PKU-IAI Technical Report: PKU-IAI-2024-T-0002

Decision Transformer for Modeling Theory of Mind in Multi-Agent Systems

Zhancun Mu, Xizhi Xiao
Yuanpei College
Peking University
{2100017790, 2100017773}@stu.pku.edu.cn

Abstract

The Theory of Mind (ToM) ability in multi-agent systems is crucial for coordinating cooperation and understanding communication. ToM involves the capacity to reason about the mental states of other agents, encompassing their beliefs, desires, intentions, and more. However, in modeling ToM, many existing works rely on assumptions like rationality, which may not hold true in real-world scenarios. To tackle this issue, we leverage the sequence modeling capability of Transformers in the offline setting. In this paper, we (i) introduce the *multi-agent decision transformer* (MADT) for agent modeling and demonstrate its generalization ability with new partners. Additionally, we (ii) propose a framework to enhance online reinforcement learning (RL) policies with ToM modeling using MADT. We evaluate our approach in the Overcooked-AI environment and illustrate its satisfactory generalization ability, even with limited data.[1]

1 Introduction

Understanding and reasoning about the mental states of other agents are fundamental abilities for human interaction. Even without explicit communication or a shared set of common knowledge, we can infer others' intentions, beliefs, and desires from their actions [11]. In multi-agent cooperation scenarios, this capability, often referred to as theory-of-mind (ToM), becomes crucial for task completion when confronted with challenges such as asymmetric information and uncoordinated goals [26, 25].

To model ToM effectively, a key aspect is the modeling of teammates and opponents. One approach involves assuming that agents are rational and utilize a utility function to approximate their behaviors [10, 30]. However, this assumption may not be applicable to diverse agents with different preferences. For instance, in a cooperative cooking game, an agent might prioritize serving a dish over cooking it, leading to incorrect utility function estimation. Another approach is learning from data, as seen in works such as [19, 29, 8, 12], but these methods often lack cooperation and generalization abilities.

Recent advancements in sequence modeling, particularly in natural language processing (NLP) with large models like GPT-4 [15], have demonstrated surprising effectiveness. In reinforcement learning (RL) settings, these techniques open new possibilities for offline RL, aiming to learn from a fixed dataset without direct interaction with the environment. A notable method is the Decision Transformer

[1] See our code at ~~https://github.com/namespacebilibili/ToMDT~~https://github.com/namespacebilibili/ToMDT.

(DT) [5], which exhibits strong performance and generalization across tasks with provided trajectory data.

We posit that the sequence modeling ability of Transformers [27] is suitable for modeling teammates or opponents in multi-agent systems, without relying on assumptions or shared knowledge. In this study, we evaluate the capabilities of the multi-agent decision transformer in the well-known Overcooked-AI environment—a cooperative cooking game where two agents collaborate to serve dishes to customers [3]. This task, requiring coordinated actions and goals, serves as a robust testbed for ToM modeling.

Our contributions are twofold:

- We introduce the Decision Transformer in multi-agent settings and demonstrate its generalization ability with new partners.
- To harness the sequence modeling capability of DT while mitigating its limitations, we propose a framework to enhance online RL policies with ToM modeling using MADT. This framework is compatible with any RL algorithm with value function estimation and can be easily extended to other tasks. This framework highlights a promising direction for combining offline and online RL methods in multi-agent settings.

The subsequent sections will address the following questions:

- Why do we prefer DT over other models for imitation learning?
- What architecture of DT is suitable for modeling ToM?
- How can DT be modified and integrated with online RL policy?

2 Preliminaries and Related Work

Multi-Agent Reinforcement Learning The problem of multi-agent reinforcement learning can be formulated as a Multi-Agent Markov Decision Process (MMDP), represented by the tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, T, n, \gamma \rangle$, where $\mathcal{S}$ is the state space of n agents: $S_1 \times S_2 \times \cdots \times S_n \to \mathcal{S}$. $\mathcal{A}_i$ denotes the action space of agent i, $T : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the transition function, and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \boldsymbol{R}$ is the reward function. The goal of each agent is to maximize the long-term reward $\sum_t \gamma^t r_t$, where γ is the discounting factor, and $r_t^i \in \boldsymbol{R}$ is the reward of agent i at timestep t. Various works have been proposed to address this problem, and a comprehensive formulation and review can be found in [35, 7, 38, 16]. In this work, our focus is primarily on the cooperative setting with implicit communication.

Offline Reinforcement Learning Offline RL is an RL paradigm aiming to learn from static and previously collected data without interacting with the environment [18]. According to the taxonomy in [18], there are mainly three types of offline RL methods: (i) learning a dynamics model, (ii) learning a trajectory distribution, and (iii) directly learning a model-free policy. Our work falls into the second category. Offline MARL remains a relatively unexplored area due to its high complexity. For instance, Yang et al. [34] focus on alleviating the extrapolation error, Pan et al. [17] aim to avoid falling into bad optima, and Wang and Zhan [28] attempt to utilize the underlying decomposable problem structure for offline modeling.

Decision Transformers Transformers have exhibited remarkable capabilities to generalize across a diverse range of tasks, spanning language modeling and text generation to image synthesis and representation learning. Leveraging the Transformer's ability, Decision Transformer (DT) [5] was introduced to model an RL problem as a sequence prediction problem and has outperformed previous offline RL methods in various environments. Trajectories τ are represented in the format:

$$\tau = \{\hat{R}_1, s_1, a_1, \hat{R}_2, s_2, a_2 \cdots, \hat{R}_T, s_T, a_T\},$$

where $\hat{R}_T$ represents the returns-to-go at timestep t, i.e., $\hat{R}_t = \sum_{t'=t}^{T} r_{t'}$. DT predicts a_t in an auto-regressive manner for the current action but does not predict the future state or returns-to-go. Numerous works propose algorithms and architectures building upon DT, such as Prompt DT [31], Online DT [37], Waypoint DT [1], and others. Nevertheless, DT faces challenges, including suboptimal data and a lack of stitching ability. In [32], DT is combined with Q-learning to address this issue. In multi-agent settings, Meng et al. [13] introduced DT to multi-agent settings, proposing

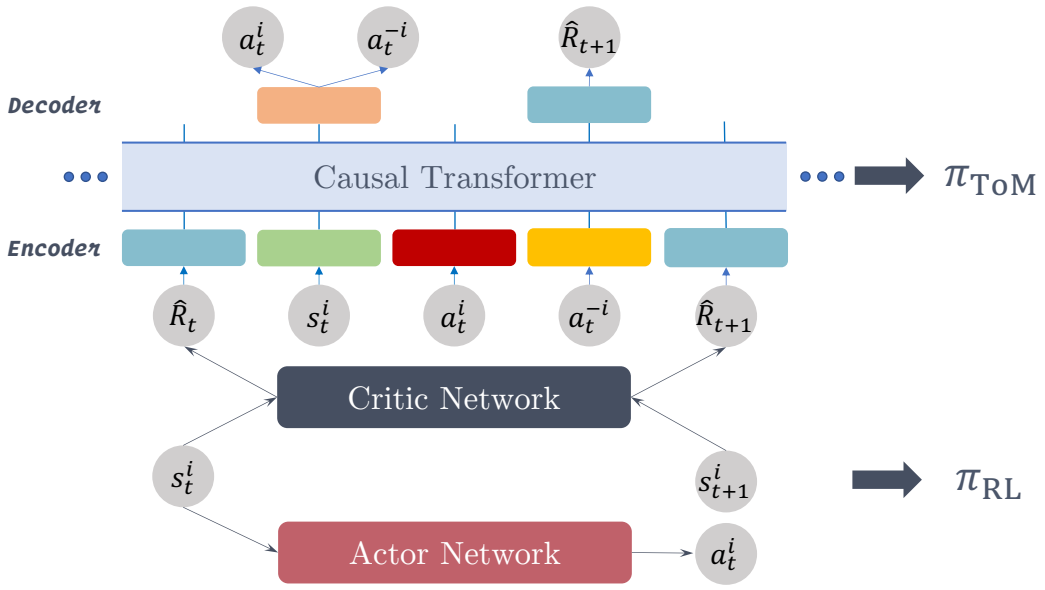

Figure 1: The above model is the architecture of MADT. Returns-to-go, actions of all agents, and states are fed into a GPT-2 architecture as tokens to autoregressively predict actions and returns-to-go. It can be viewed as a ToM policy. To combine with online RL policy, we modify the returns-to-go with the value function estimation and use the RL policy as a prior for the ToM policy.

a novel architecture of multi-agent decision transformer (MADT). Their trajectory formulation is $\tau_i = \{s_1, o_1^i, a_1^i, s_2, o_2^i, a_2^i, \cdots, s_T, o_T^i, a_T^i\}$, where o_t^i is the individual observation of agent i at timestep t. However, they do not explicitly model other agents' behaviors, and the absence of returns-to-go information makes it no different from imitation learning. In our work, we present a different MADT architecture from [13], demonstrating that predicting returns-to-go makes it more suitable to combine with online RL policy.

Theory of Mind Modeling Theory of Mind has been extensively studied in psychology and cognitive science, revealing that even 6-month-old infants can understand others' intentions and beliefs from their actions. Formally, ToM has been modeled as a Bayesian inference problem (Bayesian ToM, BToM) [2], i.e.,

$$P(m|a) \propto P(a|m)P(m), \tag{1}$$

where m denotes the mental state of the other agent, and a refers to its actions. The posterior probability $P(m|a)$ in Eq. (1) is highly relevant to opponent/teammate modeling in multi-agent systems. Two main approaches are employed to model other agents: one leverages the rationality assumption, and the other learns from data or interactions. The first approach assumes that other agents act to maximize their utility function, expressed as:

$$P(a|m) \propto \exp\{\beta U(a, m)\}. \tag{2}$$

Under this assumption, Lim et al. [10], Wu et al. [30] use BToM for subgoal coordination and test their capability with human partners. However, this assumption cannot be universally applied to all types of agents, and the utility function needs to be hand-crafted or learned via RL or other methods. The second approach, as seen in Rabinowitz et al. [19], utilizes meta-learning to predict the agent's behavior using the current trajectory. Unlike our work, it cannot model a *general theory of mind* and lacks interaction ability. Wen et al. [29], He et al. [8], Lowe et al. [12] model other agents in online approaches, which cannot be applied to ad-hoc agents. Our work draws inspiration from [33] but avoids explicitly learning the transition function due to the high dimensionality of the state space and considers long horizons.

3 Multi-Agent Decision Transformer for First-order ToM Modeling

Here, we aim to build a first-order Theory of Mind (ToM) model for multi-agent systems, focusing on modeling other agents and predicting how an action taken by us will affect the behavior of the

other agent. As depicted in Fig. 1, the model predicts the action distribution of other agents at each timestep given the past trajectory. For simplicity, we initially consider a two-agent setting, but the model can be easily extended to more agents.

3.1 MADT Architecture

For the ToM modeling problem, we choose the architecture of the decision transformer as the upper part of Fig. 1, which differs from [13]. Our architecture is based on GPT-2 [20], predicting the next token autoregressively given past tokens. The causal transformer encodes agent i's current trajectory τ_i^t at timestep t and generates output tokens according to our needs. The trajectory formulation satisfies the lowest need of ToM modeling, with $\hat{R}$ conditioning action generation. We predict $\hat{R}$ to guide the ToM policy, which will be discussed in the next section. For state tokens, we deprecate the global shared state in Multi-Agent Reinforcement Learning (MARL) and use the individual observation s^i for simplicity. We feed the action tokens of all agents into the model. Agent i's own action a^i is predicted for a Decision Transformer (DT) policy. Predicting other agents' actions a_{t+1}^{-i} is equivalent to modeling transition probability $T(s_{t+1}^i|\tau_i^t, a_{t+1}^i)$ as in [33]. Formally, the trajectory of agent i is given by:

$$\tau_i^t = \{\hat{R}_1, s_1^i, a_1^i, a_1^{-i}, \hat{R}_2, s_2^i, a_2^i, a_2^{-i}, \cdots, \hat{R}_t, s_t^i, a_t^i, a_t^{-i}\}.$$

To embed these tokens, we use embedding matrices for a^i and a^{-i} and a linear layer for $\hat{R}$ and s^i. For positional encoding, we use the timestep encoding method described in [14] and add a role embedding to distinguish different agents.

3.2 MADT Training

In the training process, the learning objective is to reconstruct trajectories in the offline datasets in a supervised manner. The trajectory is masked to predict the token at timestep t only using the previous history. Mathematically, the autoregressive prediction process is defined as:

$$\hat{R}_{t+1} \sim p_R(\hat{R}_{t+1}|\tau_i^t), a_{t+1}^i \sim p_{a_i}(a_{t+1}^i|\tau_i^t, \hat{R}_{t+1}, s_{t+1}^i), a_{t+1}^{-i} \sim p_{a_{-i}}(a_{t+1}^{-i}|\tau_i^t, \hat{R}_{t+1}, s_{t+1}^i). \quad (3)$$

The prediction targets include a^i, a^{-i}, and $\hat{R}$, and the loss function is defined as:

$$\mathcal{L} = \alpha\mathcal{L}_{a_i} + \beta\mathcal{L}_{a_{-i}} + \eta\mathcal{L}_R. \quad (4)$$

We employ cross-entropy loss for action prediction and mean squared error for returns-to-go prediction. The hyperparameters α, β, η are used to balance the loss of different targets (details in Appendix A.1).

4 Combining MADT with Online RL Policy

~~After learning a decision transformer for Theory of Mind (ToM) modeling, we can use it as a ToM policy with an RL model and further combine them to improve performance. Any~~ Though MADT demonstrates good performance, it still has limitations. As stated in [32], DT lacks the ability to stitch trajectories of suboptimal data, while online RL algorithms do not have the sam In this section, we propose a framework to combine MADT with online RL policy to leverage their advantages and mitigate drawbacks. Any online RL algorithm with value function estimation can be used in our framework.

4.1 MADT as a ToM Policy

Inspired by [33], the ToM policy can be expressed as

$$\pi_{\text{ToM}}(a_t^i|\tau_{t-1}^i, \hat{V}_t, s_t^i) \propto \exp\left\{\frac{1}{\beta}\sum_{a_t^{-i}} p_{a_{-i}}(a_t^{-i}|\tau_i^{t-1}, \hat{V}_t, s_t^i) \sum_{\hat{R}_{t+1}} p_R(\hat{R}_{t+1}|\tau_i^t)[\hat{V}_t - (1-\gamma)\hat{R}^{t+1}]\right\}. \quad (5)$$

To address the issue of suboptimal data, which may lead to negative value estimation in our experiments, we modify the returns-to-go with the value function estimation at each timestep, i.e.,

$\hat{V}_t = \max\{\hat{R}_t, V(s_t^i)\}$, similar to [32]. $V(s_t^i)$ is given by a trained RL policy, and we relabel the returns-to-go each timestep. In the above equation, $p_{a_{-i}}(a_t^{-i}|\tau_i^{t-1}, \hat{V}_t, s_t^i)$ is the first-order ToM modeling term. To avoid the need for obtaining s_i^{t+1} as in [33], we use the predicted return-to-go $\hat{R}_{t+1}$, which estimates the value of the next state, since $V(s_{t+1}) = \mathbb{E}_\pi[\hat{R}_{t+1}]$. Eq. (7) is an estimation of $Q(s_i^t, a_i^t)$, because

$$\begin{aligned} Q(s_i^t, a_i^t) &= r(s_i^t, a_i^t) + \gamma \sum_{s_i^{t+1}} V^\pi(s_i^{t+1}) \\ &\approx \hat{R}_t - \hat{R}_{t+1} + \gamma \hat{R}_{t+1} = \hat{R}_t - (1-\gamma)\hat{R}_{t+1}. \end{aligned} \tag{6}$$

β controls the temperature of the Boltzmann distribution, meaning π_{ToM} becomes deterministic as $\beta \to 0$ and random as $\beta \to \infty$. To calculate the ToM policy, methods like Monte Carlo sampling can be used, but for our experiments, we use the greedy policy for a_t^{-i} and only sample $\hat{R}_{t+1}$ once. For the choice of γ and β, we do ablation tests and report the best hyperparameters in Appendix A.1.

4.2 Using RL Policy as a Prior

Online RL algorithms like self-play can hardly coordinate with new partners [3], while decision transformers seldom outperform the best offline data. To leverage their advantages and mitigate drawbacks, we combine them as follows:

$$\pi(a_t^i|\tau_{t-1}^i, \hat{V}_t, s_t^i) \propto \pi_{\text{RL}}(a_t^i|\tau_{t-1}^i, s_t^i) \cdot \pi_{\text{ToM}}(a_t^i|\tau_{t-1}^i, \hat{V}_t, s_t^i). \tag{7}$$

~~The first term in the equation above is the online RL policy, and the second term is the ToM policy. According to [33], the RL policy serves as a prior $P(a_t^i|\tau_{t-1}^i, s_t^i)$, and the ToM policy modifies the posterior in the RL policy $P(a_t^i|\text{best return}, \tau_{t-1}^i, s_t^i)$ by the ToM estimation~~The reason why we use the product of the two policies can be seen in [33] similarly. Here π_{RL} can be considered a prior $\Pr(a_t^i|\tau_{t-1}^i, s_t^i)$ and π_{ToM} is equivalent to $\Pr(\text{best reward}|a_t^i, \tau_{t-1}^i, s_t^i)$ by definition. Combining them together we get the posterior probability:

$$\Pr(a_t^i|\text{best reward}, \tau_{t-1}^i, s_t^i) \propto \Pr(a_t^i|\tau_{t-1}^i, s_t^i) \cdot \Pr(\text{best reward}|a_t^i, \tau_{t-1}^i, s_t^i), \tag{8}$$

which means the action distribution to get best reward in the long run. It is worth noting that the intuition here is similar to the usage of return-to-go in DT, which is to guide the policy to the optimal trajectory.

5 Experiments

In this section, we perform experiments to evaluate the performance of MADT and our framework. First, we test the teammate modeling ability of MADT comparing with the imitation learning model in [3]. Second, we test the performance of imitation learning, MADT, RL algorithm and the combination framework paired with a human proxy model.

5.1 Environment and Dataset

We set up our experiments in the Overcooked-AI environment [3]. It is a cooperative cooking game that requires two agents to work together to serve dishes to customers. The environment has been considered a good testbed for multi-agent cooperation and ad-hoc teammate modeling [30, 4, 9, 6, 21]. It contains 5 layouts and the details can be found in Appendix A.1.

We adopt the dataset provided by [3] which contains 126 trajectories as training data and 45 trajectories as test data for 5 minigames. It should be noticed that the data amount is very small in comparison to the datasets used in [5, 13, 14] and the trajectories are not optimal.[2]

5.2 Testing Teammate Modeling Ability of MADT

Though some works argue that transformer is not crucial for sequence modeling task [24], we argue that the capacity of transformer is important for generalization ability to new partners. We choose

[2]This can be seen by the dataset-building code in our codebase.

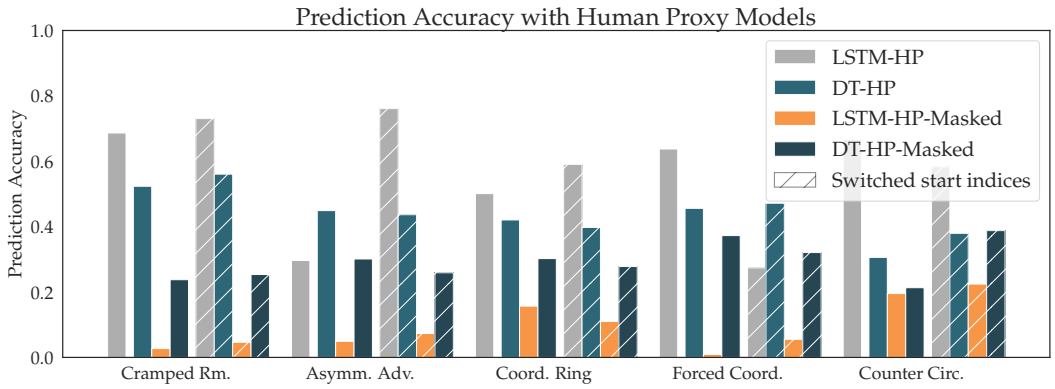

Figure 2: Prediction accuracies of the human proxy model (HP)'s behavior paired with MADT and BC model for 5 seeds.The hashed bars show the results when the roles of the two agents are swapped.Due to the poor performance of the BC model, whose prediction outputs are always STAY, we show the masked results which mask the STAY action.

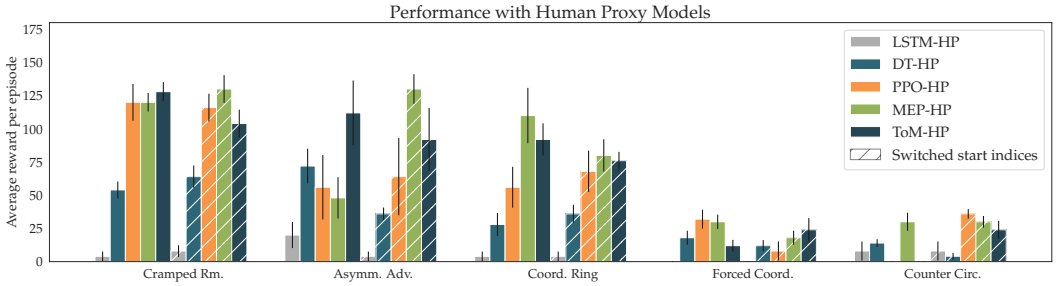

Figure 3: Average rewards of agents paired with the human proxy model, with standard error over 5 runs.

the behavior cloning model (BC model) in [3] but train it on the whole dataset. It should be noted that the MADT model is trained on the entire dataset. According to [3], a human proxy model can be obtained by imitation learning on the testing human data. The prediction accuracy of the human proxy model's behavior can be used to evaluate the ToM modeling ability of the model. For each minigame, we separately pair the human proxy model with the MADT and BC model, and report the prediction accuracy in Fig. 2. Even though the BC model's prediction accuracy is higher, we find this is due to the fact that the BC model acts poorly paired with the human proxy model, thus always predicting the action STAY, which can be seen in Fig. 3. We mask the STAY action and calculate the remaining accuracy.

The results show that for unmasked actions, MADT has a higher prediction accuracy than the BC model when paired with new partners, demonstrating its capability for sequence modeling. Though in some layouts the prediction accuracy of MADT is relatively low, we argue this is owned to the fact that the actions are equivalent in most cases.

5.3 Testing the Combination Framework

To test the performance of our framework, we pair it again with the human proxy model and compare it with the BC model, MADT, ~~and RL algorithm. We choose the PPO algorithm [23]~~ PPO [23] and MEP [36]. MEP is a population-based training method to avoid distributional shift when paired with unencountered partners. The PPO algorithm is trained in self-play as the RL algorithm according to [3]. Following the settings in [3], we count the cumulative rewards over 400 timesteps in all the 5 layouts.

The results can be seen in Fig. 3. ~~We find the results of~~ In CRAMPED ROOM, ASYMMETRIC ADVANTAGES, and COORDINATION RINGS, our framework (ToM+HP) ~~perform significantly better than single PPO on some of the environments like asymmetric advantages and coordination rings, indicating that the ToM policy could augment the PPO algorithm~~outperforms the other baselines except for MEP. For the first order of agent pairs, we find that the performance of our method

outperforms all the baselines on CRAMPED ROOM and ASYMMETRIC ADVANTAGES. And in ~~the asymmetric advantages~~ particular, we find the results of our framework perform significantly better than the baselines on ASYMMETRIC ADVANTAGES which is a challenging layout for agents in asymmetric roles. In this layout, MADT outperforms the PPO algorithm, which may help explain the significant improvement of our framework. ~~Our framework fails~~ However, our framework performs badly on the counter circuit layout ~~,~~ and even the DT model performs better than our framework. This is potentially due to the ~~limitation on data amount~~poor performance of self-play PPO on this layout, which can be seen in Fig. 3.

Overall, our framework demonstrates quite satisfying results with ad-hoc teammates compared to the baselines.

6 Discussion and Conclusion

In this work, we introduce multi-agent decision transformer (MADT) for first-order ToM modeling and propose a framework to combine it with an online RL policy. In the Overcooked-AI environment, we show that MADT has better opponent modeling ability than the LSTM-based BC model. We also show that our framework can improve the performance of the online RL policy and MADT.

Due to the limited time and computational resources, we only test our method in the Overcooked-AI environment paired with the human proxy model. While these initial results are quite promising, further evaluation is needed to fully demonstrate the capabilities of our proposed approach.

For future work, we plan to test our method in other complex multi-agent environments such as StarCraft [22], which provides a large dataset for training and evaluation. Testing in additional environments will allow us to better analyze the generalization ability of MADT across different tasks and teammates.

Furthermore, experiments with a wider variety of agent types beyond the human proxy model would be valuable, including both artificial agents with different capabilities and behaviors as well as studies with real human teammates. Human experiments are especially important to truly validate the benefits of MADT's teammate modeling for human-AI cooperation.

In addition, we currently only demonstrate first-order ToM modeling in MADT using implicit mental state representation~~and communication~~. Incorporating more advanced hierarchical ToM frameworks that model higher-order recursive reasoning is a promising direction for enabling deeper multi-agent coordination. There remain many exciting opportunities to build upon this work to create more flexible, generalizable multi-agent learning techniques. We look forward to pursuing these extensions in future research.

References

[1] Anirudhan Badrinath, Yannis Flet-Berliac, Allen Nie, and Emma Brunskill. Waypoint transformer: Reinforcement learning via supervised learning with intermediate targets. *Advances in Neural Information Processing Systems*, 2023. 2

[2] 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. 3

[3] Micah Carroll, Rohin Shah, Mark K Ho, Tom Griffiths, Sanjit Seshia, Pieter Abbeel, and Anca Dragan. On the utility of learning about humans for human-ai coordination. *Advances in neural information processing systems*, 32, 2019. 2, 5, 6, 10

[4] Rujikorn Charakorn, Poramate Manoonpong, and Nat Dilokthanakul. Investigating partner diversification methods in cooperative multi-agent deep reinforcement learning. In *Neural Information Processing: 27th International Conference, ICONIP 2020, Bangkok, Thailand, November 18–22, 2020, Proceedings, Part V 27*, pages 395–402. Springer, 2020. 5

[5] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021. 2, 5

[6] Matthew C Fontaine, Ya-Chuan Hsu, Yulun Zhang, Bryon Tjanaka, and Stefanos Nikolaidis. On the importance of environments in human-robot coordination. *arXiv preprint arXiv:2106.10853*, 2021. 5

[7] Sven Gronauer and Klaus Diepold. Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review*, pages 1–49, 2022. 2

[8] 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. 1, 3

[9] Paul Knott, Micah Carroll, Sam Devlin, Kamil Ciosek, Katja Hofmann, Anca D Dragan, and Rohin Shah. Evaluating the robustness of collaborative agents. *arXiv preprint arXiv:2101.05507*, 2021. 5

[10] Terence X Lim, Sidney Tio, and Desmond C Ong. Improving multi-agent cooperation using theory of mind. *arXiv preprint arXiv:2007.15703*, 2020. 1, 3

[11] Shari Liu, Tomer D Ullman, Joshua B Tenenbaum, and Elizabeth S Spelke. Ten-month-old infants infer the value of goals from the costs of actions. *Science*, 358(6366):1038–1041, 2017. 1

[12] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017. 1, 3

[13] Linghui Meng, Muning Wen, Chenyang Le, Xiyun Li, Dengpeng Xing, Weinan Zhang, Ying Wen, Haifeng Zhang, Jun Wang, Yaodong Yang, et al. Offline pre-trained multi-agent decision transformer. *Machine Intelligence Research*, 20(2):233–248, 2023. 2, 3, 4, 5

[14] Lina Mezghani, Piotr Bojanowski, Karteek Alahari, and Sainbayar Sukhbaatar. Think before you act: Unified policy for interleaving language reasoning with actions. *arXiv preprint arXiv:2304.11063*, 2023. 4, 5

[15] OpenAI. Gpt-4 technical report, 2023. 1

[16] Afshin Oroojlooy and Davood Hajinezhad. A review of cooperative multi-agent deep reinforcement learning. *Applied Intelligence*, 53(11):13677–13722, 2023. 2

[17] Ling Pan, Longbo Huang, Tengyu Ma, and Huazhe Xu. Plan better amid conservatism: Offline multi-agent reinforcement learning with actor rectification. In *International Conference on Machine Learning*, pages 17221–17237. PMLR, 2022. 2

[18] Rafael Figueiredo Prudencio, Marcos ROA Maximo, and Esther Luna Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *IEEE Transactions on Neural Networks and Learning Systems*, 2023. 2

[19] 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. 1, 3

[20] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. 4, 10

[21] João G Ribeiro, Cassandro Martinho, Alberto Sardinha, and Francisco S Melo. Assisting unknown teammates in unknown tasks: Ad hoc teamwork under partial observability. *arXiv preprint arXiv:2201.03538*, 2022. 5

[22] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder De Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043*, 2019. 7

[23] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. 6

[24] Max Siebenborn, Boris Belousov, Junning Huang, and Jan Peters. How crucial is transformer in decision transformer? *arXiv preprint arXiv:2211.14655*, 2022. 5

[25] Stephanie Stacy, Siyi Gong, Aishni Parab, Minglu Zhao, Kaiwen Jiang, and Tao Gao. A bayesian theory of mind approach to modeling cooperation and communication. *Wiley Interdisciplinary Reviews: Computational Statistics*, page e1631. 1

[26] Michael Tomasello, Malinda Carpenter, Josep Call, Tanya Behne, and Henrike Moll. Understanding and sharing intentions: The origins of cultural cognition. *Behavioral and brain sciences*, 28(5):675–691, 2005. 1

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2

[28] Xiangsen Wang and Xianyuan Zhan. Offline multi-agent reinforcement learning with coupled value factorization. *arXiv preprint arXiv:2306.08900*, 2023. 2

[29] Ying Wen, Yaodong Yang, Rui Luo, Jun Wang, and Wei Pan. Probabilistic recursive reasoning for multi-agent reinforcement learning. *International Conference on Learning Representations*, 2019. 1, 3

[30] Sarah A. Wu, Rose E. Wang, James A. Evans, Joshua B. Tenenbaum, David C. Parkes, and Max Kleiman-Weiner. Too many cooks: Coordinating multi-agent collaboration through inverse planning. *Topics in Cognitive Science*, 2021. 1, 3, 5

[31] Mengdi Xu, Yikang Shen, Shun Zhang, Yuchen Lu, Ding Zhao, Joshua Tenenbaum, and Chuang Gan. Prompting decision transformer for few-shot policy generalization. In *international conference on machine learning*, pages 24631–24645. PMLR, 2022. 2

[32] Taku Yamagata, Ahmed Khalil, and Raul Santos-Rodriguez. Q-learning decision transformer: Leveraging dynamic programming for conditional sequence modelling in offline rl. In *International Conference on Machine Learning*, pages 38989–39007. PMLR, 2023. 2, 4, 5

[33] Runzhe Yang, Jingxiao Chen, and Karthik Narasimhan. Improving dialog systems for negotiation with personality modeling. *arXiv preprint arXiv:2010.09954*, 2020. 3, 4, 5

[34] Yiqin Yang, Xiaoteng Ma, Chenghao Li, Zewu Zheng, Qiyuan Zhang, Gao Huang, Jun Yang, and Qianchuan Zhao. Believe what you see: Implicit constraint approach for offline multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34:10299–10312, 2021. 2

[35] 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. 2

[36] Rui Zhao, Jinming Song, Yufeng Yuan, Haifeng Hu, Yang Gao, Yi Wu, Zhongqian Sun, and Wei Yang. Maximum entropy population-based training for zero-shot human-ai coordination. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 6145–6153, 2023. 6

[37] Qinqing Zheng, Amy Zhang, and Aditya Grover. Online decision transformer. In *international conference on machine learning*, pages 27042–27059. PMLR, 2022. 2

[38] Changxi Zhu, Mehdi Dastani, and Shihan Wang. A survey of multi-agent reinforcement learning with communication. *arXiv preprint arXiv:2203.08975*, 2022. 2

A Appendix

A.1 Implementation Details

The Overcooked-AI environment contains 5 minigames with the following names: cramped room, asymmetric advantages, coordination rings, forced coordination, and counter circuit. The action space for each agent consists of: UP, DOWN, LEFT, RIGHT, STAY, INTERACTION. For the state space, we use the self-centered representation provided by the environment, which is a 96-dimensional vector. The high dimensionality of the state space makes modeling the transition function challenging.

For the multi-agent decision transformer, we utilize the GPT-2 architecture [20]. The action tokens are encoded by embedding matrices, while the returns-to-go and state tokens are encoded by linear layers. The prediction heads are linear layers. The detailed model hyperparameters are provided in Table 1. For the training process, we use a weighted cross-entropy loss for action prediction to account for the imbalance in the action distribution. The weights are calculated by counting the number of each action a_i in the training data, with the weight of each action being $\frac{\#(a_i)}{\max \#(a_i)}$. The training hyperparameters are shown in Table 2. The definitions of α, β, η can be found in Equation 4.

We use the same settings as [3] for the human proxy model and RL algorithm. For our combination framework, we choose the discounting factor $\gamma = 0.9$ and temperature values $\beta = [2, 2, 2, 1, 1]$ for the corresponding minigames after performing ablation experiments.

Table 1: Model hyperparameters of MADT.

Hyperparameter	Value
Embedding size	128
Number of layers of GPT-2	3
Number of heads of GPT-2	1
Max Episode Length	1250
Context Length	10
Residual Dropout	0.1
Attention Dropout	0.1

s

Table 2: Training hyperparameters of MADT.

Hyperparameter	Value
Batch Size	256
Learning Rate	$5e-4$
Weight Decay	0.01
Number of Epochs	20
α, β, η	$[30, 30, 1]$

A.2 Contributions

All authors contributed significantly throughout all stages of this research, including active discussion and collaboration. For the foundational work, Zhancun and Xizhi jointly led the literature review, topic selection, method proposal, and experiment design. Zhancun focused on implementing the codebase and conducting the experiments, while Xizhi proposed ideas on refining the method and experiments. For the paper writing, Zhancun wrote the initial draft, then Xizhi thoroughly polished the full paper. The authors worked closely together in an integrated way on all aspects of the research, each leveraging their distinct perspectives and strengths to push this challenging problem forward.