Emergence of Pragmatics from Referential Game between Theory of Mind Agents

Luyao Yuan
yuanluyao@ucla.edu

Zipeng Fu
fu-zipeng@engineering.ucla.edu

Jingyue Shen
brianshen@ucla.edu

Lu Xu
luxunxn7@ucla.edu

Juhong Shen
jhshen@ucla.edu

Song-Chun Zhu
sczhu@cs.ucla.edu

Department of Computer Science
University of California, Los Angeles

Abstract

Pragmatics studies how context can contribute to language meanings. In human communication, language is never interpreted out of context, and sentences can usually convey more information than their literal meanings. However, this mechanism is missing in most multi-agent systems, restricting the communication efficiency and the capability of human-agent interaction. In this paper, we propose an algorithm, using which agents can spontaneously learn the ability to "read between lines" without any explicit hand-designed rules. We integrate theory of mind (ToM) in a cooperative multi-agent pedagogical situation and propose an adaptive reinforcement learning (RL) algorithm to develop a communication protocol. ToM is a profound cognitive science concept, claiming that people regularly reason about other's mental states, including beliefs, goals, and intentions, to obtain performance advantage in competition, cooperation or coalition. With this ability, agents consider language as not only messages but also rational acts reflecting others hidden states. Our experiments demonstrate the advantage of pragmatic protocols over non-pragmatic protocols. We also show the teaching complexity following the pragmatic protocol empirically approximates to recursive teaching dimension (RTD).

1 Introduction

The study of emergent languages has become an important topic in cognitive science and artificial intelligence for years, as effective communication is the prerequisite for successful cooperation in both human society [1, 2] and multi-agent systems (MAS) [3, 4, 5]. Communication, as Grice characterized in his original model of pragmatics [6], should follow cooperative principles, where listeners make exquisitely sensitive inferences about what utterances mean given their knowledge of the speaker, the language, and the context, and the speaker collaboratively makes contributions to the conversational goals [7, 8]. For example, if there are three boys in a class and teacher A told teacher B that "**some** of the boys went to the party", then teacher B will legitimately infer that one or two of the boys went to the party [6, 9]. Although according to the literal meaning, it could be that all three of the boys went to the party, teacher B is less likely to interpret the utterance that way, because teacher A, a cooperative speaker, would have said "**all** of the boys went to the party" in that case to prevent ambiguity. This seemingly trivial example (called scalar implicature in pragmatics) illustrates that pragmatic conversation is, in most of the time, taken for granted in human communication,

and shows how significant hidden information can be acquired from literal meanings. However, current MAS tend to model communication merely as information exchange between agents, among which messages are deciphered only by their literal meanings [10, 11, 3, 12]. Even with perfect understanding among each other, this type of communication cannot achieve optimal efficiency, as the intention of the communicator implicitly suggested by the message is ignored.

To perform pragmatic communication, the listener needs to not only comprehend the speaker's utterance, but also infer his mental states. It has been shown that humans, during an interaction, can reason about others' beliefs, goals, intentions and predict opponent/partner's behaviors [13, 14, 15], a capability called ToM. In some cases, people can even use ToM recursively, and form beliefs about the way others reason about themselves [16]. Thus, in order to collaborate and communicate with people smoothly, artificial agents must also bear similar potentially recursive mutual reasoning capability. Despite the recent surge of multi-agent collaboration modeling [10, 12, 17, 18], integrating ToM is still a nontrivial challenge. A few approaches attempted to model nested belief of other agents in general multi-agent systems, but extensive computation restricts the scale of the solvable problems [19, 20]. When an agent has an incomplete observation of the environment, it needs to form a belief, a distribution over the actual state of the environment, to take actions [14, 20]. ToM agents, besides their own beliefs about the state, or 0-th level beliefs, also model other agents' beliefs, forming 1-st level beliefs. They can further have beliefs about others' 1-st level beliefs about their 0-th level belief, so on and so forth [19, 21, 14, 16, 22]. The intractability of distribution over distribution makes exact solving for ToM agents' nested beliefs extremely complicated [19].

Therefore, an approach to acquire the sophistication of high-level recursions without getting entangled into the curse of intractability is needed. In this paper, we propose an adaptive training process, following which pragmatic communication protocol can emerge between cooperative ToM agents modeling only the 1-st level belief over belief. The complexity of higher level recursions can be preserved by the dynamic evolving of agents' tractable belief estimation functions. We don't assume agent has a certain level of recursions, which requires modeling nested beliefs from the 0-th level up to the desired level [19, 21, 16, 22]. Instead, we directly learn a function to approximate partners' actual beliefs and how to react accordingly. In cooperative games, this learning becomes mutual adaptation, with controlled exploration rate, improving the performance of the multi-agent system [23]. Intuitively, for a pair of agents, we update them alternatively, namely, fixing one while training the other, simulating the iterative best response (IBR) model, which proves to converge to a fixed point Nash equilibrium in strong and weak interpretation game [9]. We demonstrate the effectiveness and advantage of the pragmatic protocol with referential games, a communication game widely used in linguistic and cognitive studies in the context of language evolution [24, 25, 5]. It provides a good playground for pragmatic and pedagogical interactions between a teacher and a student and can easily generalize to a comprehensive teaching task with a large concept space.

We tested our algorithm and evaluated the pragmatic teaching protocol with both symbolic and pixel data, and achieved significant performance gain over previous algorithms in terms of referential accuracy. Also, we found that if messages are grounded with human expertise before interaction, the emerged protocol achieves teaching complexity empirically approximates recursive teaching dimension (RTD) [26], the worst-case number of examples required for cooperative agents concept learning [27].

This paper makes two major contributions: 1) we showed that a pragmatic communication protocol can emerge between ToM agents through adaptive reinforcement learning. This protocol significantly improves the cooperative multi-agent communication by enabling the agents to extract hidden meanings from the context to enrich the literal information; 2) we proposed an algorithm to develop protocols empirically approximating the teaching complexity bound between cooperative agents provided by RTD. Extensive experiments on both the symbolic and 3D object datasets demonstrate the effectiveness of our proposed protocol.

2 Background: Referential Game

There are a teacher and a student in a referential game. The teacher has a target in mind and aims to send a message to the student so that the student can identify the target out of a set of distractors after receiving this message. The motivation of our algorithm is the rational speech act (RSA) model between ToM agents (also termed as bilateral optimality [28]): in order to establish proper

communication, the speaker has to take into account the perspective of the listener, while the listener has to take into account the perspective of the teacher [8]. Figure 1a shows an example. There are three objects, a blue sphere, a red sphere, and a blue cone. Suppose the target is the blue sphere. If the only allowed messages are colors and shapes, then, for a literal student, there is no unique identifier for the blue sphere, because both "blue" and "sphere" have more than one consistent candidates. Nonetheless, a pragmatic student, after hearing "blue" from the teacher, should be able to do counterfactual reasoning and identify the blue sphere instead of the blue cone, because he knows the teacher is helpful and would have used cone to refer to the blue cone unambiguously. During pragmatic communication, a message conveys more information than the message itself. The usage of that message can usually suggest the intention of the teacher.

The referential game can be formally defined by a tuple $\langle A, B, \Omega, \mathcal{M}, \mathcal{A} \rangle$, where A and B stand for a teacher and a student. Ω is the instance space, where the distractors and targets are sampled from. $\mathcal{M}$ is the message space and $\mathcal{A}$ is the student's action space. In a specific game, a set of instances $O \subseteq \Omega$ is sampled from Ω as candidates, and one of the candidates $o^\star \in O$ is designated as the target, while the rest, $O/\{o^\star\}$, are distractors. The candidates O are available to both of the agents, while only the teacher knows the target, $o^\star$. Agents take turns in this game. In every round, the teacher first sends a message $m_t \in \mathcal{M}$ to the student, followed by an action $a_t \in \mathcal{A} = \{1, 2, ..., |O|, \Xi\}$ taken by the student, where number 1 to $|O|$ represent "identify a certain instance as the target", Ξ means "wait for next message" and t stamps the t-th round. Every message comes with a message cost, c_m, and the total gain for both the teacher and the student, given a_T the first non Ξ action, is $R = \sum_{t=1}^{T} -c_{m_t} + \mathbf{1}(o^\star = O[a_T])$. Notice that the game ends when the student performs a non Ξ action. We define a protocol between A and B as a set of policies

$$\Pi = \langle \pi_A : \mathbb{P}(\Omega) \times O \times \mathcal{M}^* \times \mathcal{M} \to [0,1], \pi_B : \mathbb{P}(\Omega) \times \mathcal{M}^* \times \mathcal{M} \times \mathcal{A} \to [0,1] \rangle$$

$\mathbb{P}(\Omega)$ is the power set of Ω, where O sampled from, and $*$ is the kleene star, standing for the history of message. Intuitively, the teacher selects a message based on the distractors, the target and communication history. The student chooses an action according to candidates, history and the latest message. The goal for both of the agents is to maximize the expected gain:

$$\mathbf{E}_{\substack{O \sim \mathbb{P}(\Omega), o^\star \sim O, \\ m_{1:T} \sim \pi_A, a_{1:T} \sim \pi_B}} \left[-\sum_{t=1}^{T} c_{m_t} + \mathbf{1}(o^\star = O[a_T]) \right] \tag{1}$$

3 Related Work

Pragmatics [6] has profound linguistic and cognitive science origin dating back to 1970s. However, the integration of this topic with multi-agent games only draws people's attention in this decade. The RSA model was raised by Golland et al. [29] and developed in [30, 31, 8, 32] in later works. Yet, all of these works model only single utterance and require hand designed production rules for the agents, while in our algorithm, all policies are learned by the agents and multi-round of communication is allowed. [33, 7, 34] model pragmatic reasoning for human action prediction, but they all require domain-specific engineering, and pragmatically annotated training data. In our work, all models are trained by self-playing with RL. [7] needs human data to initialize the literal speaker and uses RL to train listener with pragmatic reasoning, but they assumes fixed literal speaker. On the contrary, we endow both the teacher and the student ToM and allow them to do mutual pragmatic reasoning. The discriminative best response algorithm in [35] is also inspired the IBR model [9], but they used supervised learning given language data instead of RL. Also, our game states are more complicated than their toy examples.

Emergence of language in communication games. Emergence of language is a topic about a group of agents developing a communication protocol to complete a task, which can either be cooperative or competitive. In most recent studies, agents start with a set of ungrounded symbols and first learn to ground these symbols using reinforcement learning approaches supervised by the task rewards. The major novelty of our work comparing with previous methods is that pragmatic reasoning can emerge from our protocols without explicit rules coded.

In [5], two agents try to develop a protocol through playing the referential game, in which the teacher sees only the target but no distractors, eliminating the possibility of taking advantage of ToM, as no counterfactual reasoning can happen at the student side. In [24] the teacher sends a message according to the context of candidates, but no student reaction is simulated before the selection.

In [36, 37], the teacher can simulate the reaction of a fixed student, who does not model teacher's mind and cannot benefit from counterfactual reasoning about teacher's intention behind messages, terminating the recursive process after only one cycle.

A variation of the referential game was played in [38], where the teacher sees an image and the student sees a set of descriptions. The goal is for the student to identify a suitable description for the teacher's image by multi-round communication. A similar multi-round communication game using natural language was also played in [17]. In these games, the student can ask further questions after receiving the first message and is in charge of the final decision making, while the teacher will answer the question based on the target and the communication history. These works focus on learning shared embedding between objects and messages instead of grounding messages to attributes that reappear across the object space. Also, neither of these papers include agents who model their partner's minds.

Multiagent communication. Modelling multiagent communication dates back to 1998 when Kinney et al. [10] proposed adaptive learning of multiagent communication strategy as a predefined rule-based control system. To scale up from rule-based systems, decentralized partially observable Markov decision process (DEC-PODMP) was used to model multiagent interaction with communication as a special type of action among agents [11, 39]. Solving DEC-POMDP exactly is a NEXP-complete problem [11] requiring agents to remember the complete observation history for their policy. In recent works, a more compact representation of the history is often used for action selection. In [12, 4], agents maintain a memory variable and refer to it when act and speak. However, the centralized training process in these work needs channels with large bandwidth to pass gradients across different agents. Also, the messages in [12] are not discrete symbols, but continuous outputs of neural networks. Other extension of single agent value based algorithms to multiagent problems [40, 4] usually suffers from non-stationariy induced by simultaneous updates of agents.

While DEC-POMDP considers other agents as part of the environment and learns the policy as a mapping from local observation to action, LOLA in [41] learns the best response to evolving opponents. Yet, opponent/partners' real-time belief is not considered into policy. Interactive-POMDP (I-POMDP) [19, 20] moves one more step forward by actually modeling opponents' mental states at the current moment and integrates other's belief into the agent's own policy. However, I-POMDP requires extensive sampling to approximate the nested integration over the belief space, action space and observation space, limiting its scalability. Because whenever the teacher sends a message, she thinks about not only the student's current belief but also current distractor sets, the value iteration process in I-POMDP needs to be repeated for every single game. In our algorithm, training only needs to be performed once for one pair of agents and reusable to all games. The Bayesian action decoder (BAD)-MDP proposed by Foerster et al. [42] also yields counterfactual reasoning in their belief update, but their method is more centralized in the testing process than ours. The BAD-agent is a super-agent controlling all other agents collectively. Deterministic partial policies can easily reveal agents' private information to the BAD-agent and make it public. Instead, our model doesn't depend on any implicit information flowing between agents during testing.

4   Adaptive Emergence of Pragmatic Protocol

Emergence of Pragmatic Protocol: Our goal is to learn a protocol for agent A and B so that they can communicate with the contextual information being considered. To avoid tracking the message history, which scales exponentially with the time, we use beliefs as sufficient statistics for the past. Hence, ToM can be embodied as estimating partner's current and future belief, then choose the most ideal action to manipulate them as needed. In the referential game, since the teacher knows the target, only the student holds a belief, b_B, about the target. Utilizing the obverter technique [36, 37], we let the teacher holds a belief b_{BinA} as her estimation of student's belief. As b_{BinA} is an estimation of the student's belief, it should be a belief over belief, i.e. a distribution over a distribution over the candidates. Since a distribution is a continuous random variable, distribution over a continuous variable can be represented as a set of particles. However, to avoid the complexity we only use one particle to approximate. That is, b_{BinA} is still a distribution over the candidates. This is reasonable because the belief update process is deterministic for rational agents following the Bayesian rule [7, 43]. Given b_B^0 a uniform distribution over candidates, $P(b_B^t)$ is a single mode distribution and can be approximated with a particle.

(a) Referential Game
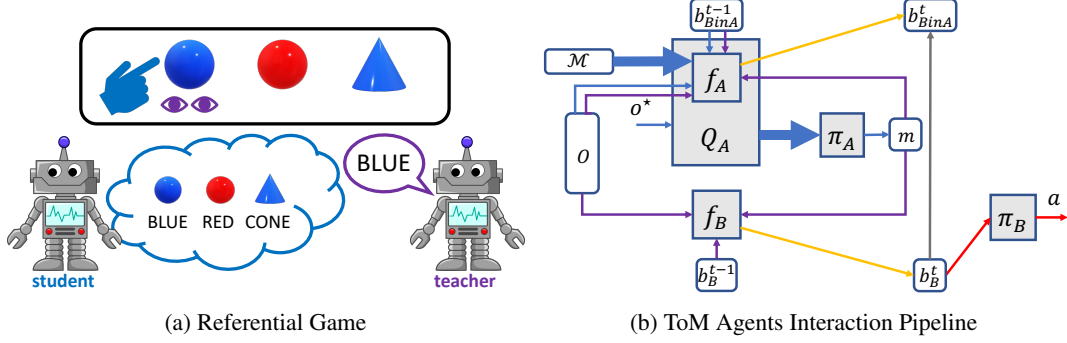
(b) ToM Agents Interaction Pipeline

Figure 1: (a) An example referential game. There are three objects, a blue sphere, a red sphere, and a blue cone. If a student hears "blue" from the teacher, he should be able to identify the blue sphere instead of the blue cone. (b) ToM Agents Interaction Pipeline. First, the teacher chooses a message according to the context and her prediction of the student's reaction (**blue** arrows). After a message is sent, the student updates his belief and the teacher updates her estimation of student's belief (**purple** and **orange** arrows). Then, the student either waits or selects a candidate (**red** arrows). Only in the training phase, the actual student belief will be returned to the teacher (**gray** arrow). Bold arrows stand for the whole message space being passed. Notice f_A is part of the Q_A. O and b_{BinA}^{t-1} are passed to f_A twice, for message selection and teacher's new belief estimation. Empty boxes are game and time variants; shadowed boxes are agents' mental structures. Notations are introduced in section 4 with θ omitted from subscripts.

Algorithm 1 Iterative Adaption Protocol Emergence

1: Randomly initialize $\theta_A, \theta'_A, \theta_B$
2: No. candidates K
3: Learning rate η, Batch size N
4: **for** each phase **do**
5:    **for** $i \in \{A, B\}$ **do**
6:       Initialize replay buffer $\mathcal{D} \leftarrow \emptyset$
7:       **while** train agent i **do**
8:          $t = 1$
9:          Initialize $\mathcal{E} \leftarrow \emptyset$
10:         **repeat**
11:            **if** $t = 1$ **then**
12:               Sample $O = \{\omega_1, ..., \omega_K\}$
13:               Random select $o^\star = \omega_j$
14:               Initialize b_B^0, b_{BinA}^0
               as uniform distribution
15:            **end if**
16:            $m_t \sim \pi_{\theta_A}(\cdot|O, o^\star, b_{BinA}^{t-1})$
17:            $b_B^t = f_{\theta_B}(O, b_B^{t-1}, m_t)$
18:            $a_t \sim \pi_{\theta_B}(b_B^t)$
19:            $r_t = -c_{m_t} + \mathbf{1}(O[a_t] = o^\star)$
20:            $b_{BinA}^t = b_B^t$
21:            **if** $i = A$ **then**
22:               $\mathcal{D} \leftarrow \mathcal{D} \cup \{(O, o^\star, b_{BinA}^{t-1}, m_t, b_B^t, r)\}$
23:            **else**
24:               $\mathcal{E} \leftarrow \mathcal{E} \cup \{(O, b_B^{t-1}, m_t, a_t, r)\}$
25:            **end if**
26:            $t \leftarrow t + 1$
27:         **until** $a_t \neq \Xi$
28:         **if** $i = A$ **then**
29:            Sample $\{(O, o^\star, b_{BinA}^{t-1}, m_t, b_{BinA}^t, r)\}_N \sim \mathcal{D}$
30:            $\xi = r + \gamma \arg\max_m Q_{\theta'_A}(O, o^\star, b_{BinA}^t, m)$
31:            $L^Q = \frac{1}{N} \sum_N ||\xi - Q_{\theta_A}(O, o^\star, b_{BinA}^{t-1}, m_t)||^2$
32:            $L^{Obv} = \frac{1}{N} \sum_N H(b_{BinA}^t, f_{\theta_A}(O, b_{BinA}^{t-1}, m_t))$
33:            $\theta_A \leftarrow \theta_A - \eta \nabla_{\theta_A}(L^Q + \lambda L^{Obv})$
34:            Update $\theta'_A \leftarrow \theta_A$ periodically
35:         **else**
36:            Compute $R_t = \sum_{k=t}^{|\mathcal{E}|} \gamma^{k-t} r_k$ for t in $1, ..., |\mathcal{E}|$
37:            $J = \frac{1}{|\mathcal{E}|} \sum_{t=1}^{|\mathcal{E}|} \log \pi_{\theta_B}(a_t|f_{\theta_B}(O, b_B^{t-1}, m_t)) R_t$
38:            $\theta_B \leftarrow \theta_B + \eta \nabla_{\theta_B} J$
39:         **end if**
40:       **end while**
41:    **end for**
42: **end for**

Before speaking, teacher traverses all messages and predicts the student's new belief after receiving each message. She then sends the message leading to the most optimal student's new belief. Hearing the message, student updates his belief and takes action. This process is visualized in figure 1b and formalized in algorithm 1 line 11 to 20. The recursive mutual modeling in ToM is integrated within the belief update process. $f_{\theta_i}, i \in \{A, B\}$ are belief update functions parameterized by θ_i, taking in candidates, current belief, message and returning a new belief. The beliefs in our model are semantically meaningful hidden variables in teacher's Q-function and student's policy network, as the student directly samples an action according to his belief. The evolving of the belief update function reflects the protocol dynamics between the agents. Within f, we code in the Bayesian rule with the

likelihood function varying across different training phases. Our implementation detail can be found in section B. In each phase, we first train the teacher for a fixed student, then adapt the student to the teacher.

Difference from multiagent Q-learning: Our algorithm considers both the physical state and agent's mental state in the value function, and has a dynamic belief update function. Moreover, since agents are never trained simultaneously, our algorithm doesn't suffer from non-stationarity [40, 4].

Teacher: The teacher selects messages according to her Q-values and belief update function. We use $f_{\theta_A}(O, b, m)$ to denote teacher's belief update function, which takes in the candidates set, current belief estimation and a message. The return value of this function is a new belief estimation $b' \in \Delta O$. ΔO represents all probabilistic distributions over the candidates. This function can be parameterized as a neural network with weighted candidates encoding and messages as inputs and softmax as the output layer. The return value of the belief update function is directly fed into the Q-function. In practice, we implement it as a submodule of the Q-net. That is, the output of the belief update function is used in A's Q-function and to predict student's belief in next step during testing. The teacher chooses messages according to her Q-value following equation 2.

$$\pi_{\theta_A}(m|O, o^\star, b) = \frac{\exp\left(\beta Q_{\theta_A}(O, o^\star, b, m)\right)}{\sum_{m' \in \mathcal{M}} \exp\left(\beta Q_{\theta_A}(O, o^\star, b, m')\right)} \tag{2}$$

$$Q_{\theta_A}(O, o^\star, b, m) = \mathbf{E}_{a \sim \pi_{\theta_B}\left(f_{\theta_B}(O, b, m)\right)}\Big[\mathbf{1}(O[a] = o^*) + \tag{3}$$
$$\mathbf{1}(a = \Xi) \max_{m'} Q_{\theta_A}\left(O, o^\star, f_{\theta_B}(O, b, m), m'\right)\Big] - c_m$$

Equation 3 defines the teacher's Q-function. $\mathbf{1}(O[a] = o^*)$ indicates whether the student makes a correct prediction. $\mathbf{1}(a = \Xi)$ indicates if the game is still ongoing. Student's belief b is the state of teacher's MDP. Since student's actions determine the game states, the expectation is over the student's policy. By definition, the teacher's Q-function relies on student's policy and belief update function. She has no access to these student's functions, but since we never train the teacher and student simultaneously, the expectation can be approximated through Monte-Carlo (MC) sampling. To form a protocol, agent A needs to learn two functions, her belief update function f_{θ_A} and Q_{θ_A}. In the training phase, every time the student receives a message, he returns his new belief b_B^t to the teacher. During testing, she needs to use the output of f_{θ_A} to approximate student's new belief. We train f_{θ_A} by minimizing the cross-entropy, H, between b_B^t and teacher's prediction, denoted as L^{Obv}, the obverter loss. Teacher's Q-function is learned with Q-learning [44]. The λ in line 33 controls the scale of the two losses.

Student: We directly learn the belief update function and policy of the student through the REINFORCE algorithm [45]. In the referential game, student's policy is quite simple. If his belief is certain enough, he will choose the target based on his belief; otherwise, wait for further messages. The output of the policy network is a distribution with $|O| + 1$ dimensions. The last dimension is a function of the entropy of the original belief. If the belief is uncertain, this value will be dominant after normalization. f_{θ_B} has the same structure as f_{θ_A}. f_{θ_B} and π_{θ_B} can be parameterized as an end-to-end trainable neural network, with the candidates encoding, original belief and received a message as the input and returning an action distribution.

Adaptive Training: The whole training process can then be summarized as Algorithm 1. Both the teacher and student are adaptively trained to maximize their expected gain defined in Eq. (1). The training details for the teacher and the student are illustrated in Line 28-34 and Line 35-39 of Algorithm 1 respectively.

5   Experiments

We evaluated our algorithm with two datasets, number set and 3D objects, and played referential games with four or seven candidates. The number set is a symbolic dataset, with an instance as a set of categorical numbers. For example, $[(1, 2, 3, 9), (1, 2, 4), (2, 3), (3, 4, 5)]$ consists a referential game with four candidates. Notice that the numbers are merely symbols without numerical order. If there are four candidates, we randomly choose numbers from 0 to 9, with maximum four numbers in a set; if seven candidates, we choose from 0 to 11, with maximum five numbers in a set. Each set is
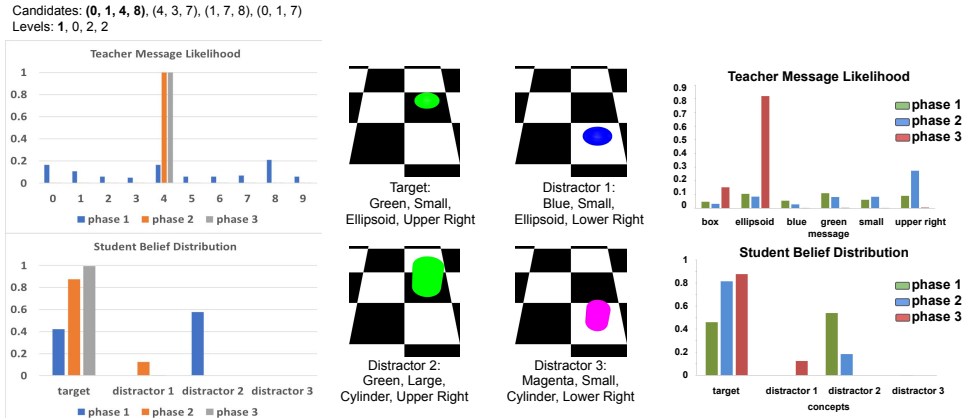
Figure 2: 4 distractors referential game example. Number set on the left (candidates listed in the title with the target in bold fonts) and 3D objects on the right. Due to the space limit, we only show the message distribution for the target and student's new belief after receiving the most probable message. As for the teacher's message distribution for distractors, all probability weights concentrate on the unique identifiers after the first phase of training. Student's belief illustrates that teacher's most probable message, though consistent with multiple candidates, can successfully indicate the target with more confidence as training goes. In general, both agents' behavior becomes more certain, and the certainty coordinates.

Table 1: The **first three rows** are the total accuracy, while the **second three rows** measure the accuracy for difficult games (numbers in percentage). We define game difficulty with average cosine similarity between the target and the distractors. The larger the cosine similarity, the harder the game is. We report the accuracy for top 10% hard games. L.A. stands for Lazaridou et el. To verify the generalization of our algorithm, we form training sets with 70% of all instances and test using the rest 30% unseen instances. The performance doesn't exhibit noticeable degradation: $\mathbf{99.1 \pm 0.1}\%$ and $\mathbf{93.0 \pm 1.7}\%$ for 4 and 7 candidates number set respectively (comparable with the No. Set results in the third line). Mean and std calculated using 3 different random splits, 2 experiments per split.

No. Set	3D objects	$\leftarrow 4 \mid 7 \rightarrow$	No. Set	3D Objects
79.1 ± 3.3	86.9 ± 4.1	L. A. [2018]	64.2 ± 6.0	77.3 ± 2.8
96.8 ± 0.2	97.0 ± 0.5	L. A. [2017]	80.8 ± 3.1	88.2 ± 1.7
$\mathbf{98.9 \pm 0.1}$	$\mathbf{99.6 \pm 0.2}$	Pragmatics	$\mathbf{93.2 \pm 1.0}$	$\mathbf{97.4 \pm 1.2}$
79.3 ± 3.1	86.9 ± 4.5	L. A. [2018]	67.2 ± 5.8	77.2 ± 2.6
91.5 ± 0.4	88.0 ± 1.9	L. A. [2017]	66.2 ± 3.1	68.2 ± 2.7
$\mathbf{98.1 \pm 0.3}$	$\mathbf{98.8 \pm 0.3}$	Pragmatics	$\mathbf{88.3 \pm 0.6}$	$\mathbf{94.1 \pm 2.3}$

encoded by multi-hot encoding. There are 385 and 1585 different possible number sets, consisting up to 9.0×10^9 and 4.9×10^{18} different games with four and seven candidates. Number sets make a generic referential game prototype, where each instance can be disentangled into independent attributes perfectly. To verify the generality of our algorithm on more complicated candidates, we used MoJoCo physical engine to synthesize RGB images of resolution 128×128 depicting single 3D object scenes. For each object, we pick one of six colors (blue, red, yellow, green, cyan, magenta), six shapes (box, sphere, cylinder, pyramid, cone, ellipsoid), two sizes and four locations, resulting in 288 combinations. In every game, candidates are uniformly sampled from the instances space. We use a message space with the same size as the number of attributes appeared in the dataset, i.e., 10 or 12 for number set, and 18 for 3D objects. In every game, we only allow one round of communication with one message. To prevent collusion using trivial position indicator, candidates are presented to the agents in different orders.

5.1 Referential Game with Symbolic and Pixel Input

For number set, we encode candidates with multi-hot vectors and messages with one-hot vectors. For 3D objects, we used a convolutional neural network (CNN) to extract features of the candidates. For all datasets, we manually generated 600k and 100k mutually exclusive games for training and testing. Namely, the same instances can appear in both datasets, but not any identical candidates combinations. To win a game, the same instance needed to be handled differently given different contexts, so, as
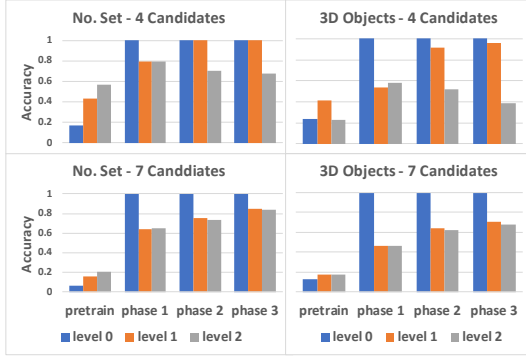
Figure 3: Finishing one phase of adaptation, agents can correctly identify level 0 targets. Namely, the teacher always sends a unique identifier as long as there is one. After phase 2, the student knows how the teacher will teach level 0 targets. Thus, he can prune out all level 0 candidates if he doesn't receive their unique identifiers, leaving level 1 candidates the "simplest". The decrease of level 2 accuracy among 4 candidates might be caused by the lack of level 3 targets. Only 3% of the number set and 0.02% 3D Objects games have level 2 targets, so agents have no motivation to explore for a higher level of reasoning.

long as the games are exclusive between training and testing sets, sharing instances won't cause over-fitting. To test the robustness, we also report results using exclusive testing instances in table 1. We compared the pragmatic protocol developed using our algorithm against previous works on referential game [24, 5]. Both utilized RL to train a protocol, but neither modeled recursive mind reasoning between agents. In [24], only the teacher considers the context, while no context is included in [5]. We trained our model for 3 phases, with 20k iterations for each phase and switch the training agent in the middle of every phase. Both benchmarks were trained for 100k iterations. Since there is no official code released by the authors, we implemented their model by ourselves and did thorough hyper-parameter grid search. Results shown in table 1. Our experimental results in all settings are significantly better than both of the benchmarks. Using the paired T-test, the one-tail p-value is smaller than 0.001 for all settings in table1. We found that even with simpler representation, number set games are more difficult than 3D objects, because we don't have any limitations generating the instances in number sets. 3D objects, on the other hand, form special cases of number sets, as some attributes can never coexist. E.g. a shape cannot be a sphere and a cone simultaneously.

5.2 Connection with RTD

The iterative adaptive idea of our algorithm is similar to the definition of RTD, which measures the number of examples needed for concept learning between a pair of cooperative and rational agents [27]. We included the formal definition of RTD in section A of the appendix. Intuitively, in a concept class, there is a subset of concepts which are the simplest to learn i.e. has the minimum sized teaching set among all concepts. One can first learn those concepts and remove them from the concept class. Now, for the remaining set of concepts, one can recursively learn the simplest concepts and so on. The teaching complexity of this learning schema lower bounds classic teaching dimension [26]. In every phase of our iterative training, the agent learns to identify the optimal teaching set for the "simplest" remaining candidates. In our experiments, candidates identifiable with a unique message are the simplest. If a candidate becomes the simplest after k times of removal, then we call it a level k candidate. To better illustrate the connection, we show two example referential games in figure 2 and the accuracy improvement after each phase of training in figure 3. We can see from figure 3 that after one phase of training all level 0 targets can be perfectly identified. Thus, the student, if shown the four 3D objects in figure 2, will know that the teacher will send "Blue" for distractor 1, "Large" for distractor 2 and "magenta" for distractor 3. Hence, "Upper Right" and "Ellipsoid", though consistent with multiple objects, must indicate the target. The accuracy for higher-level targets in figure 3 keeps increasing as they become uniquely identifiable after lower-level targets are pruned out. We can observe the emergence of pragmatics from these results. From the student's perspective, the messages from the teacher are no longer merely comprehended by their literal meanings, and from the teacher's perspective, she selects the most helpful message to teach. In the 4-candidate scenario, most questions with level 0 and 1 are correctly answered, a similar capability shown in human one-shot referential game study [46].

Notice that RTD is derived under the assumption that both agents only decipher the candidates as sets of discrete attributes. To eliminate the usage of other hidden patterns in the candidates, we need to pretrain the agents to ground messages to instances attributes. This can be easily achieved by initializing agents' belief update function as Bayesian belief update. That is, before running algorithm

No. Set	3D Objects	← 4 | 7 →	No. Set	3D objects
59.5 ± 0.4	53.3 ± 1.8	L. A. [2018]	35.0 ± 0.2	37.7 ± 1.8
52.3 ± 2.9	60.8 ± 2.9	L. A. [2017]	33.3 ± 2.0	46.0 ± 1.4
$\mathbf{97.5 \pm 0.4}$	$\mathbf{98.7 \pm 0.5}$	Pragmatics	$\mathbf{73.9 \pm 0.3}$	$\mathbf{84.2 \pm 0.1}$
81.5 ± 2.6	83.7 ± 3.4	L. A. [2018]	74.9 ± 0.5	75.3 ± 2.9
$\mathbf{98.6 \pm 1.0}$	98.1 ± 0.2	L. A. [2017]	$\mathbf{93.0 \pm 0.5}$	91.2 ± 0.5
98.1 ± 0.3	$\mathbf{99.7 \pm 0.3}$	Pragmatics	88.2 ± 0.8	$\mathbf{91.2 \pm 0.1}$

Table 2: All models are pretrained with Bayesian belief update. The three rows above are tested with agents seeing different but equivalent candidates, while the three rows below is tested with agents sees identical candidates. The larger the performance difference is, the more the protocol diverges from the initial grounding.

	Valid %		Accuracy	
	4	7	4	7
Pragmatics	$\mathbf{98.8 \pm 0.29}$	$\mathbf{98.1 \pm 0.36}$	$\mathbf{98.6 \pm 0.08}$	87.9 ± 0.61
L.A. [2017]	74.5 ± 5.69	79.1 ± 4.73	97.0 ± 0.02	$\mathbf{88.8 \pm 0.25}$

Table 3: A message is valid if it corresponds to one of the attributes of a novel target. Std calculated with 3 different random splits.

1, we train the belief neural network f with cross-entropy loss between the generated new belief and ground truth Bayesian belief. Afterward, every message grounds to an attribute.

Bayesian pretraining provides human decipherable examples for failure cases. Most of our model's mistakes are on targets with $RTD > 1$ or $RTD = 1$ but requiring high (≥ 2) level unique identifiers (hard games). Since we only allow one message, targets with $RTD > 1$ are theoretically impossible to be certainly identified, even between cooperative agents with ToM. As for the hard games, their relatively low frequency in the training set may impinge the acquiring of high-level best response. Failure to handle a certain type of scenarios is a common empirical defect of the current RL algorithms, in our case, the hard games. Another benefit of Bayesian pretrain is that the initial message grounding is decipherable to human. In the next section, we show this interpretability can be preserved by our algorithm.

5.3 Stability of the Protocol

We also explored how much the communication changes after the mutual adaptation process. Suppose the agents are initialized with human-understandable message groundings, ideally, we want the emerged protocol preserves its human interpretability. To test this property, we give the teacher and student different but equivalent candidates. Namely, we randomly generate a one-to-one mapping from attributes to attributes, such as replacing all 1 with 7 in number sets or replacing all red with blue in 3D objects. After the teacher sends a message to the student, both the message and candidates are converted with the same mapping before presented to the student. The converted candidates form an equivalent game for the original one. For example, if the teacher sees $[(0, 1, 5, 8), (0, 4, 6), (4, 8, 9), (0, 1, 4, 5)]$ and sends 4. Converted by a mapping which adds 1 to all attributes, the student gets $[(1, 2, 6, 9), (1, 5, 7), (5, 9, 0), (1, 2, 5, 6)]$ and 5.

We pretrain agents with Bayesian belief update separately, then train them together without equivalent games but test them with equivalent games. See table 2 for the results. We can see that our algorithm preserves human interpretability the most. The iterative adaptation contributes to stability because, in every phase, one agent is fixed while optimizing the other. Hence, the evolving of the protocol is not arbitrary and will maintain the effective part of the existing protocol while improving the rest. This property can facilitate human-robot communication, as we only need to provide natural language grounding to robots, and they can self-evolve to take the best advantage of this grounding without developing human undecipherable protocols.

5.4 Global Mapping and Local Selection

The core of pragmatics is the consideration of the context while comprehending the language. In this experiment, we want to show that the teacher using a pragmatic protocol can learn global mappings from instances to messages and select messages dynamically according to the context. First, we test global mapping by checking if messages used by the teacher for a target are consistent with this
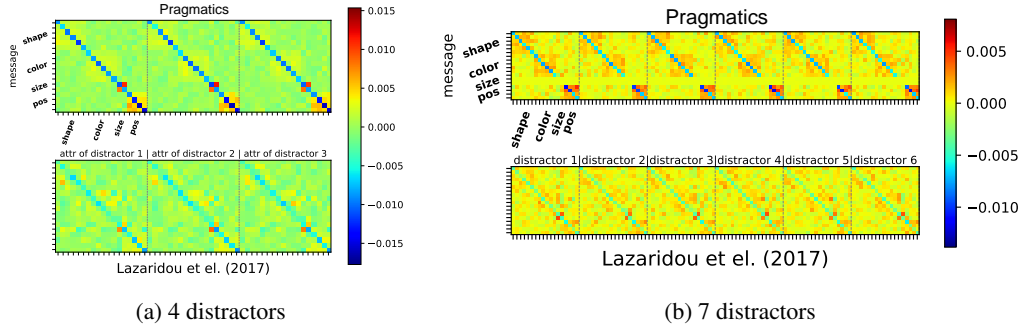
Figure 4: Covariance between messages and distractor attributes given the same target but different distractors. In both experiments, our algorithm has a more clear pattern. The blue diagonals show that if an attribute appears among distractors the teacher tends to avoid the corresponding message. The brighter sub-squares illustrate that messages are significantly influenced by distractors' attributes in the same category. Namely, color messages have stronger covariance with color attributes than with shapes, sizes or positions. Notice that size messages are seldom used in 7-distractor game, thus their covariance with distractors is close to 0. Better be viewed in high resolution, e.g. using Adobe Acrobat Reader.

target's attributes. Then we evaluate message selection through seeing whether the same target yields different messages given different distractors. To make the message grounding easier to understand, we still pretrain agents with Bayesian belief update. We took 80% of the instances to generate training data. In the testing phase, the rest 20% of images are used as targets with 3/6 distractors randomly selected from all images. Results in table 3 justify that the pragmatic protocols achieve the best balance between message validity and referential accuracy. Then we calculated covariance between messages and distractors attributes given the same target but changing distractors. For a target, the covariance is calculated using 100 games. In figure 4, we visualized mean covariance for 58 targets. Since the teacher doesn't have access to distractors in [5], we only compared our model and [24].

6 Conclusion

In this paper we propose an end-to-end trainable adaptive training algorithm integrating ToM in multi-agent communication protocol emergence. The pragmatic protocol developed by our algorithm yields significant performance gain over non-pragmatic protocols. With human knowledge pre-grounding, the teaching complexity using the pragmatic protocol approximates RTD. Our algorithm incorporates a global mapping from instances to messages with a local message selection mechanism sensitive to the context. In future research, we hope to generalize the referential game to a new communicative learning framework, where students, instead of learning from data or an oracle, learn from a helpful teacher with ToM. We also plan to apply our algorithm to more generic communication settings, where agents have more symmetric roles. Namely, we have agents each holding some information unknown to the group and need communication to accomplish a task. We also want to relax the need of exchanging beliefs directly in the training phase, replacing it with discrete feedback requiring smaller channel bandwidth.

References

[1] M. H. Christiansen and S. Kirby, "Language evolution: Consensus and controversies," *Trends in cognitive sciences*, vol. 7, no. 7, 2003.

[2] R. Ibsen-Jensen, J. Tkadlec, K. Chatterjee, and M. A. Nowak, "Language acquisition with communication between learners," *Journal of The Royal Society Interface*, vol. 15, no. 140, p. 20180073, 2018.

[3] C. V. Goldman, M. Allen, and S. Zilberstein, "Learning to communicate in a decentralized environment," *Autonomous agents and multi-agent systems*, vol. 15, no. 1, pp. 47–90, 2007.

[4] J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Advances in Neural Information Processing Systems*, 2016.

[5] A. Lazaridou, K. M. Hermann, K. Tuyls, and S. Clark, "Emergence of linguistic communication from referential games with symbolic and pixel input," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=HJGv1Z-AW

[6] H. P. Grice, P. Cole, J. Morgan *et al.*, "Logic and conversation," *1975*, pp. 41–58, 1975.

[7] A. Vogel, M. Bodoia, C. Potts, and D. Jurafsky, "Emergence of gricean maxims from multi-agent decision theory," in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2013, pp. 1072–1081.

[8] N. D. Goodman and M. C. Frank, "Pragmatic language interpretation as probabilistic inference," *Trends in cognitive sciences*, vol. 20, no. 11, pp. 818–829, 2016.

[9] G. Jäger, "Game theory in semantics and pragmatics," *Semantics: An international handbook of natural language meaning*, vol. 3, pp. 2487–2425, 2012.

[10] M. Kinney and C. Tsatsoulis, "Learning communication strategies in multiagent systems," *Applied intelligence*, vol. 9, no. 1, pp. 71–91, 1998.

[11] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of markov decision processes," *Mathematics of operations research*, vol. 27, no. 4, pp. 819–840, 2002.

[12] S. Sukhbaatar, R. Fergus *et al.*, "Learning multiagent communication with backpropagation," in *Advances in Neural Information Processing Systems*, 2016, pp. 2244–2252.

[13] D. Premack and G. Woodruff, "Does the chimpanzee have a theory of mind?" *Behavioral and brain sciences*, vol. 1, no. 4, pp. 515–526, 1978.

[14] W. Yoshida, R. J. Dolan, and K. J. Friston, "Game theory of mind," *PLoS computational biology*, vol. 4, no. 12, p. e1000254, 2008.

[15] C. L. Baker, J. Jara-Ettinger, R. Saxe, and J. B. Tenenbaum, "Rational quantitative attribution of beliefs, desires and percepts in human mentalizing," *Nature Human Behaviour*, vol. 1, no. 4, p. 0064, 2017.

[16] H. De Weerd, R. Verbrugge, and B. Verheij, "Higher-order theory of mind in the tacit communication game," *Biologically Inspired Cognitive Architectures*, vol. 11, pp. 10–21, 2015.

[17] A. Das, S. Kottur, J. M. Moura, S. Lee, and D. Batra, "Learning cooperative visual dialog agents with deep reinforcement learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2951–2960.

[18] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[19] P. Doshi and P. J. Gmytrasiewicz, "Monte carlo sampling methods for approximating interactive pomdps," *Journal of Artificial Intelligence Research*, vol. 34, pp. 297–337, 2009.

[20] Y. Han and P. Gmytrasiewicz, "Learning others' intentional models in multi-agent settings using interactive pomdps," in *Advances in Neural Information Processing Systems*, 2018, pp. 5634–5642.

[21] H. De Weerd, R. Verbrugge, and B. Verheij, "Theory of mind in the mod game: An agent-based model of strategic reasoning." in *ECSI*, 2014, pp. 128–136.

[22] H. de Weerd, D. Diepgrond, and R. Verbrugge, "Estimating the use of higher-order theory of mind using computational agents," *The BE Journal of Theoretical Economics*, vol. 18, no. 2, 2017.

[23] C. Claus and C. Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," *AAAI/IAAI*, vol. 1998, 1998.

[24] A. Lazaridou, A. Peysakhovich, and M. Baroni, "Multi-agent cooperation and the emergence of (natural) language," in *International Conference on Learning Representations*, 2017. [Online]. Available: https://openreview.net/forum?id=Hk8N3Sclg

[25] K. Cao, A. Lazaridou, M. Lanctot, J. Z. Leibo, K. Tuyls, and S. Clark, "Emergent communication through negotiation," *arXiv preprint arXiv:1804.03980*, 2018.

[26] T. Doliwa, G. Fan, H. U. Simon, and S. Zilles, "Recursive teaching dimension, vc-dimension and sample compression," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3107–3131, 2014.

[27] X. Chen, Y. Cheng, and B. Tang, "On the recursive teaching dimension of vc classes," in *Advances in Neural Information Processing Systems*, 2016, pp. 2164–2171.

[28] R. Blutner, "Some aspects of optimality in natural language interpretation," *Journal of semantics*, vol. 17, no. 3, pp. 189–216, 2000.

[29] D. Golland, P. Liang, and D. Klein, "A game-theoretic approach to generating spatial descriptions," in *Proceedings of the 2010 conference on empirical methods in natural language processing*. Association for Computational Linguistics, 2010, pp. 410–419.

[30] M. C. Frank and N. D. Goodman, "Predicting pragmatic reasoning in language games," *Science*, vol. 336, no. 6084, pp. 998–998, 2012.

[31] P. Shafto, N. D. Goodman, and T. L. Griffiths, "A rational account of pedagogical reasoning: Teaching by, and learning from, examples," *Cognitive psychology*, vol. 71, pp. 55–89, 2014.

[32] J. Andreas and D. Klein, "Reasoning about pragmatics with neural listeners and speakers," *arXiv preprint arXiv:1604.00562*, 2016.

[33] M. Frank, N. Goodman, P. Lai, and J. Tenenbaum, "Informative communication in word production and word learning," in *Proceedings of the annual meeting of the cognitive science society*, vol. 31, no. 31, 2009.

[34] F. Khani, N. D. Goodman, and P. Liang, "Planning, inference and pragmatics in sequential language games," *Transactions of the Association for Computational Linguistics*, vol. 6, pp. 543–555, 2018.

[35] A. Vogel, A. Gomez Emilsson, M. C. Frank, D. Jurafsky, and C. Potts, "Learning to reason pragmatically with cognitive limitations," in *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 36, no. 36, 2014.

[36] B. Bogin, M. Geva, and J. Berant, "Emergence of communication in an interactive world with consistent speakers," *arXiv preprint arXiv:1809.00549*, 2018.

[37] E. Choi, A. Lazaridou, and N. de Freitas, "Compositional obverter communication learning from raw visual input," *arXiv preprint arXiv:1804.02341*, 2018.

[38] K. Evtimova, A. Drozdov, D. Kiela, and K. Cho, "Emergent communication in a multi-modal, multi-step referential game," *arXiv preprint arXiv:1705.10369*, 2017.

[39] C. V. Goldman and S. Zilberstein, "Optimizing information exchange in cooperative multi-agent systems," in *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*. ACM, 2003, pp. 137–144.

[40] R. Lowe, Y. Wu, A. Tamar, J. Harb, O. P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems*, 2017, pp. 6379–6390.

[41] J. Foerster, R. Y. Chen, M. Al-Shedivat, S. Whiteson, P. Abbeel, and I. Mordatch, "Learning with opponent-learning awareness," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 122–130.

[42] J. N. Foerster, F. Song, E. Hughes, N. Burch, I. Dunning, S. Whiteson, M. Botvinick, and M. Bowling, "Bayesian action decoder for deep multi-agent reinforcement learning," *arXiv preprint arXiv:1811.01458*, 2018.

[43] J. F. Fisac, M. A. Gates, J. B. Hamrick, C. Liu, D. Hadfield-Menell, M. Palaniappan, D. Malik, S. S. Sastry, T. L. Griffiths, and A. D. Dragan, "Pragmatic-pedagogic value alignment," *arXiv preprint arXiv:1707.06354*, 2017.

[44] C. J. C. H. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, Cambridge, 1989.

[45] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[46] L. Bergen, N. Goodman, and R. Levy, "That's what she (could have) said: How alternative utterances affect language use," in *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 34, no. 34, 2012.
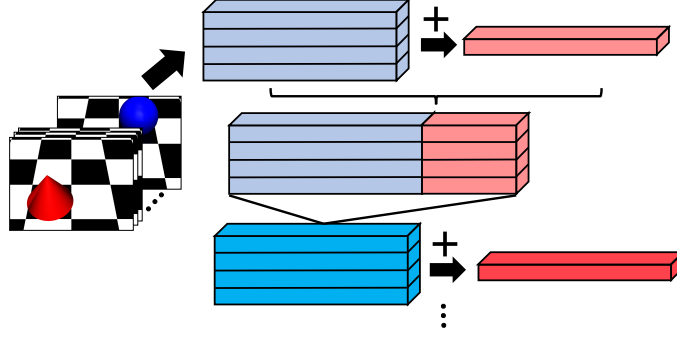
Figure 5: Candidates encoder.

Appendix A   Recursive Teaching Dimension

In computational learning theory, the teaching dimension (TD) measures the minimum number of examples required to uniquely identify all concepts within a concept class. We draw a comparison between identifying a concept using examples and identifying the target using messages. Formally, given an instance space $X = \{1, ..., n\}$ and a concept class $\mathcal{C} \subseteq \{0, 1\}^n$, a teaching set for a concept $c \in \mathcal{C}$ with respect to $\mathcal{C}$ is a subset $S \subseteq X$ such that $\forall c' \neq c \in \mathcal{C}$ $\exists x \in S$ s.t. $c(x) \neq c'(x)$. Intuitively, the teaching set is a subset of instance space that can uniquely identify c in $\mathcal{C}$. TD of c in $\mathcal{C}$, denoted by $TD(c, \mathcal{C})$, is the size of the smallest teaching sets. TD of the whole concept class is $TD(\mathcal{C}) = \max_{c \in \mathcal{C}} TD(c, \mathcal{C})$. However, this definition of TD often overestimates the number of examples needed for cooperative agents, who teach and learn using "helpful" examples. For instance, the game in figure 1a in the main text has TD = 2, but all concepts can be taught with one message.

The recursive teaching dimension (RTD), a variation of the classic TD, can model the behavior between cooperative agents. Define the teaching hierarchy for $\mathcal{C}$ as the sequence $((\mathcal{C}_1, d_1), ..., (\mathcal{C}_h, d_h))$ such that for all $j \in \{1, ..., h\}$, $\mathcal{C}_j = \{c \in \overline{\mathcal{C}}_j | d_j = TD(c, \overline{\mathcal{C}}_j) \leq TD(c', \overline{\mathcal{C}}_j), \forall c' \in \overline{\mathcal{C}}_j\}$ where $\overline{\mathcal{C}}_1 = \mathcal{C}$ and for all $i \in \{1, ..., h - 1\}$, $\overline{\mathcal{C}}_{i+1} = \mathcal{C} \backslash (\mathcal{C}_1 \bigcup ... \bigcup \mathcal{C}_i)$. RTD of c in $\mathcal{C}$ is defined as $RTD(c, \mathcal{C}) = d_j$. RTD of $\mathcal{C}$ is $RTD(\mathcal{C}) = \max\{d_j | 1 \leq j \leq h\}$. Intuitively, one can first learn simplest concepts, $\mathcal{C}_i$, and remove them from the concept class. Then, for the remaining set of concepts, $\overline{\mathcal{C}}_{i+1}$, recursively learn the simplest concepts and so on. RTD lower bounds the classic teaching dimension, i.e. $RTD(\mathcal{C}) \leq TD(\mathcal{C})$ [26]. Also, $RTD(\mathcal{C})$ measures the worst-case number of labeled examples needed to learn any target concept in $\mathcal{C}$, and the teaching hierarchy can be derived by the teacher and student separately without any communication [27].

Appendix B   Network Structure

Our network consists of 3 modules: a belief update network, teacher's Q-net and student's policy network. The teacher and student share the same structure for the belief update network but initialized differently. The output of the belief update network will feed into the Q-net or policy network for message and action selection. Figure 1b shows the structure of our models. When dealing with image inputs, we have an additional perceptual module to process the visual inputs. We use a convolutional neural network to extract features from images.

We align candidates embedding (multi-hot vectors or dense features) into a $1 \times |O| \times D$ tensor and apply 1×1 convolution to every candidate, where D is the candidate embedding dimension. We sum the candidates embedding as the context embedding and concatenate it after each candidate's embedding, followed by another 1×1 convolution. This structure can be repeated as needed. Figure 5 shows the structure. The final embedding of the candidates forms a tensor with shape $1 \times |O| \times D_M$, where D_M is the dimension of the message encoding. Then we do another 1×1 convolution with the message encoding of m as the only tensor and get $|O|$ numbers, which is then fed into a sigmoid layer and returns the likelihood $P(m|o)$ for all $o \in O$. This likelihood then multiplies with the input prior and returns posterior $P(o|m)$ after normalization. Teacher's Q-net reuses candidates last embedding layer. The $1 \times |O| \times D_M$ tensor is weighted by the return value of teacher's belief update function and teacher's ground truth belief (one-hot vector in referential games). We then sum the weighted

tensors with respect to the second dimension and get two vectors. The concatenation of these vectors is passed into a fully connected layer and outputs a real number as the Q-value.

Student's policy network is relatively simple. We first pass in the output of student's belief update network to a fully connected layer followed by a sigmoid function, outputting the probability of waiting. Then action is sampled directly from the new belief. If wait, this action will be discarded, otherwise, it becomes the student's prediction of the target.