can see that the Character Net learns different internal representations for different profile "families" denoting distinct behavior tendencies in the task. As in [20], this shows that the ToMnet learns useful representations about teammates' characteristics. However, because we condition ToMnet predictions on the observer's own profile, our results further show that it distinguishes between different types of interacting partners, who may require different types of collaboration. Fig. 3c plots the prediction accuracy of the teammate's next action when varying the length of past trajectories, T_{past} and the number of previous steps in the current trajectory T_{cur}, provided to the ToMnet, while fixing $N_{\text{past}} = 1$. We observe that with a prior trajectory of only approximately $T_{\text{past}} = 40$ steps and information on the last $T_{\text{cur}} = 10$ steps, the ToMnet achieves maximal performance. The main insight is that the ToMnet does not require much past information about a teammate in order to make accurate predictions about its underlying behavior characteristics. This is an indication that, even in the absence of prior data about a teammate, an observer agent could potentially use observations collected during a trajectory and feed those as past trajectories to compute character embeddings.

Regarding the MADiff model, we studied the impact of the different ToM conditioning variables and history in-painting on both task and individual performance, which is measured as the cumulative task reward, R_{task}, and individual reward, R_i, received over the course of an episode. The latter is important because it is a measure of how well an agent adheres to the role it was assigned for the task. Table 1 shows the mean performance for the different types of conditioning, $y(\tau_i)$, when models were trained with ($C = 16$) and without ($C = 0$) history in-painting. Episodes were generated by having two ToMCAT agents, equipped with the same MADiff model but different profiles, independently generate a plan at each time step and perform the first action of the plan in the environment.

In general, we observe that the history information provides the necessary context for prediction in models conditioned by ToM-based variables, *i.e.*, e_{char}, e_{ment} and ψ_i, leading to significantly better performance compared to the no-history counterparts. Furthermore, by looking at the different conditioning sets for the history case, we see that the more ToM "features" are used, the better the performance is. We note that even though conditioning only on task rewards, *i.e.*, R_{task}, attains the highest team performance, it leads to very poor individual performance. We hypothesize that because the MADiff model is unaware of individual preferences, it learns to mimic the behavior of the best team, leading to improved task performance but failing to carry out its individual responsibilities. In addition, as will be shown in our experiments, conditioning on returns works only if *both* agents use the same MADiff model.

4.4 Agent Experiments

We now address different research questions to assess the usefulness of online dynamic replanning and the extent to which ToMCAT agents, equipped with different ToM capabilities, can adapt to teammates, both in the presence and absence of priors about their behavioral characteristics. For all experiments reported below and for each test condition, we generated 500 episodes of 200 timesteps each, where we paired a ToMCAT agent with a RL agent, randomly assigning profiles to each agent at the start of an episode. In addition, in each episode, the policy of the RL agent was selected from the set trained via MARL, as detailed in Sec. 4.1, corresponding to the policy that was concurrently trained with another RL agent using the same profile as the ToMCAT agent. In practice, this means that the RL agent "knows" the profile of the ToMCAT agent and selects the corresponding optimal policy.

4.4.1 Dynamic Replanning. We want to address the following research question: *"Can the ToMCAT system perform well when deployed in a dynamic environment without requiring replanning at every step?"* To address that question, unlike prior works that measure CPU time on a specific platform, here we use the number of

Table 1: Effects of conditioning variables, $y(\tau_i)$, on MADiff training. We also compare the use of history ($C = 16$) vs. no history ($C = 0$) for in-painting constraining. Reported values correspond to averages of cumulative task and individual rewards over 500 episodes of length 200 timesteps. Errors correspond to the 95% CI for the mean.

	Task Reward		Individual Reward	
Conditioning, $y(\tau_i)$	**No History**	**History**	**No History**	**History**
[ ] *(unconditioned)*	6.68 ± 0.82	9.44 ± 1.44	−54.33 ± 0.58	−52.98 ± 0.76
$[R_{\text{task}}]$	**19.36 ± 1.30**	**34.60 ± 1.61**	−85.57 ± 1.05	−87.84 ± 0.78
$[\psi_i]$	4.68 ± 0.83	9.16 ± 1.28	−16.83 ± 0.29	−33.19 ± 0.39
$[e_{\text{char}}]$	6.20 ± 1.09	18.92 ± 1.77	−11.88 ± 0.25	−7.90 ± 0.34
$[e_{\text{ment}}]$	9.48 ± 1.31	12.12 ± 1.51	−13.24 ± 0.30	−28.03 ± 0.43
$[e_{\text{char}}, e_{\text{ment}}]$	10.04 ± 1.36	22.76 ± 1.89	**−9.44 ± 0.24**	−2.52 ± 0.31
$[\psi_i, e_{\text{char}}, e_{\text{ment}}]$	8.20 ± 1.28	24.88 ± 1.92	−10.67 ± 0.29	**1.66 ± 0.30**
$[R_{\text{task}}, \psi_i, e_{\text{char}}, e_{\text{ment}}]$	7.60 ± 0.95	19.20 ± 1.52	−19.63 ± 0.35	−11.05 ± 0.34

(a) Task Reward **(b) Individual Reward** **(c) Profile Probability**

Figure 4: (a)–(b): impact of various replanning schemes on the agents' task and individual cumulative rewards. Covariance ellipses represent the 95% CI of the mean. (c): mean probability over the course of trials of the ground-truth profile of the teammate under the ToMnet prediction in the presence (*Prior*) vs. absence (*No Prior*) of prior information about the teammate.

planning steps as a measure of computational "resource usage" and assess its impact on task and individual agent performance.

In this experiment, we used the MADiff model trained with history (*i.e.*, $C = 16$) conditioned on all the ToM variables (*i.e.*, $y(\tau_i) := [\psi_i, e_{\text{char}}, e_{\text{ment}}]$) as the base model for the ToMCAT agent. To test the usefulness of the online dynamic replanning capability detailed in Sec. 3.4, we compare it against replanning at fixed time intervals, namely: at every timestep (*Always*), every 10 timesteps (*10 Steps*), and every $H = 64$ timesteps (*Horizon*). In order to compute the character embedding, e_{char}, the ToMCAT agent is provided $N_{\text{past}} = 4$ past trajectories sampled from $\mathcal{D}$, where an agent with the same profile interacted with an agent with the teammate's profile.

Figs. 4a and 4b show the average task and individual performance compared to the plan count for different replanning schemes (see Table A.2 of the Appendix for the corresponding numeric data). As we can see, there is a trade-off between task performance and resource usage, and this trade-off is not linear. In particular, *Dynamic* replanning makes the best use of resources compared to fixed-interval schemes, achieving approximately the same level of performance as agents that replan at every step (*i.e.*, *Always*), but using only about one third of resources. This shows the advantage of our replanning mechanism that automatically detects the divergence between projected and actual states of the world and replans accordingly, using the entire plan if everything goes as expected.

4.4.2 Known Teammates. In this experiment, we want to address the question: *"Can ToM reasoning improve team performance in the presence of strong teammate priors?"* Here we want to understand whether ToM is important for ToMCAT agents when paired with teammates for whom prior behavior demonstrations are provided, corresponding to supplying $N_{\text{past}} = 4$ past trajectories as detailed in the previous experiment. To address that question, we test ToMCAT agents equipped with MADiff models under different conditioning variables, $y(\tau_i)$, namely: *Unconditioned*, *Returns* ($[R_{\text{task}}]$), *Profile* ($[\psi_i]$), and *ToM* ($[\psi_i, e_{\text{char}}, e_{\text{ment}}]$). As an upper bound on performance, we also test against a team of RL agents, where at each episode we sample a pair of policies that were concurrently trained via MARL, corresponding to the *Know RL* condition.

The results are shown in Table 2, where we measure resource usage (*i.e.*, plan count) and the task and individual performance as in the previous experiment. Overall, we observe that agents with ToM capabilities (*i.e.*, *ToM)* attain better task performance, adhere the best to their pre-assigned roles (compare with *Profile*), and use

Table 2: ToM impact in the presence of teammate priors.

Condition	Plan Count	Task Rwd	Indiv. Rwd
Unconditioned	78.63 ± 3.62	10.88 ± 1.92	−30.39 ± 8.83
Returns	99.71 ± 3.14	11.36 ± 1.56	−37.54 ± 9.75
Profile	87.28 ± 3.99	14.24 ± 2.28	−29.29 ± 9.57
ToM	**64.89 ± 3.82**	23.52 ± 2.89	−1.74 ± 6.90
RL Known	-	**41.96 ± 3.47**	**23.85 ± 6.71**

Table 3: ToM impact in the absence of teammate priors.

Condition	Plan Count	Task Rwd	Indiv. Rwd
Unconditioned	**79.79 ± 3.61**	10.64 ± 1.83	−34.32 ± 9.72
Returns	105.11 ± 3.09	10.88 ± 1.50	−41.37 ± 10.58
Profile	87.00 ± 3.95	**14.24 ± 2.33**	−31.29 ± 9.79
ToM	83.97 ± 3.81	12.52 ± 2.19	−29.25 ± 9.34
RL Unknown	-	10.36 ± 1.88	**−11.05 ± 5.76**

less resources. In contrast, in this experiment, agents conditioned only to maximize task returns fail to correctly perform the task as they are unable to adapt to the teammate's characteristics. In addition, although the pure RL team unsurprisingly achieves the best performance, we note that this would require agents to know their exact teammates' profiles and co-training for each possible pair, while the ToMCAT agent can use a single ToMnet and MADiff model and adapt to a multitude of teammate behaviors.

4.4.3 Unknown Teammates. Here we want to address the question: *"Can ToMCAT agents learn from and adapt to unknown teammates?"* We adopted a methodology similar to that of the previous experiment, but where the *ToM* agent was *not* provided any past trajectories about its teammate, *i.e.*, $N_{\text{past}} = 0$. Instead, for the first 100 timesteps of an episode, the agent updated a buffer of observations, and used that to compute the character embedding, e_{char}. We also used a different baseline against which to compare the MADiff agents, namely a team of RL agents that did not "know" their teammate. Recall from Sec. 4.1 that we co-trained RL policies via MARL for all pairs of agent profiles. Since we designed 7 profiles, this results in 6 policies trained for each profile. As such, for the *RL Unknown* condition, we selected for each RL agent a policy from the corresponding set of profile policies uniformly at random.

Table 3 shows the results for the different conditions. The pure RL agents failed to coordinate with their teammates due to the random policy selection. Notwithstanding, they attained the best mean individual performance since their policies were trained to maximize the reward function associated with their profiles, *irrespectively* of the teammate. We also see that despite not being provided prior information about a teammate, the *ToM* agent is still able to perform better than the unconditioned agent and the RL team (difference not statistically significant, $p \gg 0.05$). Moreover, the *Profile* condition attained the highest mean task performance.

The are a few possible reasons for the lower performance of the ToMCAT agent in this unknown teammate scenario. First, the MADiff models were trained with data produced by RL policies that have a predictable behavior when paired with the corresponding RL teammate—since their policies were co-trained—and were provided with $N_{\text{past}} = 4$ past trajectories. However, here the Character embeddings are not informative enough, especially at the beginning of an episode. This results in the agents not coordinating their behavior, which in turn influences plan generation since we use history in-painting. This analysis is supported by Fig. 4c, where we see that when provided with prior information, a ToMCAT agent is able to correctly identify the teammate's profile ($P \approx 1$) from the start. This contrasts with the *No Prior* condition, where the probability increases as more observations are collected, but on average is no higher than 0.89. We also tested updating the current observation buffer continuously beyond $t = 100$ in a sliding window manner, but observed a decline in ToM performance, presumably because all past trajectories used to train the ToMnet and MADiff models were collected for the first $T_{\text{past}} = 100$ steps of episodes.

Overall, we would need to train the ToMnet and MADiff models from past trajectories collected at different timesteps, and vary the number of past trajectories during training, including providing no prior data, to increase the robustness of ToMCAT agents when paired with unknown teammates. In addition, we believe that training would benefit from data collected by pairing different agents, *e.g.*, randomly pairing RL agents, which would better capture the distribution of possible team behaviors in the task. We acknowledge the current limitations of our framework and leave these developments for future work.

5 CONCLUSIONS & FUTURE WORK

This paper presented ToMCAT, a new framework that combines ToM reasoning with multiagent planning for fast adaptation in complex multiagent tasks. It integrates a ToM network (ToMnet) that learns a strong prior of possible team behaviors and leverages meta-learning to make predictions about underlying characteristics of teammates from minimal data, with a multiagent denoising-diffusion policy (MADiff) approach that generates plans for the agent and its teammates conditioned by ToM embeddings of the team. An online dynamic replanning mechanism monitors plan execution and triggers the sampling of a new plan whenever there is a discrepancy between the plan and the current state of the world. We performed various experiments using different parameterizations of ToMCAT agents under different conditions in a simulated collaborative cooking domain. The results show the usefulness of the replanning mechanism in attaining good pairwise performance without the need to replan at short, fixed-time intervals. It also underscores the importance of reasoning about teammates' characteristics as informed by the ToM embeddings and the agent's own characteristics to allow for fast adaptation to different teammates, especially when no prior information is provided about them.

Future work: We are currently designing a joint model approach, where ToMnet and MADiff systems are combined into a probabilistic generative model that allows sampling multiagent plans conditioned on ToM information, while "forcing" the model to predict desired characteristics of teammate motivations and behaviors. We are also addressing the challenges of making ToMCAT agents more robust to unknown teammates and exploring how the framework can be applied in ad hoc teamwork settings. Finally, we plan to apply ToMCAT to learn from human data to infer profiles, behavior trends of people performing complex joint tasks, and explore its application in adversarial settings requiring nested ToM.

REFERENCES

[1] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B. Tenenbaum, Tommi S. Jaakkola, and Pulkit Agrawal. 2023. Is Conditional Generative Modeling all you need for Decision Making?. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=sP1fo2K9DFG

[2] Simon Baron-Cohen, Alan M. Leslie, and Uta Frith. 1985. Does the autistic child have a "theory of mind"? *Cognition* 21, 1 (1985), 37–46. https://doi.org/10.1016/0010-0277(85)90022-8

[3] Micah Carroll, Rohin Shah, Mark K Ho, Tom Griffiths, Sanjit Seshia, Pieter Abbeel, and Anca Dragan. 2019. On the Utility of Learning about Humans for Human-AI Coordination. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2019/file/f5b1b89d98b7286673128a5fb112cb9a-Paper.pdf

[4] Allan Dafoe, Edward Hughes, Yoram Bachrach, Tantum Collins, Kevin R. McKee, Joel Z. Leibo, Kate Larson, and Thore Graepel. 2020. Open Problems in Cooperative AI. arXiv:2012.08630 [cs.AI] https://arxiv.org/abs/2012.08630

[5] Piotr J Gmytrasiewicz and Prashant Doshi. 2005. A framework for sequential planning in multi-agent settings. *J. Artif. Int. Res.* 24, 1 (jul 2005), 49–79.

[6] Pengjie Gu, Mengchen Zhao, Jianye Hao, and Bo An. 2022. Online Ad Hoc Teamwork under Partial Observability. In *International Conference on Learning Representations*. https://openreview.net/forum?id=18Ys0-PzyPI

[7] Nikolos Gurney, Stacy Marsella, Volkan Ustun, and David V. Pynadath. 2022. Operationalizing Theories of Theory of Mind: A Survey. In *Computational Theory of Mind for Human-Machine Teams (AAAI-FSS 2021)*, Nikolos Gurney and Gita Sukthankar (Eds.). Springer Nature Switzerland, Cham, 3–20. https://doi.org/10.1007/978-3-031-21671-8_1

[8] Yanlin Han and Piotr Gmytrasiewicz. 2018. Learning others' intentional models in multi-agent settings using interactive POMDPs. *Advances in Neural Information Processing Systems* 31 (2018).

[9] Yanlin Han and Piotr Gmytrasiewicz. 2019. Ipomdp-net: A deep neural network for partially observable multi-agent planning using interactive pomdps. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 6062–6069.

[10] John C. Harsanyi. 1967. Games with Incomplete Information Played by "Bayesian" Players, I–III Part I. The Basic Model. *Management Science* 14, 3 (nov 1967), 159–182. https://doi.org/10.1287/mnsc.14.3.159

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 6840–6851. https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf

[12] Jonathan Ho and Tim Salimans. 2022. Classifier-Free Diffusion Guidance. arXiv:2207.12598 [cs.LG] https://arxiv.org/abs/2207.12598

[13] Max Jaderberg, Wojciech M Czarnecki, Iain Dunning, Luke Marris, Guy Lever, Antonio Garcia Castaneda, Charles Beattie, Neil C Rabinowitz, Ari S Morcos, Avraham Ruderman, et al. 2019. Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science* 364, 6443 (2019), 859–865.

[14] Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. 2022. Planning with Diffusion for Flexible Behavior Synthesis. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.). PMLR, 9902–9915. https://proceedings.mlr.press/v162/janner22a.html

[15] Julian Jara-Ettinger. 2019. Theory of mind as inverse reinforcement learning. *Current Opinion in Behavioral Sciences* 29 (2019), 105–110. https://doi.org/10.1016/j.cobeha.2019.04.010 Artificial Intelligence.

[16] Peter Karkus, David Hsu, and Wee Sun Lee. 2017. QMDP-Net: Deep Learning for Planning under Partial Observability. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*. Long Beach, California, USA, 4697–4707. https://doi.org/10.5555/3295222.3295223

[17] Ken Ming Lee, Sriram Ganapathi Subramanian, and Mark Crowley. 2022. Investigation of independent reinforcement learning algorithms in multi-agent environments. *Frontiers in Artificial Intelligence* 5 (2022), 805823.

[18] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 6382–6393.

[19] David V. Pynadath and Stacy C. Marsella. 2005. PsychSim: modeling theory of mind with decision-theoretic agents. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence* (Edinburgh, Scotland) *(IJCAI'05)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1181–1186.

[20] Neil Rabinowitz, Frank Perbet, Francis Song, Chiyuan Zhang, S. M. Ali Eslami, and Matthew Botvinick. 2018. Machine Theory of Mind. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 4218–4227. https://proceedings.mlr.press/v80/rabinowitz18a.html

[21] João G. Ribeiro, Cassandro Martinho, Alberto Sardinha, and Francisco S. Melo. 2023. Making Friends in the Dark: Ad Hoc Teamwork Under Partial Observability. *Frontiers in Artificial Intelligence and Applications* 372 (sep 2023), 1954–1961. https://doi.org/10.3233/FAIA230486

[22] Michele Rocha, Heitor Henrique da Silva, Analúcia Schiaffino Morales, Stefan Sarkadi, and Alison R. Panisson. 2023. Applying Theory of Mind to Multi-agent Systems: A Systematic Review. In *Intelligent Systems (BRACIS 2023)*, Murilo C. Naldi and Reinaldo A. C. Bianchi (Eds.). Springer Nature Switzerland, Cham, 367–381. https://doi.org/10.1007/978-3-031-45368-7_24

[23] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347 [cs.LG] https://arxiv.org/abs/1707.06347

[24] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815* (2017).

[25] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 37)*, Francis Bach and David Blei (Eds.). PMLR, Lille, France, 2256–2265. https://proceedings.mlr.press/v37/sohl-dickstein15.html

[26] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*. https://arxiv.org/abs/2010.02502

[27] Peter Stone, Gal Kaminka, Sarit Kraus, and Jeffrey Rosenschein. 2010. Ad Hoc Autonomous Agent Teams: Collaboration without Pre-Coordination. *Proceedings of the AAAI Conference on Artificial Intelligence* 24, 1 (jul 2010), 1504–1509. https://doi.org/10.1609/aaai.v24i1.7529

[28] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Proceedings of the 12th International Conference on Neural Information Processing Systems (NIPS'99)*. MIT Press, Cambridge, MA, USA, 1057–1063.

[29] Haochen Wu, Pedro Sequeira, and David V Pynadath. 2023. Multiagent Inverse Reinforcement Learning via Theory of Mind Reasoning. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*. 708–716.

[30] Sarah A. Wu, Rose E. Wang, James A. Evans, Joshua B. Tenenbaum, David C. Parkes, and Max Kleiman-Weiner. 2021. Too Many Cooks: Bayesian Inference for Coordinating Multi-Agent Collaboration. *Topics in Cognitive Science* 13, 2 (2021), 414–432. https://doi.org/10.1111/tops.12525 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/tops.12525

[31] Zhengbang Zhu, Minghuan Liu, Liyuan Mao, Bingyi Kang, Minkai Xu, Yong Yu, Stefano Ermon, and Weinan Zhang. 2024. MADiff: Offline Multi-agent Learning with Diffusion Models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. https://openreview.net/forum?id=PvoxbjcRPT