What Kind of Information Is Needed? Multi-Agent Reinforcement Learning that Selectively Shares Information from Other Agents

Riku Sakagami[1] and Keiki Takadama[2,3][a]

[1]*Department of Informatics, The University of Electro-Communications, Tokyo, Japan*

[2]*Information Technology Center, The University of Tokyo, Tokyo, Japan*

[3]*Department of Information & Communication Engineering, The University of Tokyo, Tokyo, Japan*

Keywords: Multi-Agent Reinforcement Learning, Centralized Training with Decentralized Execution, Sharing Additional Information.

Abstract: Since agents' learning affects others' learning in multi-agent reinforcement learning (MARL), this paper aims to clarify what kind of information helps to improve learning of agents throgh complex interactions among them. For this purpose, this paper focuses on the information on observations/actions of other agents and analyzes its effect in MARL with the centralized training with decentralized execution (CTDE), which contributes to stabilizing agents' learning. Concretely, this paper extends the conventional MARL algorithm with CTDE (i.e., MADDPG in this research) to have the two mechanisms, each of which shares (i) information on observations of all agents; (ii) information on actions of all agents; and (iii) information on both the observations and actions of the selected agents. MARL with these three mechanisms is compared with MADDPG which shares information on both actions and observations of all agents and IDDPG which does not share any information. The experiments on multi-agent particle environments (MPEs) have revealed that the proposed method that selectively shares both observation and action information is superior to the other methods in both the full and partial observation environments where information on observations of all and selected agents.

1 INTRODUCTION

Recently, Multi-Agent Reinforcement Learning (MARL) (Zhang et al., 2021; Gronauer and Diepold, 2022) has attracted much attention on by successfully applying them to many complex real-world problems, such as automated driving and autonomous robot swarm control (Shalev-Shwartz et al., 2016; Hernandez-Leal et al., 2019). MARL aims at adaptively controlling the learning of agents to cope with given environments. However, it is hard for agents in MARL to learn their appropriate policies simultaneously due to nonstationary situations caused by complex interactions among agents (Busoniu et al., 2008; Hernandez-Leal et al., 2017). Such a non-stationarity in MARL is caused by updating the policy of all agents at the same time, i.e., the state of an agent is not always to be transited to the unique one even if the agent selects the same action. This is because the actions of other agents may be different.

The simplest way to tackle this difficulty in simultaneous learning is to employ global observation (which includes the state of all agents) and the actions of other agents when agents learn their policies. From this viewpoint, the Centralized Training with Decentralized Execution (CTDE) method was proposed in MARL (Lowe et al., 2017; Yu et al., 2022), which assumes that agents can access extra information (i.e., global observation and the actions of other agents) only during learning (as the centralized training) and determine their actions according to their individual information (as decentralized execution). This approach contributes to not only stabilizing the learning of agents but also shortening the learning time. The previous works of CTDE revealed its superiority to conventional methods (Lowe et al., 2017; Foerster et al., 2018; Rashid et al., 2020; Yu et al., 2022). Such information of other agents helps agents to learn their appropriate policies consistently in MARL.

What should be noted here, however, is that too much information (i.e., "both" the observation and actions of "all" agents) in CTDE is not always needed for learning policies of agents, but rather it may have a bad influence on their learning. To tackle this issue, this paper aims to clarify what kind of information

[a] https://orcid.org/0009-0007-0916-5505

Sakagami, R. and Takadama, K.
What Kind of Information Is Needed? Multi-Agent Reinforcement Learning that Selectively Shares Information from Other Agents.
DOI: 10.5220/0013390100003890
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 17th International Conference on Agents and Artificial Intelligence (ICAART 2025) - Volume 1*, pages 235-242
ISBN: 978-989-758-737-5; ISSN: 2184-433X
Proceedings Copyright © 2025 by SCITEPRESS – Science and Technology Publications, Lda.

helps agents learn in CTDE. Concretely, this paper focuses on the information on observations and actions of other agents, which is essential for cooperation in CTDE, and analyzes its effect by employing (i) information on “observations” of “all” agents; (ii) information on “actions” of “all” agents; or (iii) information on both the observations and actions of the “selected” agents. This information classification aims at clarifying what is an essential difference between “observations” and “actions” from the viewpoint of learning, and how the performance of agents changes by employing information of the “selected” agents from that of “all” agents.

Among the many MARL methods based on CTDE, this paper starts to employ MADDPG (Multi-Agent Deep Deterministic Policy Gradient) (Lowe et al., 2017) which shares information on both observations and actions and of all agents, and extends it by changing three kinds of the above information from (i) to (iii). MADDPG is the Actor-Critic based MARL with CTDE and is focused on in this paper because of the following reasons: (1) MADDPG is the simple method developed in the early stage of CTDE, which helps us to directly clarify an effect of the three kinds of the above information without considering any other effect caused by the complex or sophisticated CTDE proposed after MADDPG; (2) Unlike QMIX (Rashid et al., 2020) and VDN (Value Decomposition Network) (Sunehag et al., 2018) which are based on IGM principle that assumes that an action of an agent with the highest value from the local (agent) viewpoint is the same as that from the global (multi-agent) viewpoint, MADDPG can derive a cooperation where some agents give up to select their own best actions to improve a global performance because MADDPG is not based on the IGM (Individual Global-Max) principle; (3) Unlike MAPPO (Multi-Agent Proximal Policy Optimization) (Yu et al., 2022) which shares parameters of neural networks of all agents which is useful for homogeneous agents acquiring the same policy among the agents, MADDPG can be applied for heterogeneous agents acquiring the different policies among the agents because MADDPG has independent policies of agents.

To clarify what kind of information helps agents to learn in CTDE, this paper conducts the experiments on multi-agent particle environments (MPEs) and compares the results of the modified MADDPGs which share one of three kinds of information, the original MADDPG which shares all information, and IDDPG (Independent Deep Deterministic Policy Gradient) which does not share any information as MARL with decentralized training with decentralized execution (DTDE).

The paper is organized as follows: Section 2 introduces the related works of MARL with CTDE, and Section 3 describes MADDPG as the conventional method. Section 4 proposes the information-sharing methods. Section 5 conducts the experiment and analyzes the experimental results. Finally, the conclusions and future work of this study are discussed in Section 6.

2 RELATEDWORK

2.1 Centralized-Training with Decentralized-Execution

The Centralized-Training with Decentralized-Execution (CTDE) paradigm is an essential concept in recent MARL. In a CTDE setting, each agent i has an independent policy π_i and makes independent action decisions according to local observations only. This behavior is the same as that at execution time (decentralized execution). On the other hand, during training, agents have access to additional information, such as global observations and information sharing among agents (centralized training). This additional information is discarded at runtime and never used. The strength of CTDE lies in its potential to improve learning speed and accuracy by mitigating the partial observability and non-stationarity of multi-agent systems with additional information during training. Its potential application to real-world problems is also high since it behaves as a fully independent agent at runtime. Many problems are still challenging to solve with Decentralized-Training with Decentralized-Execution (DTDE), and CTDE shows excellent results while reducing the strong assumptions of Centralized-Training with Centralized-Execution (CTCE)(Gronauer and Diepold, 2022; Zhang et al., 2021).

MADDPG(Lowe et al., 2017), COMA(Foerster et al., 2018), QMIX(Rashid et al., 2020), and MAPPO(Yu et al., 2022) are known as representative MARL methods based on CTDE. MADDPG extends DDPG((Lillicrap et al., 2015)) to a multi-agent environment and, following the CTDE, stabilizes learning in a multi-agent environment by allowing the centralized critic to use additional information (in the paper, pairs of observations and actions of all other agents). In addition to being usable in cooperative, adversarial, and mixed tasks, the method has the advantage of being a very versatile algorithm, supporting both homogeneous and heterogeneous agents. COMA addresses the problem of multi-agent credit assignment by training a single centralized critic to calculate each agent's advantage function based on a counterfactual baseline inspired by difference rewards(Wolpert and Tumer, 2001). QMIX is a method proposed as an improvement of the value decomposition network (VDN)(Sunehag et al., 2018). It learns a joint action-value function that can be decomposed into individual action-value functions by using a centralized network called a mixing network that bundles the Q-function networks of each agent. This decomposition is based on a monotonicity constraint between each Q-function and joint Q-functions. MAPPO is a method that extends PPO(Schulman et al., 2017) to a multi-agent environment following the CTDE. The original MAPPO uses additional information through a shared policy network, but can also consider individual policy networks, as can MADDPG, a method based on the same actor-critic. In that case, the major difference between the two methods is which algorithm is used, on-policy PPO or off-policy DDPG. In general, on-policy methods are known to be inferior to off-policy methods in terms of sample efficiency.

As discussed in the previous paragraph for algorithms that use CTDE, several types of additional information can be used during learning in the CTDE, especially three commonly used types: network parameter sharing, value factorization, and information sharing.

- **Network Parameter Sharing.** In parameter sharing, a single network is used to optimize the policies of all agents, or each agent has its network for policy improvement, and its parameters are shared (e.g., at regular learning intervals); MAPPO (original work) is classified as this method. This method has the advantage of being insensitive to the scalability of agents, but its network sharing nature limits it to cooperative tasks performed by homogeneous agents.
- **Value Factorization.** In value factorization, the joint Q-function is learned to optimize individual Q-functions based on the IGM principle (Son et al., 2019); VDN and QMIX are classified as this method in which the joint Q-function is additional information. This method allows for accurate Q-function estimation but requires the assumption that the joint Q-function is factorizable. Also, its application to competitive tasks is limited due to its nature.
- **Information Sharing.** In information sharing, additional information such as the state, observation, and action of each agent and the global state of the environment is used for information sharing; MADDPG and MAPPO are classified as this method. Since this method is a simple approach to adding environmental information, it can be used regardless of the nature of the task (cooperative/competitive) or the nature of the agents (homogeneous/heterogeneous) and can optimize the agents' policies independently. On the other hand, loose constraints tend to result in local solutions, and the addition of environmental information is easily influenced by the agent's scalability.

In this study, we focus on information sharing, which directly addresses non-stationarity and partial observability, and evaluate the impact of additional information on learning accuracy and stability. To this end, MADDPG is employed as the baseline algorithm for comparison. This choice is justified by its status as the most concise actor-critic-based method capable of leveraging information sharing, as well as its high versatility, being independent of the specific nature of the problem or agents. In contrast, the other methods discussed above have limitations, such as reliance on the IGM principle or the assumption of homogeneous agents.

3 BACKGROUND

3.1 Problem Settings

Stochastic games (sometimes called Markov games)(Littman, 1994) is a natural extension of Markov decision processes (MDP) to multi-agent environment. In a more realistic setting, the agent may not have access to information about the complete environment, in which case it is modeled by Partially observable stochastic games (POSG)(Hansen et al., 2004). A POSG is defined by a tuple $\langle \mathcal{I}, \mathcal{S}, b^0, \{\mathcal{A}_i\}_{i\in\mathcal{I}}, \{\mathcal{O}_i\}_{i\in\mathcal{I}}, \mathcal{P}, \{R_i\}_{i\in\mathcal{I}} \rangle$, where $\mathcal{I} = \{1, \ldots, N\}$ is a set of agents indexed, $\mathcal{S}$ is a set of all states, b^0 is the initial state distribution, $\mathcal{A}_i$ is a set of actions available to agent i, $\mathcal{O}_i$ is a set of observations for agent i, $\mathcal{P} : \mathcal{S} \times \overrightarrow{\mathcal{A}} \times \mathcal{S} \times \overrightarrow{\mathcal{O}} \to [0,1]$ is a set of state transitions and observation probabilities(, where $\overrightarrow{\mathcal{A}} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n$ denotes a set of joint actions, $\overrightarrow{\mathcal{O}} = \mathcal{O}_1 \times \cdots \times \mathcal{O}_n$ denotes a set of joint observations, $\mathcal{P}(s', \overrightarrow{o} \mid s, \overrightarrow{a})$ denotes the probability that taking joint action $\overrightarrow{a} \in \overrightarrow{\mathcal{A}}$ in state $s \in \mathcal{S}$ results in a transition to state $s' \in \mathcal{S}$ and joint observation $\overrightarrow{o} \in \overrightarrow{\mathcal{O}}$), $R_i : \mathcal{S} \times \overrightarrow{\mathcal{A}} \times \mathcal{S} \to \mathbb{R}$ is a reward function for agent i, and each agent attempts to maximize its expected sum of discounted rewards, $\mathbb{E}\left[\Sigma_t^T \gamma^t R_i(s_t, \overrightarrow{a}_t, s_{t+1})\right]$, where γ is a discount factor and T is the time horizon.

3.2 Multi-Agent Deep Deterministic Policy Gradient (MADDPG)

In an N-agent POSG, if each agent employs a policy $\boldsymbol{\mu} = \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_N\}$ parameterized by $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_N\}$, the gradient of the expected return for agent i, $J(\boldsymbol{\mu}_i) = \mathbb{E}[R_i]$ in MADDPG is given by

$$\nabla_{\theta_i} J(\boldsymbol{\mu}_i) = \mathbb{E}_{\overrightarrow{o}, \overrightarrow{a} \sim \mathcal{D}} \left[\nabla_{\theta_i} \boldsymbol{\mu}_i(a_i \mid o_i) \nabla_{a_i} Q_i^{\boldsymbol{\mu}}(\overrightarrow{o}, \overrightarrow{a}) \mid_{a_i = \boldsymbol{\mu}(o_i)} \right]. \tag{1}$$

Here $Q_i^{\boldsymbol{\mu}}(\overrightarrow{o}, \overrightarrow{a})$ represents a centralized actio-value function that takes as input the observations and actions of all agents. The replay buffer $\mathcal{D}$ stores tuples of the form $(\overrightarrow{o}, \overrightarrow{a}, r_1, \ldots, r_N, \overrightarrow{o}')$, recording the experiences of all agents. The centralized action-value function is updated by

$$\begin{aligned} \mathcal{L}(\theta_i) &= \mathbb{E}_{\overrightarrow{o}, a, r, \overrightarrow{o}'} \left[(Q_i^{\boldsymbol{\mu}}(\overrightarrow{o}, \overrightarrow{a}) - y)^2 \right], \\ y &= r_i + \gamma Q_i^{\boldsymbol{\mu}'}(\overrightarrow{o}', \overrightarrow{a}') \mid_{a_j' = \boldsymbol{\mu}_j'(o_j)}. \end{aligned} \tag{2}$$

Here $\boldsymbol{\mu}' = \{\boldsymbol{\mu}_{\theta_1'}, \ldots, \boldsymbol{\mu}_{\theta_N'}\}$ is the set of target policies with delayed parameters θ_i'.

4 METHODS

The most basic information shared during learning in methods following the CTDE framework is the observations and actions of each agent. We consider that there are stages in sharing this information, shown in Table 1. Usually, independent information is used when sharing is not assumed (e.g., IDDPG is shown in Figure 1), and full information is used when sharing is assumed (e.g., MADDPG is shown in Figure 2), while

1. The case in which only observations are shared (Table 1 upper-right cell).
2. The case in which only actions are shared (Table 1 lower-left cell).
3. The case in which information is selected and shared (Table 1 middle-center cell).

Algorithms that rely on independent information face challenges in addressing the non-stationarity inherent in multi-agent systems. Conversely, algorithms that share and utilize all available information risk having critical information obscured by other excessive information. Building on the above, this paper outlines the three types of sharing methods that are the focus of this study.

Table 1: Degree of sharing information.

act\obs	non	partial	full
non	IDDPG	–	target 1
partial	–	target 3	–
full	target 2	–	MADDPG

4.1 Sharing Full-Observations

In this shared method, each agent estimates a Q-function based on the joint observations of all other agents and its own action, as illustrated in Figure 3. Specifically, the input to critic i is a vector comprising the joint observations $\overrightarrow{o}$ and the agent's own action a_i. Consequently, even in a partially observable environment, the critic can approximate global environmental information through the joint observations. To be precise, the ability to derive an observation representation that closely approximates the global observation from multiple local observations depends on the neural network training process and the spatial distribution of agents. However, since the actions of other agents are not shared, the issue of non-stationarity remains unresolved. The Algorithm 1 represents the pseudocode for the update function of this sharing method with MADDPG.

4.2 Sharing Full-Actions

In this sharing method, each agent estimates a Q-function based on its own observation and the joint actions of all other agents, as illustrated in Figure 4. Specifically, the input to critic i is a vector comprising the agent's own observation o_i and the joint actions $\overrightarrow{a}$. Consequently, the closer the local observations are to the global observations, the more effectively this critic can mitigate non-stationarity. However, this approach cannot address the issue of partial observability. The Algorithm 2 represents the pseudocode for the update function of this sharing method with MADDPG.

4.3 Sharing Selected Observations and Actions

In this sharing method, each agent estimates a Q-function based on its own observations and actions, as well as those of selected agents, as illustrated in Figure 5. Specifically, the input to critic i is a vector comprising the selected observations $\{o_i, o_j\}_{j \in I_{i,\text{select}}}$ and the selected actions $\{a_i, a_j\}_{j \in I_{i,\text{select}}}$, where $I_{i,\text{select}} \subseteq I$ denotes the set of indices corresponding to the selected agents for each agent i. The criteria for selecting agents can be arbitrary (e.g., based on distance, role, attention, etc.). Consequently, this critic cannot completely mitigate non-stationarity and partial observability unless the number of selected agents matches the total number of other agents in the environment. Nevertheless, excluding agents that are weakly related to the target agent may alleviate scalability issues while reducing redundancy in the shared information. The Algorithm 3 represents the pseudocode for the update function of this sharing method with MADDPG.

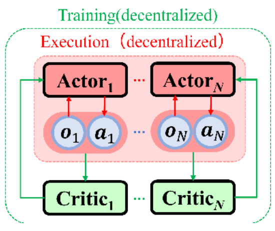

Figure 1: IDDPG

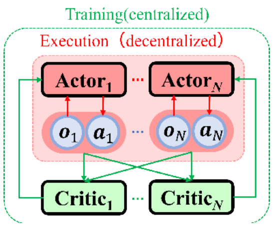

Figure 2: MADDPG.

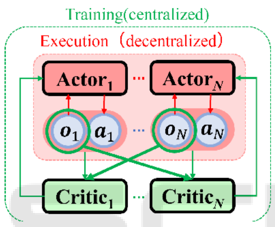

Figure 3: MADDPG with Sharing Full-Observations

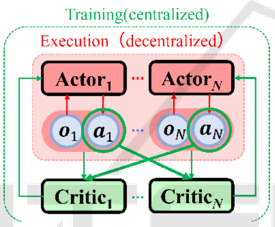

Figure 4: MADDPG with Sharing Full-Actions.

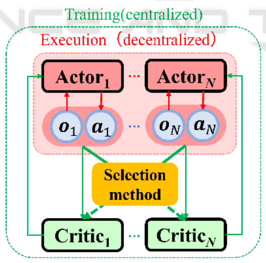

Figure 5: MADDPG with Sharing selected Observations & Actions.

5 EXPERIMENTS

5.1 Multi-Agent Particle Environments

For the experimental environment, we use cooperative navigation, one of the multi-agent particle environments (MPEs) provided in (Lowe et al., 2017).

Cooperative Navigation. This environment is a cooperative task in which multiple agents and an equal number of goals are placed in a two-dimensional real-valued space, and the target is for all agents to cover

Algorithm 1: Update actor and critic function for "Sharing Full-Observations".

```
function UPDATE_AGENT($i$)
    Sample a random minibatch of $S$ samples
    $(\vec{o}^{\,j}, \vec{a}^{\,j}, \mathbf{r}^j, \vec{o}^{\,\prime j})$ from replay buffer $\mathcal{D}$
    Set $y^j = r_i^j + \gamma Q_i^{\mu'}(\vec{o}^{\,\prime j}, a_i^{\prime j})\,|_{a_k'=\mu_k'(o_k')}$
    Update critic by minimizing the loss:
        $\mathcal{L}(\theta_i) = \frac{1}{S}\sum_j \left(y^j - Q_i^{\mu}(\vec{o}^{\,j}, a_i^j)\right)^2$
    Update actor using the sampled policy gradient:
        $\nabla_{\theta_i} J \approx \frac{1}{S}\sum_j \nabla_{\theta_i}\mu_i(o_i^j)\nabla_{a_i} Q_i^{\mu}(\vec{o}^{\,j}, a_i^j)\,|_{a_i=\mu_i(o_i^j)}$
```

Algorithm 2: Update actor and critic function for "Sharing Full-Actions".

```
function UPDATE_AGENT($i$)
    Sample a random minibatch of $S$ samples
    $(\vec{o}^{\,j}, \vec{a}^{\,j}, \mathbf{r}^j, \vec{o}^{\,\prime j})$ from replay buffer $\mathcal{D}$
    Set $y^j = r_i^j + \gamma Q_i^{\mu'}(o_i^{\prime j}, \vec{a}^{\,\prime j})\,|_{a_k'=\mu_k'(o_k')}$
    Update critic by minimizing the loss:
        $\mathcal{L}(\theta_i) = \frac{1}{S}\sum_j \left(y^j - Q_i^{\mu}(o_i^j, \vec{a}^{\,j})\right)^2$
    Update actor using the sampled policy gradient:
        $\nabla_{\theta_i} J \approx \frac{1}{S}\sum_j \nabla_{\theta_i}\mu_i(o_i^j)\nabla_{a_i} Q_i^{\mu}(o_i^j, \vec{a}^{\,j})\,|_{a_i=\mu_i(o_i^j)}$
```

all goals. In this task, the goal of each agent is not explicitly specified, so it must decide its action while considering the goals of other agents. In our setting, the initial state is as in Figure 6 (left), where each agent is placed at random coordinates and the goal position is set uniformly in the environment. When the time step is t, and the agent ID is i, each agent receives an observation o_t^i, which includes its absolute position, the relative positions of each goal, and the relative positions of other agents. Based on the policy $\boldsymbol{\mu}_i$, the agent determines its action a_t^i, specifying movement in the x and y directions. The reward for each agent is given by $r_i = -\sum_{j=0}^{N}(\min_{k\in I}||\mathbf{g}_{pos}^j - \mathbf{a}_{pos}^k||) - \textit{collision-penalty}$, where N represents the number of agents and the number of goals, $\mathbf{g}_{pos}$ denotes the position vector of a goal, $\mathbf{a}_{pos}$ denotes the position vector of an agent, and *collision-penalty* refers to the penalty reward assigned for agent collisions.

5.2 Settings and Parameters

The actor and critic networks consisted of three and four fully connected layers, respectively, with a unit

Algorithm 3: Update actor and critic function for ”Sharing Selected Observations and Actions”.

```
function UPDATE_AGENT($i$)
    Sample a random minibatch of $S$ samples $(\vec{o}^{\,j}, \vec{a}^{\,j}, \mathbf{r}^j, \vec{o}^{\,\prime j}, I_{select})$ from replay buffer $\mathcal{D}$
    $\vec{o}_i^{\,j} \leftarrow \{o_i^j, o_k^j\}_{k \in I_{i,\text{select}}}$
    $\vec{a}_i^{\,j} \leftarrow \{a_i^j, a_k^j\}_{k \in I_{i,\text{select}}}$
    $\vec{o}_i^{\,\prime j} \leftarrow \{o_i^{\prime j}, o_k^{\prime j}\}_{k \in I_{i,\text{select}}}$
    Set $y^j = r_i^j + \gamma Q_i^{\boldsymbol{\mu}'}(\vec{o}_i^{\,\prime j}, \vec{a}_i^{\,\prime j}) \mid_{a'_k = \boldsymbol{\mu}'_k(o'_k)}$
    Update critic by minimizing the loss:
        $\mathcal{L}(\theta_i) = \frac{1}{S}\sum_j \left(y^j - Q_i^{\boldsymbol{\mu}}(\vec{o}_i^{\,j}, \vec{a}_i^{\,j})\right)^2$
    Update actor using the sampled policy gradient:
        $\nabla_{\theta_i} J \approx \frac{1}{S}\sum_j \nabla_{\theta_i}\boldsymbol{\mu}_i(o_i^j)\nabla_{a_i} Q_i^{\boldsymbol{\mu}}(\vec{o}_i^{\,j}, \vec{a}_i^{\,j}) \mid_{a_i = \boldsymbol{\mu}_i(o_i^j)}$
```

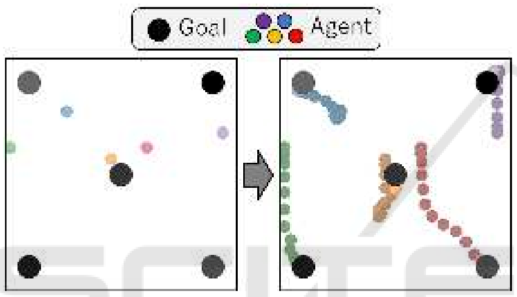

Figure 6: Cooperative navigation (5 agents).

size 64 for the hidden layer and an Adam optimizer with a learning rate of $\alpha = 0.01$ for updating the network and $\tau = 0.01$ for updating the target network. The discount factor $\gamma = 0.95$. The replay buffer size is 10^6, and the network parameters are updated after every 100 sample is added to the replay buffer. The batch size before updating is 1024 episodes, with one episode having a maximum of 25 steps, and the environment is updated every episode. Each agent has an independent network and does not share network parameters. These parameters are following MADDPG(Lowe et al., 2017).

5.3 Comparison of Learning Accuracy by Grade of Information Sharing

5.3.1 Experiment 1: Cooperative Navigation with Fully Observable

The results of the comparison of IDDPG, MADDPG, and the three proposed methods (Sharing Full-Observations: -fo, Sharing Full-Actions: -fa, and Sharing selected Observations& Actions: -soa) in the cooperative navigation task shown in Figure 6 for learning 100,000 episodes (about 2.5M steps) are shown in Figure 7 (top). The horizontal axis shows the learning progress [steps], the vertical axis shows the task completion rate [%], the solid graph shows the average of the 31 trials, and the band shows the standard deviation. Figure 7 (bottom) shows the boxplots of the 31 trials, and Table 2 shows the Mann-Whitney U test results for MADDPG and each method. In addition, Figure 7 (bottom) shows the boxplots of average task completion rates for the evaluation experiments using the learned model for all 31 trials, and Table 2 shows the results of the Mann-Whitney U-tests for MADDPG and each method. Evaluation experiments were conducted on the latest 20% of the learned models (a total of 20 models from 80,000 to 100,000 episodes).

Figure 7, Table 2 shows that MADDPG-fa has the highest achievement rate at 73.63 ± 9.51[%], followed by MADDPG-soa (71.58 ± 17.42), MADDPG (63.89 ± 12.28), IDDPG (44.58 ± 13.31) and MADDPG-fo (36.89 ± 21.56). These results indicate that sharing the observations of other agents in an environment where complete observations are available creates redundancy and leads to undesirable results (MADDPG-fa > MADDPG and IDDPG > MADDPG-fo). The comparison of MADDPG and MADDPG-soa may serve as a basis for estimating the impact of reduced shared information. In this experiment, the performance of MADDPG-soa, where shared information is limited, was statistically significantly better than that of MADDPG. These results indicate that unnecessary information sharing has detrimental effects on performance and that performance gains can be achieved through appropriate selection. Specifically, in a fully observable environment, sharing observational information from other agents introduces redundancy and potentially degrades performance.

5.3.2 Experiment 2: Cooperative Navigation with Partially Observable

A fully observable environment is unrealistic in practical scenarios. Since the lack of observation may alter the prioritization of additional information, a similar experiment was conducted by extending the cooperative navigation task shown in Figure 1 to a partially observable environment. In this experiment, partial observability is represented by restricting the observable information to the relative coordinates of two adjacent agents. The results of the comparison of IDDPG, MADDPG, and the three proposed methods in the same setting as in Experiment 1 are shown in Figure 8 (top). The horizontal axis shows the learning progress [steps], the vertical axis shows the task completion rate [%], the solid graph shows the average of

Table 2: Mean success rate (standard deviation) and Mann-Whitney U Test results.

	MADDPG	IDDPG	MADDPG-fo	MADDPG-fa	MADDPG-soa
Exp. 1	63.89(12.28)	**44.58(13.31)	**36.89(21.56)	****73.63**(9.51)	*71.58(17.42)
Exp. 2	70.11(21.58)	**51.02(12.38)	67.7(20.78)	**51.88(18.28)	**74.99**(17.51)

*: $p < .05$, **: $p < .01$ (Mann-Whitney U)

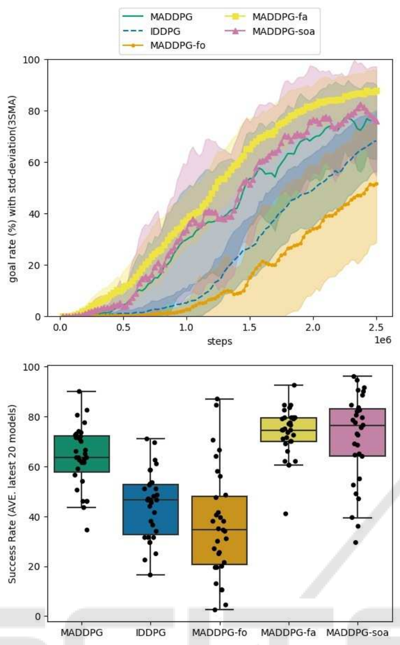

Figure 7: Results of 5 agents cooperative navigation with the fully observable setting. *Top:* Learning progress and success rate. *Bottom:* Distribution of task success rates in evaluation experiments.

the 31 trials, and the band shows the standard deviation. Figure 8 shows the boxplots of average task completion rates for the evaluation experiments using the learned model for all 31 trials, and Table 2 shows the results of the Mann-Whitney U-tests for MADDPG and each method. Evaluation experiments were conducted on the latest 20% of the learned models (a total of 30 models from 120,000 to 150,000 episodes).

Figure 8, Table 2 shows that MADDPG-soa had the highest achievement rate at $74.99 \pm 17.51[\%]$, followed by MADDPG (70.11 ± 21.58), MADDPG-fo (67.7 ± 20.78), MADDPG-fa (51.88 ± 18.28) and IDDPG (51.02 ± 12.38). These results indicate that in one of the partially observable environments, observational information has a higher priority for sharing than action information (MADDPG-fo ¿ MADDPG-fa). This finding contrasts with the results of Experiment 1 and highlights the importance of addressing the issue of partial observability. Nonetheless, the superior performance of MADDPG and MADDPG-soa over MADDPG-fo clearly demonstrates that sharing behavioral information remains effective even in partially observable environments. As in Experiment 1, the comparison between MADDPG and MADDPG-soa in Experiment 2 further suggests that selective information sharing can enhance accuracy.

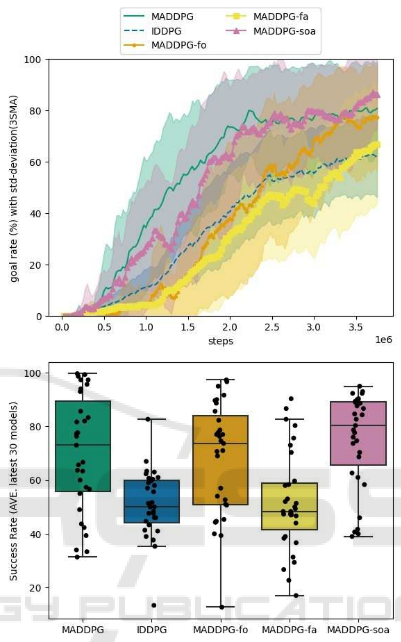

Figure 8: Results of 5 agents cooperative navigation with the partially observable setting. *Top:* Learning progress and success rate. *Bottom:* Distribution of task success rates in evaluation experiments.

6 CONCLUSIONS

This paper extends MADDPG, a MARL algorithm with CTDE, to propose three methods, i.e., MADDPG with sharing (i) information on actions of all agents, (ii) information on observations of all agents, and (iii) information on both the actions and observations of the selected agents to investigate the extent of information sharing in CTDE comprehensively. To our knowledge, the degree of information sharing varies by task setting and application destination. Since no exhaustive survey has been conducted, we believe this validation is essential. Experimental results show that selective additional information reduction can maintain learning accuracy, especially when sharing actions of all agents, and is best performed in a fully observable environment. We also showed cases where redundant additional information worsens learning accuracy and identified priorities for additional information in fully and partially observable environments.

In this paper, experiments were conducted with additional information limited to agent observations and actions. Future research should consider applying this approach to other types of additional information. Furthermore, the information selection method in this study was defined at runtime and remained fixed throughout the learning process. Given the complexity of multi-agent reinforcement learning (MARL), it is likely that the critical information may vary depending on the learning stage. Therefore, dynamic selection based on the progress of learning would be a valuable direction for future work.

REFERENCES

Foerster, J., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. (2018). Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Gronauer, S. and Diepold, K. (2022). Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review*, 55(2):895–943.

Hansen, E. A., Bernstein, D. S., and Zilberstein, S. (2004). Dynamic programming for partially observable stochastic games. In *AAAI*, volume 4, pages 709–715.

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

Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier.

Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Pieter Abbeel, O., and Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30.

Rashid, T., Samvelyan, M., De Witt, C. S., Farquhar, G., Foerster, J., and Whiteson, S. (2020). Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178):1–51.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Son, K., Kim, D., Kang, W. J., Hostallero, D. E., and Yi, Y. (2019). Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International conference on machine learning*, pages 5887–5896. PMLR.

Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., and Graepel, T. (2018). Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '18, pages 2085–2087, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Wolpert, D. H. and Tumer, K. (2001). Optimal payoff functions for members of collectives. *Advances in Complex Systems*, 4(02n03):265–279.

Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., and Wu, Y. (2022). The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, 35:24611–24624.

Zhang, K., Yang, Z., and Başar, T. (2021). Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of Reinforcement Learning and Control*, pages 321–384.