Neighborhood-oriented Decentralized Learning Communication in Multi-Agent System*

Hao Dai[1,2][0000−0003−1018−2162], Jiashu Wu[1,2][0000−0002−1347−1974], André Brinkmann[3][0000−0003−3083−2775], and Yang Wang[1,2][0000−0001−9438−6060](✉)

[1] Shenzhen Institute of Advanced Technology, Chinese Academy of Science
[2] University of Chinese Academy of Science
[3] Zentrum für Datenverarbeitung, Johannes Gutenberg-Universität Mainz
yang.wang1@siat.ac.cn

Abstract. Partial observations are one of the critical obstacles in multi-agent systems (MAS). The Centralized Training Decentralized Execution (CTDE) paradigm has been widely studied to integrate global observations into the training process. However, the traditional CTDE paradigm suffers from local observations during the execution phase, and numerous efforts have been made to study the communication efficiency among agents to promote cognitive consistency and better cooperation. Furthermore, training still operates in a centralized manner, requiring agents to communicate with each other in a broadcast fashion. As a consequence, this centralized broadcast-based training process is not feasible when being applied to more complex scenarios. To address this issue, in this paper we propose a neighborhood-based learning approach to enable agents to perform training and execution in a decentralized manner based on information received from their neighboring nodes. In particular, we design a novel encoder network and propose a two-stage decision model to improve the performance of this decentralized training. To evaluate the method, we further implement a prototype and perform a series of simulation-based experiments to demonstrate the effectiveness of our method in multi-agent cooperation compared to selected existing multi-agent methods to achieve the best rewards and drastically reduce data transmission during training.

Keywords: Multi-Agent System, Deep Reinforcement Learning, Learning Communication

* Supported by the Key-Area Research and Development Program of Guangdong Province (2021B010140005), Shenzhen Science and Technology Plan Project (Shenzhen-Hong Kong-Macau Category C, No. SGDX20220530111001003), and the Joint Research Project of Suntang-SIAT Research Lab for Big Data and AI, the National Key R&D Program of China (No. 2021YFB3300200), National Natural Science Foundation of China (No. 62072451, 92267105), Industrial application research project of Shenzhen for undertaking the national key project of China (No.CJGJZD20210408091600002), Chinese Academy of Sciences President's International Fellowship Initiative (No. 2023DT0003), Guangdong Special Support Plan (No. 2021TQ06X990), Shenzhen Basic Research Program (No. JCYJ20200109115418592, JCYJ20220818101610023).

1 Introduction

Multi-agent reinforcement learning (MARL) has emerged as a cutting-edge artificial intelligence technology that has achieved significant success in massive challenging tasks [10, 17, 13]. However, although MARL shows excellent prospects in solving optimization problems, it encounters many additional obstacles when adapted to real-world tasks [20, 19, 8]. One of them is the well-known non-stationary problem. The simultaneous action of multiple agents brings not only the dimensional explosion of the observation space, but also the difficulty of reaching consistency between actions and environment. Another is the partial observation problem. A single agent has a limited perspective and can only observe the situation in its own neighborhood, which leads to a lack of overall consideration in its decision-making. To overcome these technical hurdles, an approach is to aggregate all observations for centralized model training. Meanwhile, in the execution phase, the trained model relies on local observations to make decisions independently. Through optimization algorithms, such as value decomposition and credit assignment, this paradigm, called Centralized Training and Decentralized Execution (CTDE) [17, 13, 15], alleviates some of the difficulties in model training convergence. However, CTDE still suffers from chaotic decisions caused by partial observation at the execution stage.

Another intuitive idea comes from bionics, where animals use communication to negotiate and cooperate. Introducing communication into multi-agents means that agents can share their perceptions of the environment and their intentions to act, thus achieving unanimity. Agents typically encode their own observations and send them to other agents for decision-making. Open problems in learning communication include how to encode and decode messages, how to select communication objects, and how to design communication mechanisms [7, 14, 21, 19, 2]. Recent research including CommNet [16], IC3Net [14], TarMAC [1], and I2C [2] has significantly advanced the state of the art in these aspects and demonstrated the benefits of communication in MARL. Although promising, these methods all imply some form of centralization, such as broadcast communication [16, 6, 1], or concentrating all observations to train the encoder [7, 2].

To address these issues, we draw inspiration from social psychology, which finds that cognitive consistency within a neighborhood is important, and people tend to cooperate with their neighbors [11]. We design a novel neighbor-oriented learning communication approach, in which agents make decisions and train models using their neighbors' messages. In this paper, we make the following contributions: (1) We design an encoder network according to the idea of neighborhood cognitive consistency. By calculating the Kullback–Leibler (KL) divergence of messages from different neighbors, the encoder network can represent the consensus information of the neighborhood. (2) We propose a *pseudo-pre-acting* mechanism. This mechanism can send decision information along with messages to support the neighbors' decision-making and reduce the non-stationarity of MAS. (3) We develop a new learning communication method based on the actor-critic (AC) algorithm, *Neighborhood-Oriented Actor-Critic (NOAC)*, and construct experiments to validate our findings.

The organization of the paper is as follows: we discuss related works regarding MARL in Section 2 and introduce some background knowledge in Section 3. We illustrate the formulation and methodology in Section 4. Afterward, we present the simulation studies to validate our findings in Section 5, followed by the conclusion of the paper in the last section.

2 Related Work

An important challenge in MARL is how to characterize the interaction between agents, so numerous works propose that we can learn a joint value to guide agents' actions [10, 17, 13, 15, 5]. Based on this idea, researchers developed a widely used paradigm, Centralized Training Decentralized Execution (CTDE). In this paradigm, all agents share the information of joint value to mitigate non-stationarity. Although this paradigm has significantly improved the applicability of MARL, the lack of additional information in the execution phase still plagues the robustness of the action policy. To address the problem of agents' limited perspective, exchanging observations among agents to gain an understanding of the entire environment is an intuitive and advantageous idea. Many works have shown that learning communication is a promising approach [3, 16, 14, 1, 6, 7, 2]. These methods mainly focus on how to combine communication and deep reinforcement learning networks.

DIAL [3], a pioneer of learning communication, uses DQN to implement a learnable communication, introducing backpropagation to communication networks for the first time. The shortcoming of DIAL is that it can only handle discrete messages. Therefore, CommNet [16] employs a hidden layer to encode observations as continuous messages. DIAL and CommNet exchange messages between agents in a fully connected network, i.e. they communicate in a broadcast fashion. This mode of message delivery introduces redundant communication and massive inter-message interference. Lowe *et al.* [9] analyzed the urgency of messages and proposed two indicators, positive signaling and positive listening, to measure the utility of messages. Based on the concern that messages are not always useful for decision making, some works have tried to investigate how to communicate efficiently. One type of method, represented by ATOC [6] and IC3Net [14], introduced a gate mechanism to determine whether a message needs to be sent. In contrast, other alternative approaches, such as SchedNet [7], TarMAC [1], and I2C [2], adopted some weighting mechanisms to reduce communication between agents.

However, all these methods inevitably share a common problem: global information (including all observations and actions) is needed to compute TD-error during training. This leads to scalability problems when dealing with large-scale multi-agent systems. The dilemma stems from the fact that non-stationarity requires global information to counteract it. To this end, we designed a learning communication method that trains and executes depending on neighborhood information. One of the main differences between previous work and ours is that we no longer use global information to compute the TD-error. This setting is more practical and allows for flexible parallelization of training.

3 Background

3.1 Dec-POMDP

Decentralized Partially Observable Markov Decision Process (*Dec-POMDP*), is commonly used to characterize multi-agent systems where each agent can only partially observe the environment and follows a hidden Markov decision process for state transitions. A *Dec-POMDP* can be defined as a tuple:

$$\mathcal{D} = \langle \mathcal{N}, \mathcal{A}, \mathcal{R}, \mathcal{O} \rangle \tag{1}$$

here, $\mathcal{N}$ denotes the number of agents in total, $\mathcal{A} = a_1 \times ... \times a_{\mathcal{N}}$ is the set of joint action, $\mathcal{R} = \{r_1, ..., r_{\mathcal{N}}\}$ is the reward set, $\mathcal{O} = \{o_1, ..., o_{\mathcal{N}}\}$ is the set of observations, which satisfies $o_i \cup o_j \nsubseteq o_i$ or o_j.

Our goal is to guide the agent to achieve the maximum cumulative reward $\mathbb{E}[\sum_{t=0}^{+\infty} \gamma \vec{r}_t]$, here γ is the discount factor. To this end, we define a set of policies $\vec{\pi} = \{\pi_1(a_1|o_1), ..., \pi_{\mathcal{N}}(a_{\mathcal{N}}|o_{\mathcal{N}})\}$, and the final objective is to learn the optimal policy to maximize the cumulative reward.

$$\mathcal{J}(\theta) = \mathbb{E}_{a \sim \pi(\theta)}[\sum_{t=0}^{+\infty} \gamma \vec{r}_t] \tag{2}$$

Remarkably, considering that we are mainly focusing on the cooperative agents, there is typically only one global reward r, which is one of the main reasons why the convergence of MARLs is challenging.

3.2 Actor-Critic

Actor-Critic is a typical reinforcement learning algorithm that combines the advantages of value-based and policy gradient methods. It consists of two networks: the actor network and critic network. The critic network is used to estimate the current status value $V(o; \theta^c)$, which is updated by calculating the current TD-error:

$$\mathcal{L}(\theta^c) = \gamma * V(o'; \theta^c) + r - V(o; \theta^c) \tag{3}$$

Meanwhile, the actor network adopts the policy gradient method to perform actions for agents. The idea of policy gradient is to give the larger action value higher sampling probability. Therefore, combined with the advantage function of the critic network, its updating method is as follows:

$$\nabla_{\theta^a} \mathcal{J}(\theta^a) = \nabla_{\theta^a}[\log_{\theta^a} \pi_{\theta^a}(a|o) \mathcal{L}(\theta^c)] \tag{4}$$

It is worth noting that to avoid non-stationary problems in multi-agent training, it is typically assumed that the critic network is centralized, or that each critic can obtain global information through communication or other means. That is, o and a in the above formula are the set of observations and actions of all agents, respectively. Although this type of method has shown good convergence guarantees, the cost of aggregating all information is prohibitive in large-scale multi-agent systems.

4 Methodology

To overcome the centralized dilemma of multi-agent training, we take inspiration from [11]. In most cases, agents interact only with their neighbors, which is also consistent with interaction in human society. Therefore, we design a neighborhood-oriented method for multi-agent training in the subsequence section.

4.1 Neighborhood Cognitive

We denote the set of neighbors of agent i as $N(i)$, *i.e.*, agent $j \in N(i)$ is a neighbor of agent i. According to [11], there is a so-called **true hidden cognitive variable** C in each neighborhood, and all partial observations are the interplay of these variables. This assumption is intuitive, we can imagine that multiple

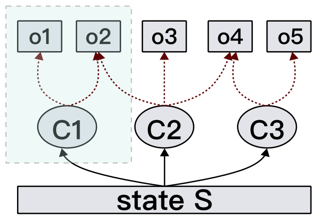


Fig. 1. The partial observations are generated from the hidden cognitive variables.

neighbors observe a global state S and attain multiple hidden variables C_k, and agent i observes C_k and get observation o_i, as shown in *Fig. 1.* We can assume that $\{C_k\}$ has a strong representation of global state S, and the observation of agent i can be derived as follows:

$$p(o_i|S) = \sum_k p(o_i|C_k) \tag{5}$$

Therefore, we can consider the aggregation of C_k as an intermediate representation of S. Although this hidden state does not change the situation that the global reward cannot be decomposed to a single agent (or neighborhood), it does mitigate some of the uncertainties through this transformation. As a consequence, we design an encoder network, $m_i = \mathcal{M}(o_i)$, to encode the observation of agent i and send it to its neighbors. We can rewrite the AC algorithm with neighborhood communication as follows:

$$\begin{aligned} \mathcal{L}(\theta_i^c) &= \gamma * V_{j\in N(i)}(m_i', m_j'; \theta_i^c) + r - V_{j\in N(i)}(m_i, m_j; \theta_i^c) \\ \nabla_{\theta_i^a}\mathcal{J} &= \nabla_{\theta_i^a}[\log_{\theta_i^a} \pi_{\theta_i^a; j\in N(i)}(a_i|m_i, m_j)\mathcal{L}(\theta_i^c)] \end{aligned} \tag{6}$$

Note that the input to the AC algorithm becomes an encoded set of messages. Neighborhood consistency assumes that all observations are based on the same hidden variable C_k, which means that o_i and o_j are inherently correlated, so we can achieve consistency in cognition by minimizing the differences between messages. Although the exact value of C_k is unknown, we can leverage KL-divergence to measure the difference between messages. Therefore, we derive the loss function of the encoder network as follows:

$$\mathcal{L}^e(\theta_i^m) = \sum_{j \in N(i)} D_{\mathrm{KL}}(P(m_i; \theta_i^m) \| P(m_j)) \tag{7}$$

This encoder network improves cognitive consistency in neighborhoods and significantly reduces the amount of data transferred.

4.2 Pseudo Pre-Acting

Passing messages brings additional information to the actor network and affects its decision-making by integrating neighborhood information. The neighborhood messages change the receiver's decision, and we use this differential to indicate the impact of these messages. Using the causal inference method, we can define indicators to show how an agent's decision making is influenced by its neighbors.

$$\mathbb{I}_i = D_{\mathrm{KL}, j \in N(i) \cup i}(\pi(a|o_i, m_j) \| \pi(a|o_i)) \tag{8}$$

Unfortunately, this influence shows the effect of neighborhood information on decision-making and does not indicate whether the changes increase or decrease the reward. Therefore, we consider the role of messages from a game theory perspective.

Multi-agent systems can usually be formalized as a game in which each agent takes its own action and receives a payoff. An important concept in game theory is the Nash equilibrium (NE), which means that the system is in a sort of steady state. Although the NE does not always maximize the social welfare (reward), it is robust enough and usually better than non-stationary solutions. Moreover, the maximum reward must also be an element of the NE set. Therefore, if we let multiple agents cooperatively reach a NE, we can change the problem of maximizing rewards to finding the optimal point among multiple NE states.

There are many ways to solve Nash equilibrium, and the most effective is to compute the best response. Let A_i and u be the action space and the utility function of agent i, respectively, then the best response can be computed as follows:

$$a^* = \arg\max_{a_i \in A_i} u(a_i, \vec{a}_{-i}) \tag{9}$$

here, $\vec{a}_{-i}$ represents the action set of all agents except i. One of the definitions of Nash equilibrium is that all agents are in the best response state, which means that no agent can unilaterally change its action to get better rewards. That is, if agents know the actions of other agents, they can adopt strategies to get better rewards. We can share agents' actions through communication and calculate actions through $\pi_i(a_i|o_i, a_j; j \in N(i))$.

However, it is not practical to send all neighborhood actions to agent i for decision-making, because agents act simultaneously rather than sequentially. When an agent receives messages from other agents and changes its action, this change will lead to new changed actions of other agents. To avoid this chain reaction, we propose a two-step decision-making method:

1) obtain $\hat{a}_i$ by $\hat{\pi}_i(\hat{a}_i|o_i)$, and send it to neighbors;

2) execute action $\pi_i(a_i|o_i, a_j; j \in N(i))$ after the actions of neighbors are received.

We refer to this approach as **pseudo-pre-acting (PPA)** mechanism, in which $\hat{a}$ and $\hat{\pi}$ are called pseudo action and pseudo policy, respectively. Note that π_i is the policy we actually learned by interacting with the environment, so we update the pseudo-policy network $\hat{\pi}_i$ with the causal inference indicator mentioned above:

$$\mathcal{L}^{\hat{\pi}}(\theta_i^m) = D_{\mathrm{KL}, j \in N(i)}(\hat{\pi}_i(\hat{a}_i|o_i; \theta_i^m) \| \pi_i(a_i|o_i, a_j, m_j)) \tag{10}$$

There are two perspectives to explain why we make the pseudo action approximate the actual action: On the one hand, the consistency of the pseudo action and the actual action makes the best response calculated by other agents effective; on the other hand, under the premise of receiving other actions, the consistency of the two types of actions indicates that the current state is in some kind of equilibrium.

4.3 Neighborhood-Oriented Actor-Critic

Combined with the above methods, we propose a neighborhood-oriented MARL approach based on actor-critic: **Neighborhood-Oriented Actor-Critic (NOAC)**. The overall architecture of NOAC is illustrated in *Fig. 2*, which consists of three parts: encoder network, actor network, and critic network.

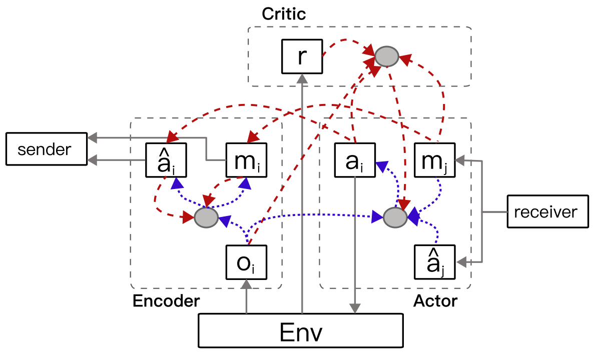


Fig. 2. The overall architecture of NOAC. The blue lines are the execution dataflow, while the red lines are training dataflow. Gray circles are neural networks.

In the execution phase, the encoder network encodes the local observations, outputs the message m_i and pseudo action $\hat{a}_i$, and then sends them to its neighbors. After receiving messages and pseudo actions from its neighbors, the actor network selects appropriate actions to interact with the environment and then moves to the next epoch.

In the training stage, the encoder network calculates $\mathcal{L}^e$ and $\mathcal{L}^{\hat{\pi}}$ according to the received messages and the actions performed by the actor network, respectively, and updates parameters with the following loss function:

$$\mathcal{L}(\theta_i^m) = \mathcal{L}^e(\theta_i^m) + \mathcal{L}^{\hat{\pi}}(\theta_i^m) \tag{11}$$

Concerning the actor network, we adopt the policy gradient to update parameters as follows:

$$\nabla_{\theta_i^a}\mathcal{J} = \nabla_{\theta_i^a}[\log_{\theta_i^a}\pi_{\theta_i^a;j\in N(i)}(a_i|o_i,m_i,a_j,m_j)\mathcal{L}(\theta_i^c)] \tag{12}$$

The main concern for training of the critic network is the calculation of TD-loss. As mentioned above, the critic network is usually centralized because it needs to estimate Q_{total}, which directly influences the global reward. Since the global reward cannot be directly assigned to individuals in multi-agent cooperation, the centralized network is needed to evaluate the value function. Likewise, although we propose that C_k can characterize part of the global state, we still cannot assign the global reward to a concrete C_k. However, considering that in some environments where agents move, each agent transforms the C value as it moves, we have relaxed this constraint. Since the size of the neighborhood directly affects the approximation of Q_C and Q_{total}, we define the TD-loss of the critic network as follows:

$$\mathcal{L}(\theta_i^c) = \gamma * V_{j\in N(i)\cup i}(m_i', m_j'; \theta_i^c) + \frac{|N(i)|}{\mathcal{N}} * r - V_{j\in N(i)\cup i}(m_i, m_j; \theta_i^c) \tag{13}$$

Due to the decentralized network design, all agents run in parallel during execution and training, which only need to be synchronized during communication and environment steps. Therefore, the networks of agents can be deployed in different servers and communicate over protocols such as GLOO, NCCL, or TCP. This feature is particularly efficient in large-scale multi-agent environments.

5 Experiments

To validate our findings, we conducted empirical studies to evaluate the performance of the proposed NOAC. We implemented a test platform based on multi-agent particle environment [12] and took a cooperative game as the environment simulator.

5.1 Setup

Environment. We took the cooperative navigation game [10] as the simulation environment. As shown in *Fig. 3*, there are $\mathcal{N}$ agents and $\mathcal{N}$ landmarks in the environment, and the agents need to cooperate with each other to occupy all the landmarks. The environment takes the sum of the minimum distance between all agents and landmarks as the global reward value, that is, there is no individual reward for each agent.

Each agent has its own observation o_i and takes the closer agents as its neighbors. Each agent can only stand on one landmark, and there is a penalty (negative reward) for collisions. In the experiment, we set $\mathcal{N} = 7$, $N(i) = 3$, and each agent starts at a random position. All other settings, such as the agent's speed, are default values from the open source library *PettingZoo* [18]. Note that the agent's environment is open and not restricted to a specific area, which may differ from some other cooperative navigation settings.

Baselines. We compared the proposed method NOAC with the following state-of-the-art MARL baselines :

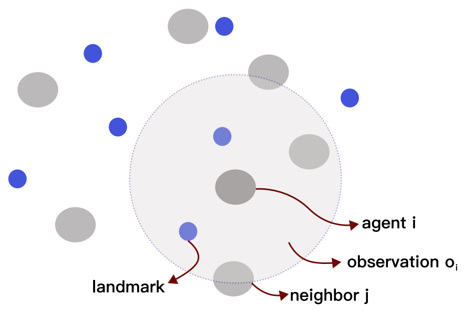


Fig. 3. Cooperative navigation environment.

- *TarMAC* [1]: An attention-based learning communication method that weighs the importance of incoming messages.

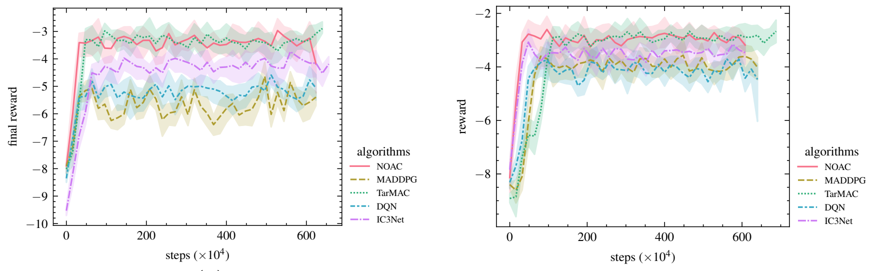
- *IC3Net* [14]:A gate-based method for deciding whether to communicate with others, in which messages are sent in broadcast mode.

- *MADDPG* [10]:A classical CTDE algorithm without communication.

- *DDQN* [4]:A typical single agent algorithm. In our setting, it can sense the global state and output all agents' actions simultaneously.

Hyperparameters. To reduce the off-site factors in the comparison, we adopted the same network structure in most baselines. In addition, we set the learning rate $lr = 1 \times 10^{-3}$ and $batch\_size = 64$ for all methods. We implemented the testbed based on *PyTorch* and *PettingZoo* and ran it on a $3 \times$ Tesla V-100 server.

5.2 Numerical Results



(a) (b)

Fig. 4. Comparison of final reward (a) and total reward (b).

Global Reward. First, we examined the rewards of multiple baselines, which is the primary concern in MARL problems. Previous cooperative navigation experiments often focused only on the average reward during the entire training (referred to as "total reward" in the subsequent section), while we are more concerned with the final state of the agent at the end of the round, the final reward. Therefore, we compared the two types of rewards of baselines: final reward and total reward.

The comparison of the final rewards is illustrated in *Fig. 4(a)*, where the final rewards of *NOAC* are approximately equal to *TarMAC*, and more significant than other baselines. This result shows the effectiveness of our approach. It is worth noting that the baselines, including *TarMAC*, require centralized information exchange, while *NOAC* only collects neighborhood information. Similarly, *NOAC* also performs well on total rewards, as shown in *Fig. 4(b)*. It should be noted that the gap between baselines on total rewards differs less than that on final rewards, so we treat the final reward as the metric for the following experiments.

We executed 500 episodes with these trained models, and *Tab. 1* shows the means and standard deviations of the total and final rewards. *NOAC* outperforms all baselines with the highest average final rewards. These results suggest that our partial information and message training approach is comparable to methods that require centralized training.

Table 1. Summary of final reward and total reward of MARL baselines

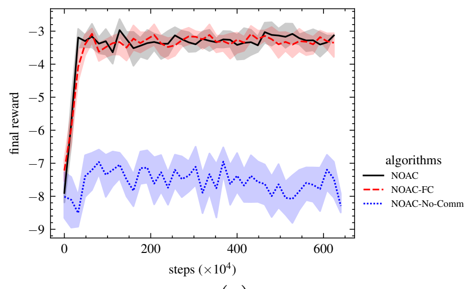
Algorithms	**NOAC**	**MADDPG**	**TarMAC**	**IC3Net**	**DQN**
Final Reward Mean	**-3.292036**	-5.627667	-3.335745	-4.164599	-5.120261
Final Reward Std	0.595101	0.646373	0.766422	0.457646	0.626979
Total Reward Mean	-3.358014	-4.168850	-3.201513	-3.512844	-4.132432
Total Reward Std	1.430986	1.219422	1.408928	0.964834	0.860662

Neighborhood Impact. Although *NOAC* is designed for neighborhoods, it can also handle broadcast messages. To investigate whether *NOAC* does indeed includes neighbors' messages into decision making, we compare it with two cases: one where there is no communication at all, which is a decentralized AC algorithm with independent control, which we call "*NOAC-No-Comm*"; the other is a fully connected communication network with *NOAC*, which we call "*NOAC-FC*".

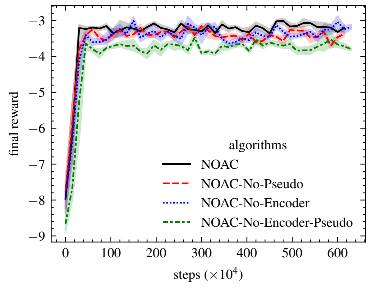
After training with the same settings, the comparison of the experimental results is illustrated in *Fig. 5(a)*. Not surprisingly, the decentralized algorithm *NOAC-No-Comm* without communication performs the worst. In fact, it performs worse than all other baselines because it uses only local observations to train the agents. This result confirms that communication significantly improves MARL.

Nevertheless, it is worth noting that *NOAC-FC* with fully connected communication does not provide a significant improvement in performance. At the beginning of training, the convergence of *NOAC-FC* is even slower than that of the partially connected case. This result illustrates that more messages are not always better, and actually is consistent with the selective communication proposition in related work. It also shows that neighborhood cognitive consistency exists, and the agent can achieve an approximate effect of global observation through the messages of neighbors.

Ablation. Finally, to further investigate the contribution of the encoder network and pseudo-pre-acting to *NOAC*, we performed ablation experiments. We conducted three different sets of experiments: 1) *NOAC* without pseudo-pre-acting mechanism, "*NOAC-No-Pseudo*"; 2) NOAC without encoding the neighbor observations, "*NOAC-No-Encoder*". Note that the raw neighbor observations are still transmitted; 3) the "*NOAC-No-Encoder-Pseudo*" algorithm



(a)



(b)

Fig. 5. Reward comparison of different neighborhood impacts (a) and ablation experiments (b).

without encoder or pseudo-pre-acting mechanism, where each agent aggregates the neighbor observations for training.

Fig. 5(b) shows a comparison of the ablation experiments, and it can be seen that the removal of either mechanism leads to a slight decrease in performance. In particular, the *NOAC-No-Encoder-Pseudo* with the encoder and **PPA** removed shows a significant drop in reward. To illustrate the difference more clearly, *Tab. 2* shows the difference in reward for each setting. Note that all methods use information from neighbors.

Table 2. Summary of the final reward of ablation experiments

	NOAC	**No-Encoder**	**No-PPA**	**No-Encoder-PPA**
Final Reward Mean	-3.292036	-3.458257	-3.514603	-3.828242
Final Reward Std	0.595101	0.411429	0.955895	0.592898

Despite the lack of an encoder, the actor and critic networks can still extract information from raw observations. Thus, the gap is not that significant. However, the encoded messages are about 3/4 times smaller than the original observations ($42 \rightarrow 12$), which shows a significant advantage in terms of latency in both network transmission and tensor operations.

6 Conclusions

In this paper, we proposed a neighborhood-oriented MARL training method that uses only messages from neighbors instead of global information to learn policies. Experiments show that this decentralized training method is comparable to mainstream CTDE methods. This approach shows the potential of decentralized learning methods for solving MARL problems. This decentralized MARL learning is not only closer to real-world scenarios, but also has excellent advantages in terms of computational efficiency. We expect that this paradigm could be further developed and applied to more practical problems.

References

1. Das, A., Gervet, T., Romoff, J., Batra, D., Parikh, D., Rabbat, M., Pineau, J.: Tarmac: Targeted multi-agent communication. In: ICML (2019)

2. Ding, Z., Huang, T., Lu, Z.: Learning individually inferred communication for multi-agent cooperation. NeurIPS (2020)
3. Foerster, J., Assael, I.A., de Freitas, N., Whiteson, S.: Learning to communicate with deep multi-agent reinforcement learning. In: NeurIPS (2016)
4. Hasselt, H.v., Guez, A., Silver, D.: Deep reinforcement learning with double q-learning. In: AAAI (2016)
5. Iqbal, S., Sha, F.: Actor-attention-critic for multi-agent reinforcement learning. In: International Conference on Machine Learning (ICML) (2019)
6. Jiang, J., Lu, Z.: Learning attentional communication for multi-agent cooperation. Advances in Neural Information Processing Systems (NeurIPS) (2018)
7. Kim, D., Moon, S., Hostallero, D., Kang, W.J., Lee, T., Son, K., Yi, Y.: Learning to schedule communication in multi-agent reinforcement learning. In: International Conference on Learning Representations (ICLR) (2019)
8. Leonardos, S., Overman, W., Panageas, I., Piliouras, G.: Global convergence of multi-agent policy gradient in markov potential games. In: ICLR (2022)
9. Lowe, R., Foerster, J., Boureau, Y.L., Pineau, J., Dauphin, Y.: On the pitfalls of measuring emergent communication. In: AAMAS (2019)
10. Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, O.P., Mordatch, I.: Multi-agent actor-critic for mixed cooperative-competitive environments. In: NeurIPS (2017)
11. Mao, H., Liu, W., Hao, J., Luo, J., Li, D., Zhang, Z., Wang, J., Xiao, Z.: Neighborhood cognition consistent multi-agent reinforcement learning. AAAI (2020)
12. Mordatch, I., Abbeel, P.: Emergence of grounded compositional language in multi-agent populations. arXiv preprint arXiv:1703.04908 (2017)
13. Rashid, T., Samvelyan, M., de Witt, C.S., Farquhar, G., Foerster, J., Whiteson, S.: Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In: International Conference on Machine Learning (ICML) (2018)
14. Singh, A., Jain, T., Sukhbaatar, S.: Individualized controlled continuous communication model for multiagent cooperative and competitive tasks. In: International Conference on Learning Representations (ICLR) (2019)
15. Son, K., Kim, D., Kang, W.J., Hostallero, D.E., Yi, Y.: Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In: International Conference on Machine Learning (ICML) (2019)
16. Sukhbaatar, S., Fergus, R., et al.: Learning multiagent communication with backpropagation. In: NeurIPS (2016)
17. Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W.M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J.Z., Tuyls, K., Graepel, T.: Value-decomposition networks for cooperative multi-agent learning based on team reward. In: AAMAS (2018)
18. Terry, J.K., Black, B., Grammel, N., Jayakumar, M., Hari, A., Sulivan, R., Santos, L., Perez, R., Horsch, C., Dieffendahl, C., Williams, N.L., Lokesh, Y., Sullivan, R., Ravi, P.: Pettingzoo: Gym for multi-agent reinforcement learning. arXiv preprint arXiv:2009.14471 (2020)
19. Wang, T., Wang, J., Zheng, C., Zhang, C.: Learning nearly decomposable value functions via communication minimization. In: ICLR (2020)
20. Wen, Y., Yang, Y., Wang, J.: Modelling bounded rationality in multi-agent interactions by generalized recursive reasoning. In: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence. IJCAI'20 (2021)
21. Zhang, S.Q., Zhang, Q., Lin, J.: Efficient communication in multi-agent reinforcement learning via variance based control. In: Advances in Neural Information Processing Systems (NeurIPS) (2019)