Local Information Based Attentional Opponent Modelling In Multi-agent Reinforcement Learning

Binqiang Chen

Beihang University, Beijing, China
chenbq@buaa.edu.cn

Abstract

Opponent modelling (OM) can enable agents to reason behaviors of others, in order to act accordingly and interact effectively in multi-agent settings. Most of the existing OM approaches perform explicit models based on opponent's observations and actions during execution, where incessant message exchange is required to predict others' real-time behavior to aid local decision. However, reliable communication is not available in many practical cases. In this paper, we propose attentional opponent modelling (ATOM) as an alternative approach, where the policy network depends on local observation without communication requirements during execution. The key consideration is that local observation usually contains knowledge of adjacent entities. By interaction-based attention mechanism, the opponent's belief for each adjacent entity can be estimated. By aggregating such predicted beliefs, each agent can choose its action to promote cooperation. Experiments show that ATOM achieves superior performance over typical OM methods without extra messages exchange. Surprisingly, ATOM gets similar performance to approaches augmented with communication channels during execution.

Introduction

Opponent modelling, also known as agents modelling agents, enables agents predict about various properties of interest of other agents and incorporate them into decision making to enhance the interaction efficiency between agents, which has a rich history in multi-agent learning (Shoham, Powers, and Grenager 2007; Albrecht and Stone 2018).

By equipping with state-of-art machine learning methods, OM approaches focus on constructing powerful models to predict the opponents' properties, such as the strategy type (He et al. 2016), task goal (Raileanu et al. 2018), and next action (Rabinowitz et al. 2018). Such models are often based on the availability of opponents' local trajectory, including observations, historical actions, and possibly received rewards, which demands incessant message exchange among all agents during execution phase. As a result, they are inapplicable to multi-agent systems without unavailable or reliable communication channels. To resolve such limitation, local information based opponent modelling serves as an alternative and promising way, where an agent reason others' properties with locally available information. Recent works applied such idea to control single agent by predict the opponents' strategy type, where other agents are with given strategy (Papoudakis, Christianos, and Albrecht 2021; Xie et al. 2020). However, due to the complexity introduced by dynamic interaction among simultaneously learning agents, the idea has not been adopt to multi agent reinforcement learning (MARL) settings.

In this paper, we are interested in fully exploiting the local observation to implicitly model the interaction among agents to learn sophisticated policy in MARL, which is inspired by the following findings. On one hand, each agent can usually obtain some knowledge of adjacent agents from local observation, which is largely overlooked and not well exploited in existing literatures. For example, in games, each player can see the position, health status, skills list, speed of other player; in car driving, each driver can see the position and the similar traffic status of others. On the other hand, the information and interaction between adjacent agents dominate the performance of multiagent system in some cases (Jiang and Lu 2018), thus we focus on modelling other agents in the visibility range.

We propose a novel approach named attentional opponent modelling (ATOM), which takes three steps to reason and incorporate opponents' beliefs. Firstly, the encoding layer is utilized to partition local observations into sub-features based on prior structure information and encode them into latent feature vectors, where each vector can reflect some characteristic of corresponding agent in proximity. Secondly, the interaction-based attention layer uses an interaction unit takes input as feature vectors combination to model latent interactions among agents and applies an attention mechanism to reason opponent belief on the importance weights of the encoded features. Thirdly, by considering both self belief and the inferred beliefs of other entities, the importance weights are aggregated into the final weights distribution over the encoded features to obtain the observation abstraction. With the help of high-level abstraction, agents are expected to act accordingly for coordination based on raw observation. It is also notable that ATOM operates the actor-critic framework, which does not explicitly train the opponent models with extra supervised signal. We conduct sufficient experiments to validate the performance of the proposed ATOM approach. Results show the superiority compared to typical opponen-

t modelling methods without communication channel. To our surprise, the results for ATOM can also surpass some methods allowing communication during the execution.

Related Works

Multi-Agent Reinforcement Learning

The exponential growth of the dimensions in action space inherent in MARL makes learning a centralized policy to simultaneously control all agents prohibitively expensive, especially when the number of agents increases (Hernandez-Leal, Kartal, and Taylor 2019; Zhang, Yang, and Başar 2019). Consequently, many prior works have followed a Centralized Training Decentralized Execution (CTDE) paradigm, where policies of agents can be executed independently but are trained in a decentralized way. This usually takes the form of agents that share a joint value function. For instance, MADDPG (Lowe et al. 2017) is proposed as an actor-critic model for the settings of mixed tasks. Value Decomposition Networks (VDN) (Sunehag et al. 2018) and QMIX (Rashid et al. 2018) are value-based models by decomposing the joint value function as the function of independent value function for each agent.

Opponent Modelling

Opponent modelling, which enables agents to reason and understanding the other agents by constructing models to predict about various properties of interest (Albrecht and Stone 2018), has a long history in artificial intelligence (Brown 1951). In this work, we are interested in opponent modelling methods in the field of multi-agent deep reinforcement learning.

Most of existing work falls into the category of explicit modelling, where an independent model for other agent is constructed for certain property prediction with luxury supervised signal. Theory of Mind Network (ToMnet) (Rabinowitz et al. 2018) propose character, mental state and prediction networks to predict the opponent's next action. The Interactive POMDP (I-POMDP) (Gmytrasiewicz and Doshi 2005) extends the partially observed MDP (Sondik 1971) by introducing an extra space of models of other agents into the MDP. LOLA (Foerster et al. 2018b) predicts opponent's policy parameters based on trajectories of other agents using maximum likelihood. PR2 (Wen et al. 2019) approximates opponents' conditional policies by variational Bayes methods. However, LOLA and PR2 only use the learned opponent model during training to coordinate all agents, which limits the emergent performance. Self Other modelling (SOM) (Raileanu et al. 2018) uses the agent's own policy as a means to predict the opponent's goal. However, it is only applicable for tasks with explicit goals. Encoder-decoder modes are trained for the learning agent to replicate opponent behaviors with only local observation (Papoudakis, Christianos, and Albrecht 2021; Xie et al. 2020), where other agents performs given policies.

Implicit modelling does not produce explicit models of other agents, but implicitly infers properties of other agents, such as their beliefs and intentions. A policy representation learning framework (Grover et al. 2018) is proposed for policy generalization, by balancing the capability of behavior imitation and type recognition. However, the representation is unexplainable and the performance gain in multi-agent settings is marginal compared to standard MADDPG. DRON (He et al. 2016) learns a representation of the opponent's policy given fixed joint policy of opponents. While our work consider the learning of all agents' policies. Another related way to understand the behavior of the other interacting agents is to directly share intention or motive. It can be seen as a kind of implicit opponent modelling augmented with communication to some extent. For example, CommNet (Sukhbaatar, Szlam, and Fergus 2016) and BicNet (Peng et al. 2017) focus on the centralized message processing; ACML (Mao et al. 2020) and ATOC (Jiang and Lu 2018) aim to prune messages among agents to filter the useless information. Unlike those methods, ATOM does not require communication channel during execution.

Technical Background

Dec-POMDP

In this work, we consider the partially observable distributed environment modeled as DEC-POMDP (Bernstein et al. 2002) and defined as the set $\langle N, \mathcal{S}, \{\mathcal{A}_{i\in\{1,...,N\}}\}, P, r, \{\mathcal{O}_{i\in\{1,...,N\}}\}, Z\rangle$. N is the number of agents. $\mathcal{S}$ is the set of environmental states shared by all agents. $\mathcal{A}_i$ represents the set of local actions of agent i. $P : S \times A \times S \to [0, 1]$ represents the state transition function. $R : S \times A \times S \to \mathbb{R}$ is the reward function shared by all agents. $\mathcal{O}_i$ is the set of local observation of agent i. $Z_i : S \times A_i \to O_i$ is the observation function for agent i. $\gamma \in [0, 1]$ is the discount factor. In a given state s, agent i produces an action a_i using its own policy $\pi_i(a_i|o_i)$ based on its local observation o_i. The environment transforms to a new state s' with probability $P(s'|s, a_1, ..., a_n)$ given the current state s and joint actions $a = \{a_1, ..., a_N\}$. Each agent receives its new observation $o'_i = Z_i(s')$ and a common reward $r = R(s, a_1, ..., a_N)$. All agents aim to maximize the total expected return $\mathbb{E}[\sum_{t=1}^{T} \gamma^t r_t]$, where γ is a discount factor and T is the total time step for each episode.

Multi-Agent Deep Deterministic Policy Gradient (MADDPG)

MADDPG (Lowe et al. 2017) is proposed by following the framework of centralized training with decentralized execution, which is an extension of the typical actor-critic methods, like DDPG (Lillicrap et al. 2015), by augmenting the critic for each agent with extra information from other agents. The critic estimates the joint action-value $Q(o_1, ..., o_N, a_1, ..., a_N)$ conditioned on all observations and actions, and the actor $\mu(o_i; \theta_i)$ is only conditioned on local observation o_i and updates its parameters θ_i in the direction suggested by the critic. The critic is important to guide the correct directions to train the policy, and commonly estimated by temporal-difference learning (Sutton, Barto et al. 1998). During training, the centralized critic is utlized to guide the optimization of actors. When testing, the critic is detached and the agent executes with its policy in a decentralized way.

Attention Mechanism

Attention mechanism is widely used in many AI fields, e.g., natural language processing (Bahdanau, Cho, and Bengio 2015) and computer vision (Wang et al. 2018). Self-attention, also known as intra-attention, is an attention mechanism relating different positions of a single sequence in order to compute a representation of the same sequence, where scaled dot-product attention is typically adopted (Vaswani et al. 2017). Graph-attention (Veličković et al. 2018) proposed to use shared transformation network to compute the attention coefficients rather than dot-product, which aims to improve the representation capacity, and in this paper we consider this.

Methods

In this section, we introduce the attentional opponent modelling (ATOM) approach that aims to learn sophisticated policy by discriminating the importance coefficients of latent feature vectors that contain knowledge of entities adjacent to given agent. The importance coefficient is obtained by implicitly capturing interaction between pair-wise agents, and the latent feature vectors are based on the partition and encoding of local observation without requirement for message exchanges among agents.

Recall that we consider N agents performing a cooperative task, with their policies $\{\mu_1, , ..., \mu_N\}$ parameterized by $\{\theta_1^\mu, ..., \theta_N^\mu\}$, respectively. At the beginning of each timestep t, agent i for $\forall\ i \in \{1, ..., N\}$ gets a local observation $\boldsymbol{o}_{i,t}$, and chooses a continuous action $\boldsymbol{a}_{i,t} = \mu(\boldsymbol{o}_{i,t}; \theta_i^\mu)$ in order to maximize long-term global reward $\mathbb{E}[\sum_{t=1}^{T} \gamma^t r_t]$. We consider fully cooperative multi-agent tasks that are modeled as Dec-POMDPs, where all agents are with the purpose of maximizing cumulative team reward.

Following prior work (Foerster et al. 2018a; Lowe et al. 2017), we operate under the paradigm of centralized learning and decentralized execution. During training, the centralized critic is allowed to collect observations and actions of all agents and guides the optimization of individual agent policies. At execution or test time, the critic is no longer used and the agent performs in decentralized way. Since no agent has complete knowledge of the environment state and there is no communication channel during execution, each agent is incentive in fully exploiting the local observation to choose action for high cumulative reward.

The key consideration in this paper is that the partial observation can include the knowledge of itself, adjacent agents and the environment, which offers opportunity to model and reason the others' property. For instance, the autonomous driving car has the knowledge of the speed and position of adjacent cars and pedestrians, and also the status of environment, such as the road friction change due to the raining or snowing. The following carefully designed model provides a feasible and efficient implement for such idea.

As illustrated in Figure 1, ATOM consists of N actor networks and a centralized critic network. The centralized critic network approximates the joint action-value function, which is implemented by typical MLP network as in (Lowe et al. 2017). Each actor network contains four layers: *encoding* layer, *interaction-based attention* layer, *aggregation-based pooling* layer and *actor* layer. In the following, we take agent i as example to demonstrate the policy network, and the subscript of agent index i and time step t are omitted in the following notations for simplicity.

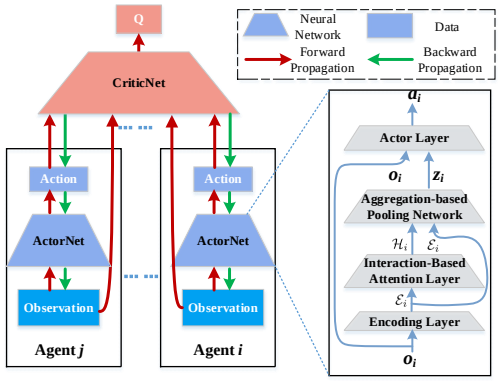

Figure 1: Architecture of ATOM.

The encoding layer takes input local observation $\boldsymbol{o}$ and outputs a set of latent feature vectors $\mathcal{E} = \{\boldsymbol{e}_0, \boldsymbol{e}_1, ..., \boldsymbol{e}_M\}$, where M is the number of adjacent entities [1]. Considering that the structure of local observation feature is known in prior, the encoding layer first splits the observation vector $\boldsymbol{o}$ into M non-overlapping sub-vectors set $\{\boldsymbol{x}_0, \boldsymbol{x}_1, ..., \boldsymbol{x}_M\}$ based on vector-partitioning (Mitchell, Tosun, and Sheppard 2015), where zero padding is utilized for ensure the same length of each vector. Then, $\boldsymbol{x}_k$ is encoded into $\boldsymbol{e}_k = \alpha_{enc}(\boldsymbol{x}_k)$ with a shared encoding function α_{enc}, where $0 \leq k \leq M$. Consequently, $\boldsymbol{e}_k$ represents some latent feature of kth entity adjacent to the given agent (i.e., agent i). Each latent feature vector can reflect the high-level property of the given agent, adjacent agent or the other entities in the proximity, and is denoted as *"feature node"* in the sequel.

The interaction-based attention layer takes as input $\mathcal{E}$, and output an attention weights distributions set $\mathcal{H} = \{\boldsymbol{h}_0, \boldsymbol{h}_1, ..., \boldsymbol{h}_M\}$. $\boldsymbol{h}_k$ is an attention weights distribution considering the interaction between the kth entity and other entities. Each elements in $\boldsymbol{h}_k$ indicates the normalized importance of each feature node in $\mathcal{E}$ from the perspective of the kth entity. In the aggregation-based pooling layer, $\mathcal{H}$ is aggregated to obtain an aggregated attention weights distribution $\boldsymbol{h}$, which represents the overall normalized importance coefficient of each feature node in $\mathcal{E}$ and is used for the computation of observation abstraction vector $\boldsymbol{z}$. Finally, both $\boldsymbol{o}$ and $\boldsymbol{z}$ are fed into the actor layer to produce a continuous action $\boldsymbol{a}$. For brevity purpose, we omit the description for the detailed actor layer, which simply adopts MLP network and can be trivially incorporated. In the following, we detail the interaction-based attention layer and the aggregation-based pooling layer, which are the key components of ATOM.

[1] The entities includes intelligent agents and other static or moving objects in the visibility range of agent i.

Interaction-based attention Layer

An intelligent agent operating in cooperative tasks must have the ability to reason about intentions of others, which largely depends on the awareness of interaction among agents. In practice, despite of unreachable global environment state, local observation usually contains plenty of knowledge for adjacent entities in the environment. It is thus vital important to explicitly consider high-order features interaction in $\mathcal{E}$ to fully extract the interplay among entities rather than simply regarding the features in observation as independent variables.

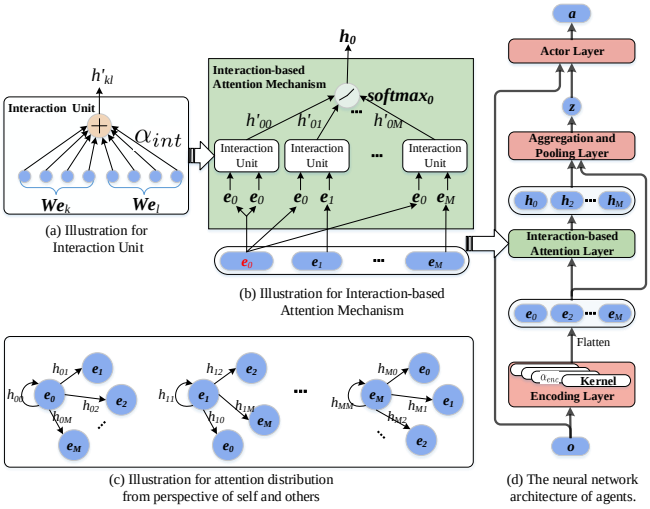

Figure 2: The neural network architecture of the agent network.

In order to model feature interaction in $\mathcal{E}$, the interaction-based attention layer resorts to second-order combinations based on interaction-unit as shown in Figure 2(a). A shared linear transformation, parameterized by a weight matrix $\boldsymbol{W}$, is utilized for each node to obtain sufficient expressive power for higher-level representation. Then, the pair-wise transformed node combinations $\boldsymbol{W}\boldsymbol{e}_k$ and $\boldsymbol{W}\boldsymbol{e}_l$ $(k, l \in \{1, 2, ..., M\})$ are processed by a shared interaction function to compute the importance of $\boldsymbol{e}_l$ from the perspective of the kth entity as $\alpha_{int}(\boldsymbol{W}\boldsymbol{e}_k||\boldsymbol{W}\boldsymbol{e}_l)$, which is directly implemented by MLP. As is in Figure 2(a), the computation for the importance coefficient h'_{0M} of $\boldsymbol{e}_M$ over $\boldsymbol{e}_0$ is illustrated. The shared weight matrix $\boldsymbol{W}$ transforms all feature nodes into the same latent space, then the interaction function α_{int} evaluates relationship and implicit interaction between each pair-wise combination.

To compute the attention weights, i.e., the normalized importance coefficients, for different feature nodes, we normalize them across all choices of l for given k using the softmax function:

$$\begin{aligned} h_{k,l} &= \mathrm{softmax}_k(\alpha_{int}(\boldsymbol{W}\boldsymbol{e}_k||\boldsymbol{W}\boldsymbol{e}_l)) \\ &= \frac{\exp(\alpha_{int}(\boldsymbol{W}\boldsymbol{e}_k||\boldsymbol{W}\boldsymbol{e}_{i,l}))}{\sum_{l'}\exp(\alpha_{int}(\boldsymbol{W}\boldsymbol{e}_k||\boldsymbol{W}\boldsymbol{e}_{l'}))}. \end{aligned} \tag{1}$$

It is notable that $h_{k,l}$ indicates the relative importance of feature node l from the perspective of entity k. We call the attention weights vector $\boldsymbol{h}_k = [h_{k,0}, h_{k,1}, ..., h_{k,M}]$ as attention weights distribution of feature node k. As the illustration for computation of $\boldsymbol{h}_0$ in Figure 2(b), $\boldsymbol{h}_0 = [h_{0,0}, h_{0,1}, ..., h_{0,M}]$ represents the importance distribution and reflects the impact of each feature node from the perspective of entity 0, i.e., agent i itself.

To act accordingly with adjacent agents, the intelligent agent is preferable to infer the importance of each feature node from the perspective of other agents in proximity. An natural question is "what would be other agent's belief if other agent had the same observation with me ?" Actually, in the interaction-based attention layer, the belief of the kth entity over the feature nodes is implicitly modeled and inferred as $\boldsymbol{h}_{k,l}$, where $k \neq 0$. The output of the interaction-based attention layer is a set of attention coefficients $\mathcal{H} = \{\boldsymbol{h}_0, \boldsymbol{h}_1, ..., \boldsymbol{h}_M\}$.

Aggregation-based pooling Layer

The attention weights distribution $\boldsymbol{h}_k$ reflects the interaction between entity k and l $(0 \leq l \leq M)$. To obtain overall weights distribution of the feature nodes considering beliefs of all entities, we compress distribution in $\mathcal{H}$ to a single weights vector $\boldsymbol{h} = [h_0, h_1, ..., h_M]$. The aggregation part is instantiated with a MLP network and utilized to aggregate such importance factors to obtain the final attention weights distribution $\boldsymbol{h}$.

Then, to obtain the final observation abstraction based on the overall attention weights distribution, we employ the pooling mechanism based on the weighted sum of all feature nodes $\mathcal{E}$, which is expressed as

$$\boldsymbol{z} = pooling(\boldsymbol{h}, \mathcal{E}) = \sum_{k=1}^{M} h_k \boldsymbol{e}_k. \tag{2}$$

The pooling mechanism compresses all feature nodes in the latent space transformed with $\boldsymbol{W}$ by distinguishing their importance. Finally, the actor layer takes as input local observation $\boldsymbol{o}$ and its abstract $\boldsymbol{z}$, and outputs the action $\boldsymbol{a}$.

Training

The centralized joint action-value function parameterized by θ_Q is denoted as $Q(\mathrm{a}, \mathrm{o}; \theta_Q)$, which takes as input the joint actions and observations of all agents. The experience replay buffer $\mathcal{D}$ contains tuple of experiences of all agents, denoted as $(\mathrm{o}, \mathrm{a}, r, \mathrm{o}')$, where $\mathrm{o} =< \boldsymbol{o}_1, \boldsymbol{o}_2, ..., \boldsymbol{o}_N >$, $\mathrm{a} =< \boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_N >$, and $\mathrm{o}' =< \boldsymbol{o}'_1, \boldsymbol{o}'_2, ..., \boldsymbol{o}'_N >$. The centralized critic is updated by temporal difference method as

$$\mathcal{L}(\theta_Q) = \mathbb{E}_{\mathrm{o},\mathrm{a},r,\mathrm{o}'\sim\mathcal{D}}[(Q^{\mu}(\mathrm{a}, \mathrm{o}; \theta_Q) - y)^2], \tag{3}$$

$$y = r + \gamma Q(\mathrm{a}', \mathrm{o}'; \overline{\theta}_Q)|_{\boldsymbol{a}'_i = \mu(\boldsymbol{o}'_i; \overline{\theta}^{\mu}_i)}, \tag{4}$$

where $\overline{\theta}_Q$ and $\overline{\theta}^{\mu}_i$ are the corresponding parameters for the target network of actor and critic, respectively.

The policy gradient of $\mu_i(\boldsymbol{o}_i; \theta^{\mu}_i)$ can be written as:

$$\begin{aligned} \nabla_{\theta^{\mu}_i} J(\theta^{\mu}_i) = \mathbb{E}_{\mathrm{o},\mathrm{a}\sim\mathcal{D}}[&\nabla_{\theta^{\mu}_i}\mu_i(\boldsymbol{o}_i; \theta^{\mu}_i) \\ &\nabla_{\boldsymbol{a}} Q(\mathrm{o}, \mathrm{a}; \theta_Q)|_{\boldsymbol{a}=\mu_i(o_i;\theta^{\mu}_i)}] \end{aligned} \tag{5}$$

Experiments

In this section, we perform experiments with three tasks, *cooperative navigation*, *meeting up* and *predator prey*, which are based on the *multi-agent particle environment* (Lowe et al. 2017), a two dimensional world with agents and landmarks in continuous space and discrete time domain. In each task, agents can only obtain partial observation, act cooperatively in continuous space, and share a global reward. We compare ATOM with MADDPG (Lowe et al. 2017), PR2 (Wen et al. 2019), ACML (Mao et al. 2020) and ATOC (Jiang and Lu 2018). MADDPG acts as the benchmark for centralized training and decentralized execution without either opponent modelling or communication channel. PR2 is an explicit opponent modelling approach, which shows the performance introduced by predicting actions of others using recursive reasoning without communication channels as ATOM. ACML and ATOC are both communication based approaches, which are baseline approaches augumented with communication channel. ACML allows communications among all agents, while ATOC limits communications within adjacent agents to filter useless messages for high-efficiency coordination.

Cooperative Navigation

Task Settings. We first start with a task named cooperative navigation, where all agents aim to cooperatively finding the right landmark for each agent to occupy. It consists of L agents and N landmarks. The team reward is based on the proximity of the agents to the landmarks, which is the sum of the negative distance between each landmark to its closest agent over all landmarks. Meanwhile, it is penalized when colliding with other agents, and $r_{collision} = -1$. At each episode, both the agents and landmarks are initialized with random positions in a given square region. Each agent must find its preferred landmark and control its trajectory to quickly occupy the landmark, while simultaneously avoiding the collisions with the other agents. It is better for each agent to have the knowledge of intentions and beliefs of other agents to assist the occupancy of right landmarks and collision avoidance. We train all methods with the settings of $L = 7$ and $N = 7$, where each agent can observe the relative position of the three nearest agents and four landmarks. The length of each episode is $T = 25$.

Table 1: Test Results for Cooperative Navigation.

	REWARD	OCCUPIED
ATOC	$\mathbf{-1.24} \pm 0.15$	$\mathbf{6.16} \pm 0.84$
ATOM	$\mathbf{-1.40} \pm 0.48$	$\mathbf{5.62} \pm 1.17$
ACML	$\mathbf{-1.89} \pm 0.18$	$\mathbf{3.06} \pm 0.68$
PR2	$\mathbf{-2.70} \pm 0.25$	$\mathbf{2.52} \pm 0.62$
MADDPG	$\mathbf{-2.63} \pm 0.46$	$\mathbf{2.57} \pm 1.02$

Quantitative Results. Figure 3 shows the learning curves of 100000 episodes in terms of mean reward, which is averaged over all agents and the episode length T. The shadowed area is enclosed by the min and max value of 5 training runs, and the solid line in the middle is the mean value (same for results in the sequel). For testing, we evaluate all methods by running 100 test games. Table 1 shows the results, including averaged mean reward and occupied landmarks. As expected, we can see ATOM outperforms both MADDPG and PR2 due to the consideration for the interaction among adjacent agents in the visibility area by fully exploiting local observation. ATOM without attentional modelling is exactly MADDPG, where ATOM largely outperforms MADDPG. It indicates that opponent modelling with only local observation indeed helps. PR2 maintains opponent model for other agents at each agent, however, the model only learns average behavior and operates only in the training phase for critic estimation, which limits its coordination ability during execution. We can also see ATOM performs better than ACML and approaches to performance of ATOM, where both ATOC and ACML allow messages exchange during execution to obtain opponent intention to help coordination. ATOC performs better than ACML, which indicates that the coordination of adjacent agents plays important role in cooperative tasks. This also explains why ATOM can achieve superior performance with only limited information from local observation.

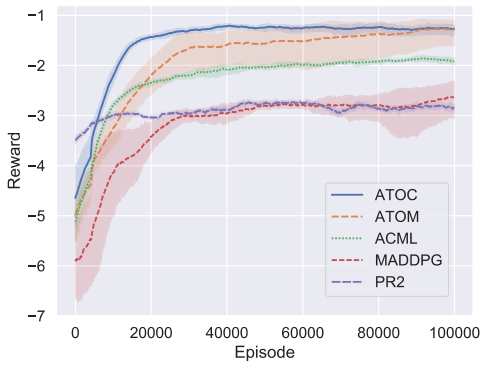

Figure 3: Learning Curve of Cooperative Navigation.

Model Interpretation. In the experiment, ATOM and ATOC largely outperforms MADDPG, ACML and PR2, which fail to learn the strategy that both ATOM and ATOC obtains. As shown in Figure 4, we can see ATOM can learn sophisticated policies, where an agent first tries to occupy the nearest landmark. However, when the landmark is more likely to be occupied by other agents in proximity, the agent will turn to another landmark rather than keeping ping-pong behavior in the proximity of the nearest landmark. For instance, in the ATOM illustration of Figure 4(a), A2 first tries to occupy L1, but considering that A1 also want to occupy L1, A2 move to L2 after several steps to avoid conflict. However, for ACML, MADDPG and PR2 their policies is to find the nearest landmark and without the appropriate coordination for the adjacent agents.

Figure 5 visualizes the aggregated attention weights $\boldsymbol{h}$ of each agent at different timesteps in one episode for the ATOC model. Recall that each element in $\boldsymbol{h}$ represents the relative importance of corresponding adjacent entity from the aspect of the given agent. The weights are visualized as

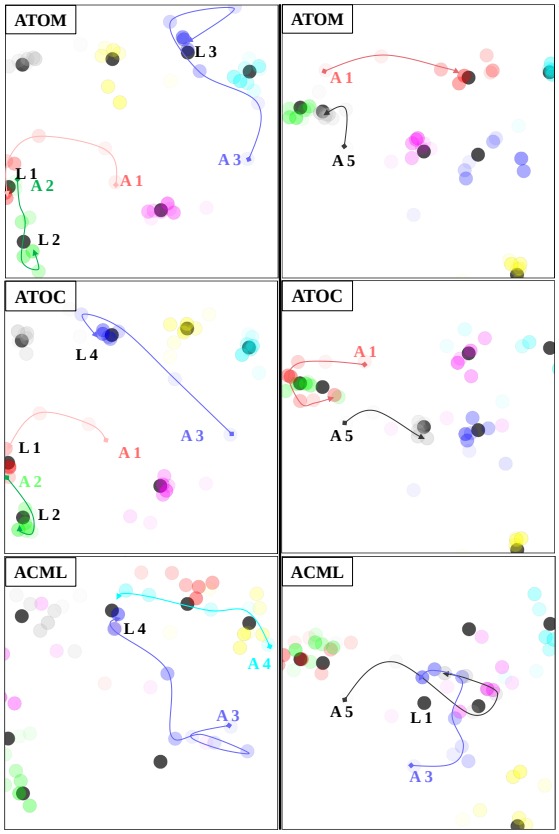

Figure 4: Illustration of the learned cooperative behavior for ATOM agents compared to ACML and ATOC agents. Each sub figures in the same column has the same initial positions for both agents and landmarks, specifically. The black circles are landmarks and the colorful circles represent trajectories of agents. 'A' stands for agent and 'L' stands for landmark.

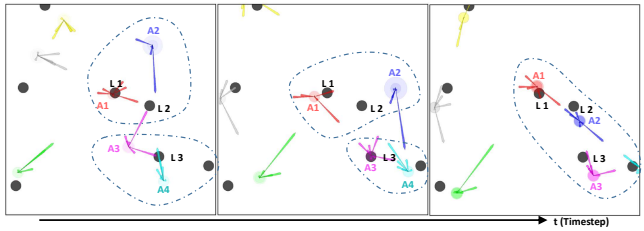

Figure 5: Illustration for the aggregated attention weights $\boldsymbol{h}$ in the aggregation based pooling layer. For an agent, the length of each arrow reflects the scale of each element in $\boldsymbol{h}$ of the agent, and each arrow points towards the entity corresponding to that element.

arrows around each agent, where the length of each arrow reflects the scale of each element in $\boldsymbol{h}$, and each arrow points towards the corresponding entity of the element. In this way, the longer arrow to an adjacent entity (including agent and landmark in this task) from certain agent suggests that the agent needs to pay more attention to the entity. In Figure 5(a), we can see that although both L1 and L2 are close to A2, A2 initially pays more attention to L2 by inferring the belief of A1, which is ready to occupy L1. The longest arrows of both A3 and A4 point to the possible landmarks to be occupied. It is notable that the attention weights does not represent the final action direction of each agent, because the actor layer of the agent network takes as input raw observation as well. For instance, A4 initially points to L3, but finally does not occupy it. In Figure 5(c), we can see that the longest arrows of A1, A2 and A3 point to closest agent, which is for avoidance of the possible collision. Besides, not all arrows in Figure 5 are meaningful, which means the aggregated attention weights may be inaccurate to guide the local decision. This is because the training of opponent model only depends on gradient of the centralized critic without any other supervised signal.

Meetup

Task Settings. Meetup is a modified version of simple scenario in (Lowe et al. 2017) to show the ability of consistent target choice in distributed manner. In the meetup task, all agents aim to collectively choose one landmark and congregate in the proximity of it. It consists of L agents and N landmarks. The team reward is based on the distance between all agents and the chosen landmark, which is the average of the negative distance between each agent and the landmark closest to all L agents. The goal landmark changes depending on the current position of all agents. At each episode, both the agents and landmarks are initialized with random positions in a given square region. Each agent must find the preferred landmark by others and optimize its trajectory to approach the landmark. Predicting intentions of others can help to quickly find the correct landmark. We set $N = 7$ and $L = 7$, where each agent can observe the relative position of the three nearest agents and four landmarks. The length of each episode is $T = 25$.

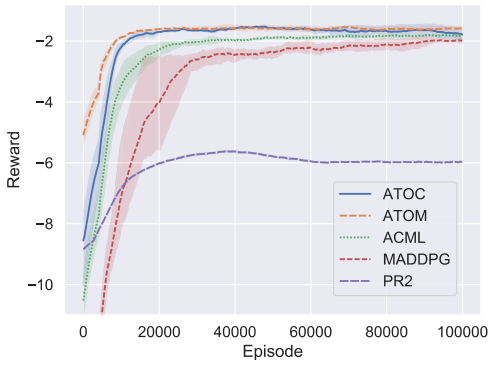

Figure 6: Learning Curve of Meetup.

Quantitative Results. Figure 6 shows the mean reward averaged over timesteps on the meeting up task. ATOM converges to the highest mean reward. For testing, we evaluate all methods by running 100 test games in terms of mean

reward. As shown in Table 2, ATOM also achieve the best mean reward. Intuitively, meeting up is a simpler task than cooperative navigation from the target allocation perspective. Because in meeting up task, only one target landmark is required to be determined, and then the movement control strategy is straightforward for each agent. However, PR2 performs extremely worse in terms of mean reward, where the difference between PR2 and other approaches increases compared to that in cooperative navigation. Unlike the cooperative navigation task, each agent finds its own target landmark while avoiding collisions with adjacent agents, agents in the meeting up task need to coordinate the global target landmark and the impacts introduced by inaccuracy of opponent model is amplified. Thus, the performance of PR2 is more worse. Nonetheless, ATOM performs slightly better than ATOC, which shows the power of inferring beliefs by implicitly modelling in ATOM.

Table 2: Test Results for the meetup task.

	REWARD
ATOC	$\mathbf{-1.55} \pm 0.31$
ATOM	$\mathbf{-1.50} \pm 0.39$
ACML	$\mathbf{-1.66} \pm 0.27$
PR2	$\mathbf{-6.09} \pm 0.15$
MADDPG	$\mathbf{-1.98} \pm 0.36$

Predator Prey

Task Settings. In the predator-prey task, N slower predators chase M faster preys to capture them in a square region. Due to the slower speed, it is impossible for a predator to capture a prey alone. Therefore, predators must cooperatively behave. We focus on the cooperation of predators rather than the competition between predators and preys, thus preys are restricted to specific area of activity and move in the opposite direction of the closest predator as in (Ding, Huang, and Lu 2020). The team reward of predators is the sum of negative distances of all the preys to their closest predators. The penalty for each collision between predators is set to $r_{collision} = -1$. In the experiment, we set $N = 7$ and $M = 3$, where each predator can observe three nearest agents and three preys with relative positions. The length of each episode is $T = 25$.

Quantitative Results. As shown in Figure 7, ATOM and ATOC converge to much higher mean reward, averaged over timesteps, than other baseline. Table 3 shows the mean reward for predator over 100 test runs. Comparing to the cooperative navigation task, the gap between ATOM and ACML/MADDPG in terms of average reward narrows down. This is because cooperation plays a more important role in predator prey, and the task is more complicated than the cooperative navigation. In particular, due to the higher velocity of prey, individual predator is almost impossible to capture a predator. Thus, when trying to catch up a prey, it is essential for adjacent predators to keep coordinated consistently. Therefore, the performance of both ATOM and ATOC are restricted due to the complexity, which makes the difference in terms of mean reward decreases.

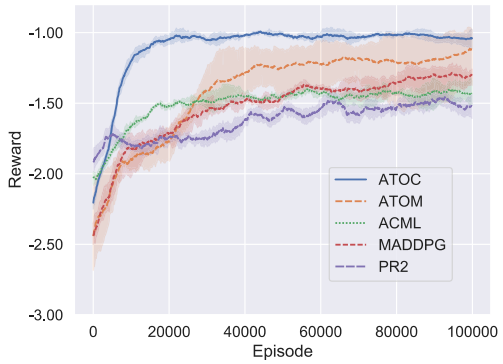

Figure 7: Learning Curve of the predator prey task.

Table 3: Test Results for Predator Prey Task

	REWARD
ATOC	$\mathbf{-0.97} \pm 0.12$
ATOM	$\mathbf{-1.10} \pm 0.35$
ACML	$\mathbf{-1.42} \pm 0.16$
PR2	$\mathbf{-1.51} \pm 0.11$
MADDPG	$\mathbf{-1.38} \pm 0.21$

Conclusion

We have proposed an attentional opponent model for multi-agent cooperation tasks, where agents predict the latent interaction between adjacent entities to implicitly predict their belief and also learn a aggregation pooling module to interpret high level abstraction for local decision. Unlike existing opponent modelling methods, ATOM can effectively and efficiently exploits local observation to understand the properties of agents in proximity for coordination. Some of the main advantages of our method are its simplicity and applicability, where ATOM does not require extra message exchange among agents during execution phase. Empirically, ATOM can learn sophisticated strategy for cooperation tasks and outperforms baselines without communication as expected. To our surprise, ATOM can achieve and even outperform methods with intention sharing among agents augmented with communication.

Acknowledgments

The authors would like to thank Dr. Kaiyang Guo and Dr. Ying Wen for their valuable suggestions and comments. This work was supported by the National Natural Science Foundation of China (NSFC) under Grant 62001509.

References

Albrecht, S. V.; and Stone, P. 2018. Autonomous agents modelling other agents: A comprehensive survey and open problems. *Artificial Intelligence*, 258: 66–95.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*.

Bernstein, D. S.; Givan, R.; Immerman, N.; and Zilberstein, S. 2002. The complexity of decentralized control of Markov decision processes. *Mathematics of operations research*, 27(4): 819–840.

Brown, G. W. 1951. Iterative solution of games by fictitious play. *Activity analysis of production and allocation*, 13(1): 374–376.

Ding, Z.; Huang, T.; and Lu, Z. 2020. Learning individually inferred communication for multi-agent cooperation. In *Advances in neural information processing systems*.

Foerster, J.; Farquhar, G.; Afouras, T.; Nardelli, N.; and Whiteson, S. 2018a. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Foerster, J. N.; Chen, R. Y.; Al-Shedivat, M.; Whiteson, S.; Abbeel, P.; and Mordatch, I. 2018b. Learning with opponent-learning awareness. In *International Conference on Autonomous Agents and Multi Agent Systems*, 122–130.

Gmytrasiewicz, P. J.; and Doshi, P. 2005. A framework for sequential planning in multi-agent settings. *Journal of Artificial Intelligence Research*, 24: 49–79.

Grover, A.; Al-Shedivat, M.; Gupta, J.; Burda, Y.; and Edwards, H. 2018. Learning policy representations in multiagent systems. In *International conference on machine learning*, 1802–1811.

He, H.; Boyd-Graber, J.; Kwok, K.; and Daumé III, H. 2016. Opponent modeling in deep reinforcement learning. In *International conference on machine learning*, 1804–1813.

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

Jiang, J.; and Lu, Z. 2018. Learning attentional communication for multi-agent cooperation. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018*, 7265–7275.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2015. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*.

Lowe, R.; Wu, Y.; Tamar, A.; Harb, J.; Abbeel, P.; and Mordatch, I. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in Neural Information Processing Systems*, 30.

Mao, H.; Zhang, Z.; Xiao, Z.; Gong, Z.; and Ni, Y. 2020. Learning agent communication under limited bandwidth by message pruning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 5142–5149.

Mitchell, B.; Tosun, H.; and Sheppard, J. 2015. Deep learning using partitioned data vectors. In *2015 International Joint Conference on Neural Networks (IJCNN)*, 1–8. IEEE.

Papoudakis, G.; Christianos, F.; and Albrecht, S. V. 2021. Local Information Agent Modelling in Partially-Observable Environments. arXiv:2006.09447.

Peng, P.; Wen, Y.; Yang, Y.; Yuan, Q.; Tang, Z.; Long, H.; and Wang, J. 2017. Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games. *arXiv preprint arXiv:1703.10069*.

Rabinowitz, N.; Perbet, F.; Song, F.; Zhang, C.; Eslami, S. A.; and Botvinick, M. 2018. Machine theory of mind. In *International conference on machine learning*, 4218–4227.

Raileanu, R.; Denton, E.; Szlam, A.; and Fergus, R. 2018. Modeling others using oneself in multi-agent reinforcement learning. In *International conference on machine learning*, 4257–4266.

Rashid, T.; Samvelyan, M.; Schroeder, C.; Farquhar, G.; Foerster, J.; and Whiteson, S. 2018. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, 4295–4304.

Shoham, Y.; Powers, R.; and Grenager, T. 2007. If multi-agent learning is the answer, what is the question? *Artificial intelligence*, 171(7): 365–377.

Sondik, E. J. 1971. The optimal control of partially observable Markov processes. Technical report, Stanford Univ Calif Stanford Electronics Labs.

Sukhbaatar, S.; Szlam, A.; and Fergus, R. 2016. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems*, 2244–2252.

Sunehag, P.; Lever, G.; Gruslys, A.; Czarnecki, W. M.; Zambaldi, V.; Jaderberg, M.; Lanctot, M.; Sonnerat, N.; Leibo, J. Z.; Tuyls, K.; et al. 2018. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, 2085–2087.

Sutton, R. S.; Barto, A. G.; et al. 1998. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Wang, X.; Girshick, R.; Gupta, A.; and He, K. 2018. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 7794–7803.

Wen, Y.; Yang, Y.; Luo, R.; Wang, J.; and Pan, W. 2019. Probabilistic recursive reasoning for multi-agent reinforcement learning. In *International Conference on Learning Representations*.

Xie, A.; Losey, D. P.; Tolsma, R.; Finn, C.; and Sadigh, D. 2020. Learning latent representations to influence multi-agent interaction. In *Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, 575–588.

Zhang, K.; Yang, Z.; and Başar, T. 2019. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *arXiv preprint arXiv:1911.10635*.