Multi-Agent Cognition Difference Reinforcement Learning for Multi-Agent Cooperation

1st Huimu Wang
School of Artificial Intelligence
University of Chinese Academy of Sciences
Institute of Automation
Chinese Academy of Sciences
Beijing, China
wanghuimu2018@ia.ac.cn

2nd Tenghai Qiu
Institute of Automation
Chinese Academy of Sciences
Beijing, China
tenghai.qiu@ia.ac.cn

3nd Zhen Liu
Institute of Automation
Chinese Academy of Sciences
Beijing, China
liuzhen@ia.ac.cn

4rd Zhiqiang Pu
Institute of Automation
Chinese Academy of Sciences
University of Chinese Academy of Sciences
Beijing, China
zhiqiang.pu@ia.ac.cn

5th Jianqiang Yi
Institute of Automation
Chinese Academy of Sciences
University of Chinese Academy of Sciences
Beijing, China
jianqiang.yi@ia.ac.cn

6th Wanmai Yuan
China Academy of Electronics and Information Technology
Beijing, China
yuanwanmai@gmail.com

***Abstract*—Multi-agent cooperation is one of the most attractive research fields in multi-agent systems. There are many attempts made by researchers in this field to promote the cooperation behavior. However, in partially-observable environments, a large number of agents and complex interactions among the agents cause huge difficulty for policy learning. Moreover, redundant communication contents caused by many agents make effective features hard to be extracted, which prevents the policy from converging. To address the limitations above, a novel method called multi-agent cognition difference reinforcement learning (MACD-RL) is proposed in this paper. The key feature of MACD-RL lies in cognition difference network (CDN) and a soft communication network (SCN). CDN is designed to allow each agent to choose its neighbors (communication targets) adaptively with its environment cognition difference. SCN is designed to handle the complex interactions among the agents with soft attention mechanism. The results of simulations including mixed cooperative and competitive tasks demonstrate that the effectiveness and robustness of the proposed model.**

I. Introduction

Effective communication is a key ability for multi-agent collaboration. Agents (human or artificial) in the real world exchange information with each other, which enables them to coordinate with each other and formulate strategies to implement complex cooperative behaviors. Similarly, communication is very important in multi-agent reinforcement learning for cooperation, especially in scenarios that require cooperation, such as smart grid control [1], resource management [2], and games [3], [4].

Recently, deep reinforcement learning (DRL) has made exciting progress in many domains, such as games [5], [6] and robotics [7], [8]. Owing to the huge potential indicated by these deep learning based approaches, the combination of deep learning and multi-agent reinforcement learning has also been widely studied [9]–[11]. However, when these algorithms are applied to environments with a large number of agents, there exists some limitations. Multi-agent deep deterministic policy gradient (MADDPG) [9] and counterfactual multi-agent policy gradients (COMA) [10] follow a common paradigm of centralized learning with decentralized execution (CTDE) to promote the cooperation behavior among the agents. These methods do not consider the communication relationship between agents, which limits the cooperation of agents.

To deal with the problem of communication, some methods are proposed [12]–[14]. [12] proposes a large feed-forward neural network that maps inputs of all agents to their actions. Each agent is controlled by the network, which additionally has access to a communication channel to receive the summed information of other agents. Bidirectionally-coordinated network (BiCNet) [14] is based on actor-critic model for continuous action. It adopts bidirectional recurrent networks to achieve mutual communication between agents. Master-Slave [13] is a communication architecture for real-time strategy games, where the action of each slave agent consists of information from the slave agents and master agent. However, the methods above assume that each agent knows the global states of the environment, which is not applicable in partially observable environments. Moreover, those methods do not filter the agents' communication, so redundant communication

This work is supported by the National Key Research and Development Program of China under Grant 2018AAA0102404, in part by Innovation Academy for Light-duty Gas Turbine, Chinese Academy of Sciences, No.CXYJJ19-ZD-02 and No.CXYJJ20-QN-05

contents caused by a large number of agents will affect the agents' policy.

To address this issue, some methods based on attention mechanism are proposed [15]–[18]. Nonetheless, for those methods with attention mechanism, all agents within the communication range will be assigned an attention coefficient. The total amount of communication is not decreased since irrelevant agents can still obtain the attention coefficient and be included in the total amount of communication. Besides, the soft attention mechanism usually assigns small but nonzero attention coefficients to irrelevant agents, which weakens the attention assigned to the significant agents in disguise.

Motivated by the above issues, we propose a multi-agent cognition difference reinforcement learning (MACD-RL) to filter communication to promote cooperation behavior of agents. The key feature of MACD-RL lies in cognition difference network (CDN) and soft communication netowrk (SCN). Specifically, the high-dimensional representation of the agents' understanding of the environment is extracted with posterior distribution in CDN. Then unrelated agents are filtered out with Kullback-Leibler (KL) divergence [19] based on the difference between the distributions. SCN is responsible for the weight distribution of the filtered agents to handle the influence of different neighbors. Besides, SCN expands communication range of the agents with the chain propagation characteristics of graph neural networks (GNN).

The effectiveness of MACD-RL is evaluated in different environments including cooperative navigation and predator-prey games. The simulation results demonstrate that our methods can enable agents to learn stable and complicated cooperative strategies in large-scale environments.

II. Background

A. Partially Observable Markov Games

The problem of paper is considered as partially observable Markov Games (POMG) that is an extension of the Markov Games [20]. It is defined by environment state S^t, action spaces A^t= $[a_1^t, \cdots, a_N^t]$ where a_i^t is the action F_i^t of agent i at time t, and observation spaces $O^t = [o_1^t, \ldots, o_N^t]$. Each agent i learns a policy $\pi : o_i^t \rightarrow P_a\left(a_i^t\right)$ which maps each agent's observation to a distribution over it's set of actions. Then the next states are produced according to the transition function $T : S^t \times a_1^t \times \cdots \times a_N^t \rightarrow P_t(S^{'})$. Each agent i obtains rewards R_i as a function of the state space and action spaces: $S^t \times a_1^t \times \ldots \times a_N^t \rightarrow \mathbb{R}$. The goal of the agents is to maximize their expected discounted returns with a policy:

$$J_i(\pi_i) = E_{a_1 \sim \pi_1, \ldots, a_N \sim \pi_N, s \sim T}\left[\sum_{t=0}^{\infty} \gamma^t r_i^t\left(S^t, a_1^t, \ldots, a_N^t\right)\right] \quad (1)$$

where r_i^t represents the reward that agent i obtains at time t and $\gamma \in [0, 1)$ denotes the discount factor determining the importance of future rewards.

B. PPO

Reinforcement learning (RL) [21] is adopted to solve special POMG problems where $N = 1$. It is a machine learning approach to solve sequential decision-making problems. Policy gradient methods are the popular choice for a variety of RL tasks. The policy π is parameterized as a policy π_θ through the parameter θ of a neural network. Its objective is to directly adjust the parameters θ of the policy in order to maximize the objective $J\left(\pi_\theta\right) = E_{a \sim \pi_\theta, S \sim T}\left[\sum_{t=0}^{\infty} \gamma^t r_i^t\left(S^t, a_1^t, \ldots, a_N^t\right)\right]$ by taking steps in the direction of $\nabla_\theta J\left(\pi_\theta\right)$:

$$\nabla_\theta J(\pi_\theta) = \nabla_\theta \log\left(\pi_\theta\left(a_t \middle| S_t\right)\right) \sum_{t'=t}^{\infty} \gamma^{t'-t} r_i^{t'}\left(s_i^{t'}, a_i^{t'}\right) \quad (2)$$

The proximal policy optimization algorithm (PPO) [22] is a novel policy graditent method based on the actor-critic framework for RL. We adopt PPO as the basic training algorithm in this paper. PPO applies DNNs to approximate the actor ($\pi_\theta(a_t \,|s_t)$) and the critic ($Q_w\left(s_t, a_t\right)$),respectively. And they are trained as follows:

$$l_t(\theta) = \frac{\pi_\theta(a_t \,|s_t)}{\pi_{\theta^k}(a_t \,|s_t)} \quad (3)$$

$l_t(\theta)$ denotes the likelihood ratio. π_{θ^k} denotes the policy of the agent before k steps. Then the objective function is optimized according to the following equations:

$$L_v\left(w\right) = E\left[\left(Q\left(s_t, a_t\right) - Q_w\left(s_t, a_t\right)\right)^2\right] \quad (4)$$

$$\begin{aligned} L_\pi(\theta) = E[&\min(l_t(\theta)\hat{A}_t^{\theta^k}(s_t, a_t), \\ &clip(l_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t^{\theta^k}(s_t, a_t)] \end{aligned} \quad (5)$$

where $\hat{A}_t^{\theta^k}(s_t, a_t)$ is the generalized advantage estimate (GAE) and $clip\left(l_t\left(\theta\right), 1-\varepsilon, 1+\varepsilon\right)$ clips $l_t\left(\theta\right)$ in the interval $[1-\varepsilon, 1+\varepsilon]$.

III. Method

In the environments with large number of agents, there are many neighbors within the communication range of the agents. If an agent communicates with all its neighbors, redundant information will affect the agent's decision. When the number of agents in the environment increases, the communication among agents is more complicated, which makes the influence of redundant information more serious.

The common methods are to use artificial rules [11] or soft attention mechanism [15]–[18] to process communication among agents. For the methods with artificial rules, they require strong prior knowledge of the environment and may not be suitable for application in complex environments. For the methods with soft attention mechanism, they assign attention coefficients to all agents within the communication range. The total amount of communication is not decreased since irrelevant agents can still obtain the attention coefficient and be included in the total amount of communication. Besides, the soft attention mechanism usually assigns small but nonzero attention coefficients to irrelevant agents, which essentially weakens the attention assigned to the significant agents. Therefore, we propose a multi-agent cognition difference reinforcement learning composed of cognition difference network (CDN) and soft communication netowrk (SCN). CDN is responsible for filtering redundant agents to reduce

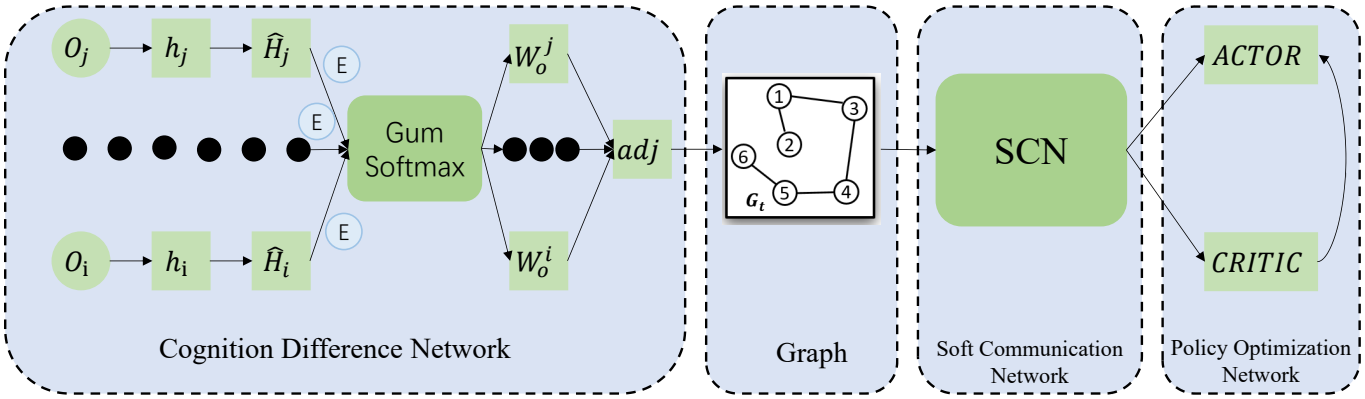


Fig. 1. Architecture of MACD-RL

redundant information. SCN is responsible for the weight distribution of the filtered agents to handle the influence of different neighbors. The details are presented in the following parts.

A. Cognition Difference Network

We construct the communication status between agents as a graph, where each node represents a single agent, and all nodes are connected in pairs by default.

***Definition** 1:* The communication status between the agents is defined as $G^t = (V, E^t)$. In particular, $V = \{1, \ldots, N\}$ is a set of the agents. $E^t \subseteq V \times V$ denotes the edge set at time t. Besides, h^t is a set of node features at time t, $h^t = \left\{\vec{h}_1^t, \vec{h}_2^t, ..., \vec{h}_N^t\right\}$, $\vec{h}_i^t \in \mathbb{R}^L$, where L represents the feature dimension of each node.

In order to filter neighbor agents more effectively, we introduce the definition of cognitive difference based on an assumption.

Definition** 2:* The ***cognition of an agent is its understanding of the local environment. It contains the states of all entities within its observation range, or the high-level hidden states or distributions extracted from these states (e.g., learned with DNNs).

Definition** 3:* The ***cognition difference is the difference of high-level hidden states or distributions between the agents measured by n-dimensional norm or distribution measures.

***Assumption** 1:* Cognition of each agent can be represented by a vector $C = [c_1, \ldots, c_k]$ or a distribution $p(C_i|o_i)$ obeying Gaussian distribution.

Under the above definition and assumptions, if the cognition difference between agent i and agent j are relatively large, it means that agents' perceptions of the environment are quite different. In other words, the states of agent j is a noise or disturbance to agent i, which will affect the policy of agent i, so agent i does not need to communicate with agent j.

After analyzing the role of the cognition difference, we need to solve two problems including the representation of the cognition and the measurement of the cognition difference.

The representation of the cognition of agent i is denoted as cognition vector C_i, it is based on the states of all entities within its observation range. The commom methods are to adopt directly multi-layer perception (MLP) [9]–[11] or graph convolution network (GCN) [18], [23], [24] to extract features of the observations as the cognition vector C_i. However, the cognitive vectors extracted by these methods are essentially the result of single-valued mapping from vector to vector. Besides, these methods cannot fully decouple the factors of the observations, such as position or velocity. Therefore, the cognition vector extracted by MLP or GCN is not appropriate as the cognition of the environment.

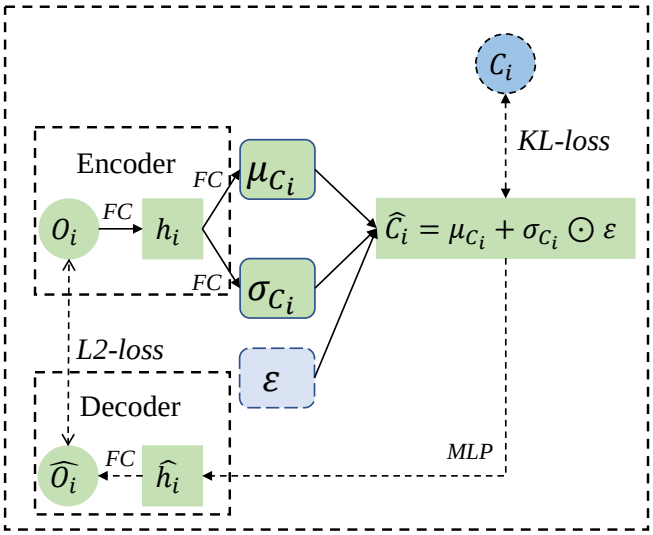


Fig. 2. Architecture of Variational Autoencode. Note that FC denotes fully-connected layers

In order to effectively represent the cognitive vector, we adopt a probability distribution method. Specifically, the cognition vector C_i is extracted with posterior distribution from observation states o_i and C_i is inferred with:

$$p(C_i|o_i) = \frac{p(o_i|C_i)\,p(C_i)}{p(o_i)} = \frac{p(o_i|C_i)\,p(C_i)}{\int p(o_i|C_i)\,p(C_i)\,dC_i} \quad (6)$$

where $p(o_i|C_i)$ is a reconstruction process from the cognition vector C_i. $p(C_i|o_i)$ is the cogniton representation for agent

i. However, the above equation is diffcult to calculate directly. Hence, we approximate $p\left(C_i \left| o_i \right.\right)$ by another tractable distribution $q\left(C_i \left| o_i \right.\right)$. The objective is that $q\left(C_i \left| o_i \right.\right)$ needs to be close to $p\left(C_i \left| o_i \right.\right)$. We achieve it by minimizing the following KL divergence:

$$\min KL\left(q\left(C_i \left| o_i \right.\right) \| p\left(C_i \left| o_i \right.\right)\right) \tag{7}$$

which equals to maximize evidence lower bound (ELBO) [25]:

$$\max E_{q\left(C_i \left| o_i \right.\right)} \log p\left(C_i \left| o_i \right.\right) - KL\left(q\left(C_i \left| o_i \right.\right) \| p\left(C_i\right)\right) \tag{8}$$

In the above equation, the former represents the reconstruction likelihood, and the latter ensures that $q\left(C_i \left| o_i \right.\right)$ is similar to the true prior distribution $p\left(C_i\right)$. This can be modelled by a variational autoencoder (VAE) [26] as shown in Fig. 2. The encoder of this VAE learns a mapping $q\left(\widehat{C_i} \left| o_i \right. ; \chi\right)$ from o_i to $\widehat{C_i}$ where χ are the parameters of DNNs. Next, we adopt the "reparameterization trick" to sample ε from a unit Gaussian, and then generate $\widehat{C_i}$ with mean μ_{C_i} and variance σ_{C_i}: $\widehat{C_i} = \mu_{C_i} + \sigma_{C_i} \odot \varepsilon$ where $\varepsilon \sim N\left(0, 1\right)$. The decoder of this VAE learns a mapping $p\left(\widehat{o_i} \left| \widehat{C_i}; \chi\right.\right)$ from $\widehat{C_i}$ back to $\widehat{o_i}$. The loss function to train this VAE is:

$$\min L2\left(o_i, \widehat{o_i}; \chi\right) + KL\left(q\left(\widehat{C_i} \left| o_i; \chi\right.\right) \| p\left(C\right)\right) \tag{9}$$

where $\widehat{C_i}$ is the result of the cognition representation of agent i and the learned distribution $q\left(\widehat{C_i} \left| o_i \right. ; \chi\right)$ is an approximation of $p\left(C_i \left| o_i \right.\right)$.

For the measurement of difference, since $\widehat{C_i}$ is sampled from the learned distribution, the distribution is more general in the cognition of the environment than $\widehat{C_i}$. Therefore, we choose the difference between the distributions as the cognition difference. To calculate the cognition difference between agent i and agent j, KL divergence is adopted to calculate a score C_d^{ij} that measures the difference.

$$C_d^{ij} = KL\left(q\left(\widehat{C_i} \left| o_i \right. ; \chi\right) \middle\| q\left(\widehat{C_j} \left| o_j \right. ; \chi\right)\right) \tag{10}$$

The cognition difference model is needed to output a one-hot vector to determine whether the edge between node i and j exist in the graph G^t or which agents need to be communicated. However, the process of outputting the one-hot vector is often unable to achieve back-propagation of gradients due to the sampling process. Therefore, Gumbel-Softmax function [27] is adopted to solve it:

$$W_c^{i,j} = gum(f\left(KL\left(q\left(\widehat{C_i} \left| o_i \right. ; \chi\right) \middle\| q\left(\widehat{C_j} \left| o_j \right. ; \chi\right)\right)\right)) \tag{11}$$

where $gum\left(\cdot\right)$ reresents the Gumbel-Softmax function.

By cognition difference model, we can get a sub-graph $G_i^{'}$ for agent i, where agent i just connected with the agents that need to communicate.

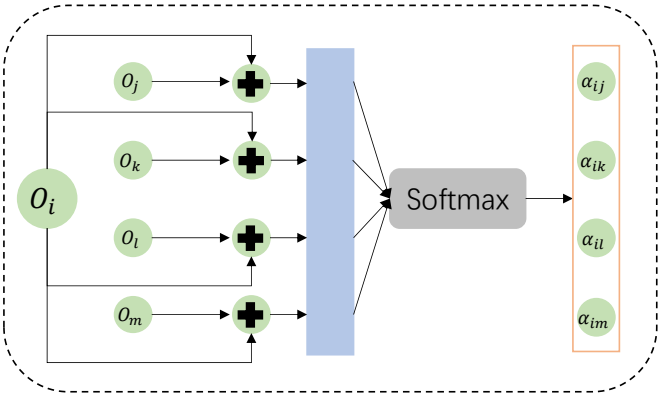


Fig. 3. Architecture of attention

B. Soft Communication Network

After getting the sub-graph $G_i^{'}$, the filtered neighbour agents need to be treated differently by agent i for promoting cooperation. Different neighbors have different cognition of the environment, and this will have different influence on agent i. Specifically, one of the neighbour agents may be farther away from agent i than the other agents, which means agent i will be influenced differently by cognition difference caused by different distances between agents. Therefore, the soft attention mechanism is adopted in SCN to enable agent i to process the other agents' states differently. SCN operates on graph-structured data and obtains the features of each graph node by aggregating the states of its neighbors. As shown in Fig. 3, the attention coefficient e_{ij} from agent j to agent i and its normalized form α_{ij} are calculated with the hidden states of the agents:

$$e_{ij} = a_G^k\left(W_G^k \widehat{C_i}, W_G^k \widehat{C_i}\right) \tag{12}$$

$$\alpha_{ij} = \mathrm{softmax}(e_{ij}) = \frac{\exp(\mathrm{LeakyReLU}(e_{ij}))}{\sum_{k \in N_i} \exp(\mathrm{LeakyReLU}(e_{ij}))} \tag{13}$$

where W_G^k is a linear learnable matrix, a_G^k is a single-layer feed-forward neural network and $LeakyReLU$ is a nonlinear activation function. [28] shows that the import of multi-head attention is helpful to stabilize the learning process of the attention coefficients. Therefore, the aggregation states of agent i with multi-head attention at t is given by:

$$h_i^{t^{'}} = \|_{m=1}^{K} \sigma\left(\sum\nolimits_{j \in N_i} \alpha_{ij}^m W^m h_i^t\right) \tag{14}$$

where N_i denotes the filtered neighbor agents for agent i. $||$ represents the concatenation, K represents the number of the heads, α_{ij}^m represents the normalized attention coefficient of the m-th attention mechanism and W^m represents the weight matrix of the m-th linear transformation.

Furthermore, K-hop communication is used to enlarge the receptive field of agent i. With SCN, the agent can know the cognition of the neighbors and enlarge the communication field, which is beneficial to the cooperation with other agents.

C. Training Method

After states are extracted by SCN, they are utilized to optimize the policy of agents. PPO is adopted to train the agents based on an actor-critic framework. Owing to CDN and SCN, each agent can choose the agents to communicate with and assign different weights to them. According to the objective function of PPO as shown in (5), it is changed as (17) after the state extracted from CDN and SCN. Then PPO is trained by minimizing an total loss L_{total}, which is conducted by the weighted summation of value loss $L_V(w)$, action loss $L_\pi(\theta)$, action entropy $H(\theta)$ and cognition difference loss $L_{cd}(\chi)$:

$$L_V(w) = E\left[\begin{pmatrix} Q(O_1, O_2, \cdots, O_N, a) \\ -Q(O_1, O_2, \cdots, O_N, a; w) \end{pmatrix}^2\right] \quad (15)$$

$$l_t(\theta) = \frac{\pi_\theta(a_t \,|(O_1, O_2, \cdots, O_N))}{\pi_{\theta^k}(a_t \,|(O_1, O_2, \cdots, O_N))} \quad (16)$$

$$\begin{aligned} L_\pi(\theta) = E[\min(&l_t(\theta)\hat{A}_t^{\theta^k}(O_1, O_2, \cdots, O_N), \\ &clip(l_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t^{\theta^k}(O_1, O_2, \cdots, O_N)] \end{aligned} \quad (17)$$

$$\begin{aligned} H(\theta) = -\sum &\pi_\theta(a_t \,|O_1, O_2, \cdots, O_N) \\ &\log(\pi_\theta(a_t \,|O_1, O_2, \cdots, O_N)) \end{aligned} \quad (18)$$

$$L_{cd}(\chi) = E\left[\left(L2(o_i, \widehat{o_i}; \chi) + KL\left(q\left(\widehat{C_i} \,|o_i; \chi\right) \| p(C)\right)\right)\right] \quad (19)$$

$$L_{total} = \beta_1 L_V(w) + \beta_2 L_\pi(\theta) - \beta_3 H(\theta) + \beta_4 L_{cd}(\chi) \quad (20)$$

where β_i is the weight coefficient of the loss function. The action entropy $H(\theta)$ is specially designed to encourage exploration for agents by penalizing the entropy of actor $\pi_\theta(a_t \,|O_1, O_2, \cdots, O_N)$. The implementation details of PPO are presented in Algorithm 1.

Algorithm 1 PPO

Input: agent's state o_i

Initialization: Initialize actor θ_a, critic θ_c and old actor θ_a^{old} network

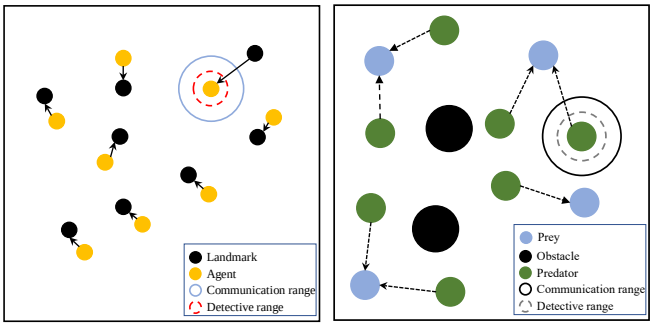
for Episode 1 **to** M **do**
  Run policy $\pi_{\theta^a}(o_1, \ldots, o_N)$ for T time-steps for each agent, collecting $\{o, a, r\}$ where $o = (o_1, \cdots, o_N)$, $a = (a_1, \cdots a_N)$ and $r = (r_1, \cdots, r_N)$
  Estimate advantages function A_t
  $\pi_{\theta_a^{old}} \leftarrow \pi_{\theta^a}$
  for k = 1 **to** 4 **do**
    Calculate value loss: $L_v(w)$
    $L_v(w) = \left(Q - Q^t_{final}(w)\right)^2$
    Calculate action loss: $L_\pi(\theta)$
    $L_\pi(\theta) = E[\min(l_t(\theta)\hat{A}_t^w(x, a_1, \ldots, a_N), clip(l_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t^w(x_{t,1}, \ldots, a_N)]$
    Calculate entropy loss: $H(\theta)$
    $H(\theta) = -\sum \pi_\theta \log(\pi_\theta)$
    Calculate cognition difference loss: $L_{cd}(\chi)$
    $L_{cd}(\chi) = E\left[\left(L2(o_i, \widehat{o_i}; \chi) + KL\left(q\left(\widehat{C_i} \,|o_i; \chi\right) \| p(C)\right)\right)\right]$
    Calculate total loss: L_{total}
    $L_{total} = \beta_1 L_V(w) + \beta_2 L_\pi(\theta) - \beta_3 H(\theta) + \beta_4 L_{cd}(\chi)$
    Update actor and critic network by minimizing L_{total}
  end for
end for

IV. Simulations

A. Simulation Settings

In this section, the performance of MACD-RL is evaluated in two tasks as shown in Fig. 4.

For the cooperative navigation, there are N agents and N landmarks in the environment as shown in Fig. 4 (a). The objective is for the agents to deploy themselves in a manner such that every agent reaches a distinct landmark. Note that we do not assign particular landmark to each agent, but instead let the agents communicate with each other and develop a consensus as to who goes where. A dense reward of mean distance between agents and landmarks is set. It means that the reward for each agent is not only related to it self, but also related to the other agents.



(a) Cooperative navigation (b) Predator-prey games

Fig. 4. The illustration of simulation tasks

For the predator-prey games shown in Fig. 4 (b), the predators move slower and need to capture all the preys. The detection and communication range of the predator is restricted, but the preys are not subject to this restriction. If a prey is caught by one predator, the predator will obtain a positive reward.

These scenarios are implemented based on [9]. The action space is discrete and each agent is able to control unit acceleration or deceleration in X and Y directions. For all the scenarios, the detective range for an agent is set as 1 unit and the communication range is set as 1.5 unit.

As a baseline algorithm for comparing the performance, TRANSFER [18] is taken into consideration. GCN and soft attention mechanism are adopted in TRANSFER, but agent filtering is not taken into consideration.

B. Evaluation Results

The simulation results of 100 independent simulations for each scenarios are presented in Tables I-II. t-test value is

TABLE I
EVALUATION RESULTS OF COOPERATIVE NAVIGATION

Method		N=6			N=15		
		Success rate	Steps	Rewards	Success rate	Steps	Rewards
TRANSFER	mean	100	14.2	-0.52	97	17.18	-0.61
MACD-RL	mean	100	14.17	-0.53	**99**	**15.13**	**-0.56**
Method		N=20			N=29		
		Success rate	Steps	Rewards	Success rate	Steps	Rewards
TRANSFER	mean	98	20.11	-0.72	96	25.2	-0.86
MACD-RL	mean	**99**	**17.05**	**-0.64**	**98**	**20.06**	**-0.69**

TABLE II
EVALUATION RESULTS OF PREDATOR-PREY GAMES

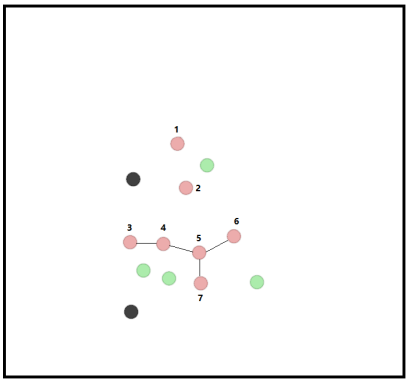
Method		3V1		5V2	
		Steps	Rewards	Steps	Rewards
TRANSFER	mean	22.17	0.47	34.25	0.82
MACD-RL	mean	**20.2**	**0.53**	**31.12**	**1.07**
Method		7V2		7V4	
		Steps	Rewards	Steps	Rewards
TRANSFER	mean	40.13	0.63	48.05	1.45
MACD-RL	mean	**35.26**	**0.85**	**40.11**	**1.73**

adopted to evaluate the effectiveness of our method statistically. According to the mean value, standard deviation and the sample data for 100 tests, we calculate the t value of AERL and the other methods on the same test scenario. "+" and "=" indicate that the index values obtained by the algorithm in this paper are superior and equal to the results of the other methods in the same test scenario in the two-tailed t-test with a significance level of $\%5$.

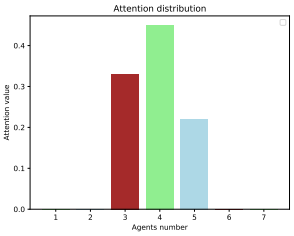
For the cooperative navigation task, as indicated by Table I, our method obtains 6 optimal measurements during all the test scenarios. Note that there is no significant performance difference between our method and TRANSFER in terms of success rate. The reason is that the task is simple so all methods can successfully complete the task. Despite this, our method is better than TRANSFER in terms of rewards and steps in scenarios with large number of agents. When the number of agents is equal to 6, all methods presents a similar performance. When the number of agents increases, our proposed MACD-RL can obtain more rewards and take fewer steps than TRANSFER to complete the task. The reason is that agents need to communicate with more neighbor agents as the number of agents increases, and MACD-RL can filter out redundant agents to promote multi-agent cooperation behavior.

For the predator-prey games, the simulation results of MACD-RL and TRANSFER are presented in Table II. MACD-RL obtain 8 optimal measurements during 4 scenarios and outperforms than TRANSFER in terms of steps and rewards. In this kind of competitive scenarios where there is a high demand for communication effectiveness, CDN in MACD-RL can filter out unrelated agents, and SCN can process the information of filtered agents with attention weights to promote cooperation.

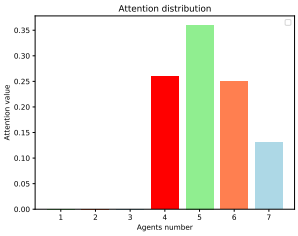
To further demonstrate the effectiveness of our method, we present the attention value distribution of predators in Fig. 5. As shown in Fig. 5, we can obtain the attention value distribution for agent 4 (Fig. 5 (b)) and agent 5 (Fig.5 (c)). Note that agent 4 only communicates with agent 3 and agent 5, and agents 5 communicates with agent 4, agent 6 and agent 7. Therefore, the attention coefficients are only assigned to those agents communicating with agent 4 and agent 5. Moreover, Since different agents have different influences, they are assigned different attention coefficients. It demonstrates that MACD-RL can filter out unrelated agents and process the information of different agents to promote cooperation.



(a) The cooperative strategy in 7V4



(b) Attention distribution for agent 4



(c) Attention distribution for agent 5

Fig. 5. Attention distribution

V. CONCLUSION

In this paper, we present a novel reinforcement learning method MACD-RL for multi-agent cooperation in environments with a large number of agents. With MACD-RL, unrelated agents are filtered out and complex interactions between the filtered agents are handled as well. MACD-RL is shown to perform a satisfying strategy and adapt to environments

with many agents. Future work will take the adaptive grouping based on cognitive differences into consideration.

REFERENCES

[1] Y. Yang, J. Hao, M. Sun, Z. Wang, C. Fan, and G. Strbac, "Recurrent deep multiagent q-learning for autonomous brokers in smart grid." in *IJCAI*, vol. 18, 2018, pp. 569–575.

[2] X. Li, J. Zhang, J. Bian, Y. Tong, and T.-Y. Liu, "A cooperative multi-agent reinforcement learning framework for resource balancing in complex logistics network," in *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2019, pp. 980–988.

[3] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[4] D. Ye, G. Chen, P. Zhao, F. Qiu, B. Yuan, W. Zhang, S. Chen, M. Sun, X. Li, S. Li, and et al., "Supervised learning achieves human-level performance in moba games: A case study of honor of kings," *IEEE Transactions on Neural Networks and Learning Systems*, p. 1–11, 2020. [Online]. Available: http://dx.doi.org/10.1109/TNNLS.2020.3029475

[5] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[6] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, 2016, pp. 1928–1937.

[7] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[8] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," 2016.

[9] 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.

[10] 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.

[11] Y. Yang, R. Luo, M. Li, M. Zhou, W. Zhang, and J. Wang, "Mean field multi-agent reinforcement learning," in *International Conference on Machine Learning*, 2018, pp. 5567–5576.

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

[13] X. Kong, B. Xin, F. Liu, and Y. Wang, "Revisiting the master-slave architecture in multi-agent deep reinforcement learning," 2017.

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

[15] J. Jiang and Z. Lu, "Learning attentional communication for multi-agent cooperation," in *Advances in Neural Information Processing Systems*, 2018, pp. 7254–7264.

[16] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *International Conference on Machine Learning*, 2019, pp. 2961–2970.

[17] A. Das, T. Gervet, J. Romoff, D. Batra, D. Parikh, M. Rabbat, and J. Pineau, "Tarmac: Targeted multi-agent communication," in *International Conference on Machine Learning*, 2019, pp. 1538–1546.

[18] A. Agarwal, S. Kumar, and K. Sycara, "Learning transferable cooperative behavior in multi-agent teams," *arXiv preprint arXiv:1906.01202*, 2019.

[19] S. Kullback, *Information theory and statistics*. Courier Corporation, 1997.

[20] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine learning proceedings 1994*. Elsevier, 1994, pp. 157–163.

[21] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[22] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[23] Y. Liu, W. Wang, Y. Hu, J. Hao, X. Chen, and Y. Gao, "Multi-agent game abstraction via graph attention neural network," 2019.

[24] A. Malysheva, D. Kudenko, and A. Shpilman, "Magnet: Multi-agent graph network for deep multi-agent reinforcement learning," in *2019 XVI International Symposium "Problems of Redundancy in Information and Control Systems" (REDUNDANCY)*, 2019, pp. 171–176.

[25] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, p. 859–877, Apr 2017. [Online]. Available: http://dx.doi.org/10.1080/01621459.2017.1285773

[26] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2014.

[27] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.

[28] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ