Efficient Multi-Agent Communication via Shapley Message Value

Di Xue[1,*] , **Lei Yuan**[1,2,*] , **Zongzhang Zhang**[1,†] and **Yang Yu**[1,3]
[1]State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
[2]Polixir Technologies, Nanjing 210000, China
[3]Peng Cheng Laboratory, Shenzhen, Guangdong, China
{xued, yuanl}@lamda.nju.edu.cn, {zzzhang, yuy}@nju.edu.cn

Abstract

Utilizing messages from teammates is crucial in cooperative multi-agent tasks due to the partially observable nature of the environment. Naively asking messages from all teammates without pruning may confuse individual agents, hindering the learning process and impairing the whole system's performance. Most previous work either utilizes a gate or employs an attention mechanism to extract relatively important messages. However, they do not explicitly evaluate each message's value, failing to learn an efficient communication protocol in more complex scenarios. To tackle this issue, we model the teammates of an agent as a message coalition and calculate the Shapley Message Value (SMV) of each agent within it. SMV reflects the contribution of each message to an agent, and redundant messages can be spotted in this way effectively. On top of that, we design a novel framework named Shapley Message Selector (SMS), which learns to predict the SMVs of teammates for an agent solely based on local information so that the agent can only query those teammates with positive SMVs. Empirically, we demonstrate that our method can prune redundant messages and achieve comparable or better performance in various multi-agent cooperative scenarios than full communication settings and existing strong baselines.

1 Introduction

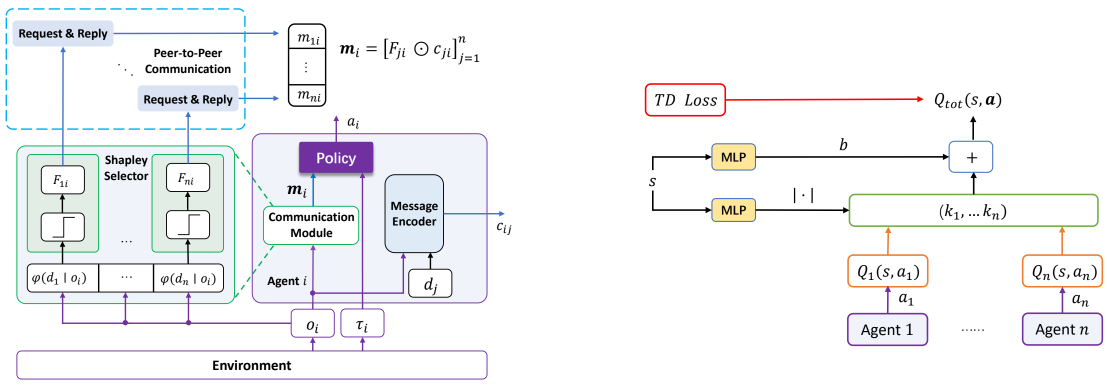
Cooperative Multi-Agent Reinforcement Learning (MARL) [Gronauer and Diepold, 2021] aims at coordinating multiple agents towards a shared goal, showing great potential to solve real-world problems. Most recent work on MARL adopts the *Centralized Training and Decentralized Execution* (CTDE) [Kraemer and Banerjee, 2016] paradigm to deal with non-stationarity caused by the concurrent learning of multiple policies. However, the coordination ability of the learned policies under CTDE can be fragile in the face of partial observability.

[*]Equal contribution.
[†]Corresponding author: Zongzhang Zhang.

Communication is a promising way to address the mentioned issue in MARL. It allows agents to exchange information, share experiences and coordinate better with teammates. To accomplish this, researchers have considered the differentiable communication channel under the CTDE paradigm. Many methods in this direction utilize a broadcast-like communication channel where the sent message is delivered to all agents. They employ techniques such as gate mechanisms to control when to send messages [Singh *et al.*, 2019; Mao *et al.*, 2020], or attention mechanisms [Das *et al.*, 2019] to extract useful information from all received messages. However, these methods treat messages as black boxes and tacitly assume that deep neural networks can identify valuable ones. As the number of agents increases, the communication channel would soon be overwhelmed with irrelevant messages, and hence communication can barely help.

Learning whom to communicate with is one of the salient properties of humans, by which we could realize more efficient communication. Rather than the "receive-then-select" approach widely used in previous work [Das *et al.*, 2019], or pruning messages according to a learned rule [Wang *et al.*, 2020b; Ding *et al.*, 2020; Niu *et al.*, 2021], it can be more efficient if we can evaluate the contribution of each message, then ask for messages only from those who benefit us. For example, in our daily research life, we can build a specific belief for each member in our lab during everyday interaction, such as expertise in different domains. When we do a project, we can decide whom to collaborate with according to their expertise to obtain the best guidance and work smoothly.

Taking inspiration from the above insight, this paper proposes a novel multi-agent communication framework that enables agents to evaluate the contribution of each message and decide whom to communicate with. Specifically, we model the teammates of an agent as a message coalition inspired by the cooperative game theory [Chalkiadakis *et al.*, 2011; Wang *et al.*, 2020a], and we calculate the Shapley Message Value (SMV) of each agent within the message coalition. The contribution of messages is captured by the SMV, by which we can distribute part of the individual utility to its teammates according to the contribution of the messages. The calculated SMVs are then used to train the Shapley Message Selector (SMS), which is equipped on each agent. During execution, the learned SMS can help an agent identify teammates whose messages can benefit its decision-making based



(a) Communication-Based Policy

(b) Linearly Decomposable Critic

Figure 1: Schematics of our proposed framework.

solely on local observation. We evaluate it on three cooperative multi-agent tasks, i.e., Listener-Speaker, Hallway [Wang *et al.*, 2020b], and StarCraft Multi-Agent Challenge (SMAC) [Samvelyan *et al.*, 2019]. Visualization and empirical results show that our method can prune redundant messages and achieve comparable or better performance in various multi-agent cooperative scenarios than full communication settings and existing strong baselines.

2 Background

We consider fully cooperative multi-agent tasks with partial observability modeled as Dec-POMDPs [Oliehoek and Amato, 2016]. A Dec-POMDP model can be formulated as $\mathcal{M} = (N, S, A, P, \Omega, O, R, \gamma)$, where $N = \{1, 2, \dots, n\}$ represents the set of n agents, S is the set of states, and $A = \prod_{i=1}^n A_i$ is the joint action space. At each timestep, agent $i \in N$ receives a local observation $o_i \in \Omega$ drawn by the observation function $O(s, i)$, and then chooses an action $a_i \in A_i$. The joint action $\boldsymbol{a} = (a_1, \dots, a_n)$ leads to the shared global reward $R(s, \boldsymbol{a})$ and the next state $s' \sim P(s' \mid s, \boldsymbol{a})$. The formal objective is to find a joint policy $\boldsymbol{\pi}(\boldsymbol{a} \mid \boldsymbol{\tau})$ to maximize the global value function $Q^{\boldsymbol{\pi}}_{\text{tot}}(s, \boldsymbol{a}) = \mathbb{E}_{s_{0:\infty}, \boldsymbol{a}_{0:\infty}}[\sum_{t=0}^{\infty} \gamma^t R(s, \boldsymbol{a}) \mid s_0 = s, \boldsymbol{a_0} = \boldsymbol{a}, \boldsymbol{\pi}]$, where $\boldsymbol{\tau} = (\tau_1, \dots, \tau_n)$ is the joint history of all agents, and $\gamma \in [0, 1)$ is the discount factor.

Shapley value is a popular solution concept to solve the payoff distribution problem for a grand coalition [Chalkiadakis *et al.*, 2011], and the advantage is that it provides a fair and unique solution [Fatima *et al.*, 2008]. A cooperative game can be defined as $G = (N, v)$, where $N = \{1, 2, \dots, n\}$ is a finite, non-empty set of players and $v : 2^N \to \mathbb{R}$ is a characteristic function assigning each coalition $C \subseteq N$ of players a value $v(C)$ representing the total payoff that can be obtained by the coalition C. The Shapley value of the ith player with respect to the grand coalition N is defined as

$$\phi^G(i) = \sum\nolimits_{C \subseteq N \setminus \{i\}} \frac{|C|!(|N| - |C| - 1)!}{|N|!} \Phi_i^G(C), \quad (1)$$

where $\Phi_i^G(C) = v(C \cup \{i\}) - v(C)$ is called the marginal contribution, which can measure how much value is attributed to the participation of the ith player to the coalition C. Intuitively, the Shapley value is the average marginal contribution over all possible coalitions.

3 Method

In this section, we first introduce our multi-agent communication framework. Then we formulate multi-agent communication as cooperative games and define SMV, which extends the concepts of the grand coalition and Shapley value. After that, we propose a method that learns to predict the SMVs of teammates and decide whether to ask for messages accordingly. We finally present the overall training scheme.

3.1 Overall Framework

As depicted in Figure 1, our framework aims to learn a peer-to-peer communication scheme under the CTDE paradigm. It is composed of two main parts, a communication-based policy (Figure 1(a)) equipped with a communication module that learns from whom to ask for information, and a linearly decomposable centralized critic (Figure 1(b)) that can be used to evaluate the contribution of each message and provide the training signal for the communication module of the policy. During execution, the centralized critic will be removed, and agents utilize the learned SMS to communicate with teammates who are believed to be beneficial to them.

At each timestep, agent i decides whom to communicate with before taking action. To be specific, agent i applies its SMS to the current observation to predict the SMVs of teammates, which gives $[\varphi(d_j \mid o_i)]_{j=1}^n$ and d_j is the ID of agent j. The predicted SMVs are then transformed into an n-dimension binary vector $[F_{ji}]_{j=1}^n$ by an element-wise indicator function $\mathbb{1}_{[>0]}(\cdot)$ that assigns one to positive numbers. F_{ji} represents whether to ask for information from agent j. If $F_{ji} = 1$, agent i would send a request for information to agent j. Otherwise, the communication link is pruned. We set $F_{ii} = 0$ to avoid circulate communication. Upon receiving a request, agent j would reply with a message $c_{ji} = g_j(o_j, d_i)$ through its message encoder, an encoding of its local observation. The messages received by agent i are gathered into $\boldsymbol{m}_i = [F_{ji} \odot c_{ji}]_{j=1}^n$, which is then fed into its local policy π_i

along with its local action-observation history τ_i to generate an action $a_i = \pi_i(\tau_i, \boldsymbol{m}_i)$.

To fulfill the purpose of considering contributions of messages to local decision-making, the design of centralized critic should be taken extra care. Popular policy-based methods, e.g., MADDPG [Lowe *et al.*, 2017] and COMA [Foerster *et al.*, 2018], adopt non-factorizable centralized critics that conditions on global state and joint action. These methods either suffer from inefficient credit assignment [Wang *et al.*, 2021] or high variance [Papoudakis *et al.*, 2021], making them not good choices to evaluate individual contribution. Value factorization methods like QMIX [Rashid *et al.*, 2018] decompose centralized value into individual utilities in a nonlinear manner and destroy the consistency between centralized value and individual utilities. To make individual critics on the same scale as the centralized critic, we employ a linearly decomposed centralized critic as in DOP [Wang *et al.*, 2021]:

$$Q_{\text{tot}}(s, \boldsymbol{a}) = \sum\nolimits_{i=1}^{n} k_i(s) Q_i(s, a_i) + b(s)\,, \tag{2}$$

where $k_i(s)$ and $b(s)$ are generated by learnable networks whose inputs are global state, and $k_i(s)$ is restricted to be non-negative to ensure monotonicity. Since the utilities of agents are additive after being scaled by their coefficients, we can evaluate the value made by each agent to the whole team.

3.2 Communication as Cooperative Games

An advantage of our framework is that it allows us the flexibility to quantitatively examine the contribution of a message to overall decision-making. Suppose there are two agents i and j working together, depending on whether the message from the agent j is received, agent i may take different actions: if agent i queries agent j, agent i would choose $a_i^j = \pi_i(\tau_i, [c_{ji}, \mathbf{0}])$, otherwise it would choose $a_i = \pi_i(\tau_i, [\mathbf{0}, \mathbf{0}])$. Then the contribution of agent j made to agent i is

$$\Phi_j^i = k_i(s)(Q_i(s, a_i^j) - Q_i(s, a_i))\,, \tag{3}$$

where we scale the individual utility by its coefficient so that the scale is consistent with the global value. Recall that the definition of the marginal contribution in Section 2 is similar to how we calculate the contribution of messages to local decision-making here. Now we are at the point to generalize the related concepts in cooperative game theory to our setting.

Message Coalition: The action of each agent is determined by its local information, as well as the messages received from teammates. Then it is natural to attribute part of the utility to other agents so that the importance of each message can be measured. To distribute agent i's utility to other agents, we model it as a cooperative game (C_i, v_i), where the grand coalition is the set of all agents but for agent i, i.e., $N \setminus \{i\}$, which is named as the message coalition C_i of i. Likewise, the characteristic function is defined on each subset of the message coalition $C \subseteq C_i$:

$$v_i(C) = k_i(s)(Q_i(s, a_i^C) - Q_i(s, a_i))\,, \tag{4}$$

where $a_i^C = \pi_i(\tau_i, m_i^C)$ and $m_i^C = [\mathbb{1}_{\in C}(j) \odot c_{ji}]_{j=1}^n$ contains the messages received from agents in C. In this cooperative game, we are actually distributing the value $v_i(C_i)$ earned by the joint messages from message coalition to each member within it.

Shapley Message Value: To solve the cooperative game in our setting (C_i, v_i), we recall the definition of the Shapley value. The generalized definition for the marginal contribution of the message from the jth agent to a coalition $C \subseteq C_i \setminus \{j\}$ is as below:

$$\begin{aligned}\Phi_j^i(C \mid s) &= v_i(C \cup \{j\}) - v_i(C) \\ &= k_i(s)(Q_i(s, a_i^{C\cup\{j\}}) - Q_i(s, a_i^C))\,.\end{aligned} \tag{5}$$

Then, we compute the Shapley value of agent j's message:

$$\begin{aligned}\phi^i(d_j \mid s) &= \sum_{C \subseteq C_i \setminus \{j\}} \frac{|C|!(|C_i| - |C| - 1)!}{|C_i|!} \Phi_j^i(C \mid s) \\ &= \mathbb{E}_{p(C|C_i\setminus\{j\})}[\Phi_j^i(C \mid s)]\,,\end{aligned} \tag{6}$$

which is called Shapley Message Value (SMV) in our paper. The coefficients of $\Phi_j^i(C \mid s)$ in Eq. 6 sum to one, implying a probabilistic interpretation and we make it explicit by introducing a notation $p(C \mid C_i \setminus \{j\})$. SMV measures the contribution of messages to the recipient agent. Intuitively, we hope to leave those messages having positive effects on local decision-making, i.e., messages with positive SMV, and prune those having negative effects.

Since each agent can be seen as a recipient agent, there are n such cooperative games existing at the same time. By selecting messages with positive SMV, each agent is expected to gain a greater local value, which is further combined monotonically by the mixer, resulting in a greater total value.

3.3 Shapley Message Selector

SMV serves as a criterion for selecting those messages having positive effects on local decision-making. However, the problem is that we cannot evaluate the SMV of an agent unless we know the global state and receive all the messages, making it infeasible to apply during decentralized execution. Therefore, we design a practical method to predict the SMV solely based on local information.

The most straightforward way is to do a regression. Here we use a neural network named Shapley Message Selector (SMS) to approximate the SMVs of teammates within its message coalition through supervised learning. The SMS takes local observation as input, and the output is a vector of length n whose jth component is the predicted SMV $\varphi^i(d_j \mid o_i)$ of agent j among message coalition C_i. For simplicity of notation, we define the SMV of an agent to itself to be zero.

Fortunately, global state and local information are correlated probabilistically. Thus we can exploit such correlation to approximate SMV via

$$\varphi^i(d_j \mid o_i) = \mathbb{E}_{p(o_i|s)}[\phi^i(d_j \mid s)]\,, \tag{7}$$

where $p(o_i \mid s)$ is induced by the current joint policy. This suggests we use on-policy trajectories to generate the dataset to train the SMS. During centralized training, we can calculate the SMV to generate the training data. To be specific, for agent i, the local observation o_i is taken as the feature and the SMVs of each agent $[\phi^i(d_j \mid s)]_{j=1}^n$ are taken as the label, forming a feature-label tuple $\langle o_i, [\phi^i(d_j \mid s)]_{j=1}^n \rangle$.

During decentralized execution, the SMS behaves like a gate. Given current observation o_i, agent i would predict the

SMV of each teammate. If the predicted SMV of agent j is positive, meaning that the information from agent j can benefit the local decision-making of agent i in expectation, agent i would send a request for information to agent j.

3.4 Training Scheme

During centralized training the parameters of the critic are updated by the standard TD loss of the global value Q_{tot}. To improve sample efficiency, we utilize both on-policy and off-policy data by minimizing the following loss:

$$\mathcal{L}(\theta_Q) = \frac{1}{2}\mathcal{L}^{\text{On}}(\theta_Q) + \frac{1}{2}\mathcal{L}^{\text{Off}}(\theta_Q)\,, \tag{8}$$

where θ_Q is the parameter of the centralized critic, and $\mathcal{L}^{\text{On}}$ and $\mathcal{L}^{\text{Off}}$ are evaluated on the on-policy and off-policy replay buffers $\mathcal{D}^{\text{On}}$ and $\mathcal{D}^{\text{Off}}$, respectively. The first loss item is $\mathcal{L}^{\text{On}}(\theta_Q) = \mathbb{E}_{\mathcal{D}_{\text{on}}}[(y^{\text{On}} - Q_{\text{tot}}(s^t, \boldsymbol{a}^t; \theta_Q))^2]$, where y^{On} is the target value defined as $r^t + \gamma Q_{\text{tot}}(s^{t+1}, \boldsymbol{a}^{t+1}; \theta_Q^-)$, and θ_Q^- is the parameter of the target network periodically copied from θ_Q. The second loss item is $\mathcal{L}^{\text{Off}}(\theta_Q) = \mathbb{E}_{\mathcal{D}^{\text{Off}}}[(y^{\text{Off}} - Q_{\text{tot}}(s^t, \boldsymbol{a}^t; \theta_Q^-))]$, the action for the next timestep is corrected by the current policy, message encoder and SMS:

$$y^{\text{Off}} = r^t + \gamma Q_{\text{tot}}(s^{t+1}, \pi_i(s^{t+1}, \boldsymbol{m}_i; \theta_\pi^-); \theta_Q^-)\,. \tag{9}$$

Here, $\boldsymbol{m}_i = \{\mathbb{1}_{[>0]}(\varphi_i(d_j \mid o_i^{t+1}; \theta_{\varphi_i}^-)) \cdot g_j(o_j^{t+1}, d_i; \theta_{g_j}^-)\}_{j=1}^n$ is the relabeled message. The policy and message encoder are jointly trained to maximize the centralized critic Q_{tot}:

$$\mathcal{J}(\theta_\pi, \theta_g) = \mathbb{E}_{\mathcal{D}_{\text{On}}}\left[\sum_{i=1}^n k_i(s) Q_i(s, \pi_i(\tau_i, \boldsymbol{m}_i; \theta_\pi))\right], \tag{10}$$

where $s \sim \mathcal{D}_{\text{On}}$, Q_{tot} is decomposed into individual critics, and the bias can be left out because it is not dependent on the action. To deal with discrete action space, we apply the Gumbel-Softmax trick [Jang *et al.*, 2016] to reparameterize the stochastic policies as deterministic functions. In order to calculate SMV, we have to evaluate the value of messages from the message coalition by $Q_i(s, a_i^C)$, where $C \subseteq C_i$. Since the policy is modeled by a neural network, extra care should be taken to avoid the possible correlation between messages so that each message can be useful on its own. To this end, we train the policy by dropping each message with probability p_{drop} which is a hyper-parameter, just like the dropout technique [Srivastava *et al.*, 2014].

The SMS $\varphi^i(d_j \mid o_i)$ is parameterized by θ_{φ_i} and is trained on the tuple $\langle o_i^t, [\phi^i(d_j \mid s^t)]_{j=1}^n \rangle$ generated from recent trajectories by minimizing the following mean squared error:

$$\mathcal{L}(\theta_{\varphi^i}) = \frac{1}{nT}\sum_{t=1}^T\sum_{j=1}^n (\varphi^i(d_j \mid o_i^t; \theta_{\varphi^i}) - \phi^i(d_j \mid s^t))^2\,. \tag{11}$$

The exact calculation of SMV $\phi^i(d_j \mid s^t)$ is intractable as its complexity goes up exponentially, but we can approximate it according to its probabilistic interpretation given in Eq. 6:

$$\phi^i(d_j \mid s^t) = \mathbb{E}_{p(C|C_i\backslash\{j\})}[\Phi_j^i(C|s^t)] \approx \frac{1}{H}\sum_{k=1}^H \Phi_j^i(C^k|s^t), \tag{12}$$

where $C^k \sim p(C \mid C_i\backslash\{j\})$ and H is the number of samples.

Recall that the SMV $\phi^i(d_j \mid s)$ is calculated from the individual critic $Q_i(s, a_i)$, which is modeled by a trainable neural network. This means that the quality of the SMS is highly relied on the accuracy of $Q_i(s, a_i)$. Thus we do not apply the SMS in execution until t_{selector} timesteps have passed. Interested readers may refer to our pseudo-code in Appendix C.

4 Experiments

In this section, we design experiments to verify that SMS can learn an efficient and targeted communication protocol in multi-agent tasks[1]. We compare SMS with multiple baselines on cooperative tasks, including Listener-Speaker, Hallway [Wang *et al.*, 2020b], and the challenging StarCraft II unit micromanagement benchmark[2] [Samvelyan *et al.*, 2019]. For evaluation, all results are illustrated with median performance with 95% confidence interval on 5 random seeds. Detailed network architecture and hyper-parameter choices are shown in Appendix B.

4.1 Baselines

QMIX [Rashid *et al.*, 2018] and DOP [Wang *et al.*, 2021] are strong non-communication baselines. QMIX is a value-based baseline, the implementation we compare is provided by PyMARL2, which has shown excellent performance on diverse multi-agent benchmarks [Hu *et al.*, 2021]. DOP, on the other hand, is a policy-based method that employs a linearly decomposable mixing network similar to our method.

TarMAC [Das *et al.*, 2019], NDQ [Wang *et al.*, 2020b] and MAGIC [Niu *et al.*, 2021] are state-of-the-art communication baselines. TarMAC utilizes an attention mechanism to integrate messages according to their relative importance. The implementation we use is provided by [Wang *et al.*, 2020b], denoted as QMIX+TarMAC. NDQ aims at learning nearly decomposable Q functions via communication minimization. MAGIC makes use of hard attention to construct a dynamic communication graph, which then combines with a graph attention neural network to process the messages.

For the ablation study, we design a baseline with only difference in the communication protocol. It adopts a full communication paradigm, where each agent queries all other teammates at each timestep, denoted as FullComm.

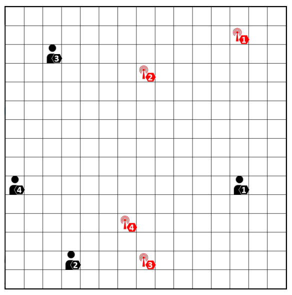
4.2 Listener-Speaker

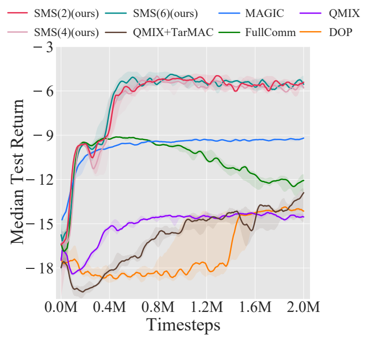
We design a listener-speaker environment (Figure 2(a)) to demonstrate (1) the existence of redundant messages that may harm the performance of learning algorithms, and (2) SMS can recognize and prune those redundant messages.

Listener-speaker is a grid world consisting of n listeners and n speakers randomly set at the beginning of each episode. Listeners and speakers are one-to-one paired, and each pair has the same number attached to them. The listeners must navigate to their paired speakers and the reward is calculated as $r^t = -0.01\sum_{i=1}^n \text{dist}(l_i^t, s_i^t)$, where $\text{dist}(l_i^t, s_i^t)$ is the Euclidean distance between the ith listener-speaker pair at

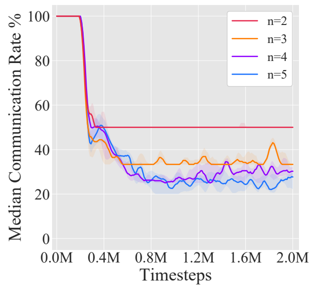
[1]Code available at https://github.com/DiXue98/SMS.

[2]The results reported in this paper use SC2.4.6.2.6923. Note that performance is not always comparable between versions.

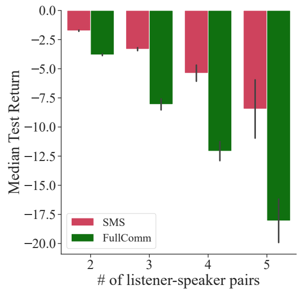

(a) Listener-Speaker



(b) Performance comparison



(c) Communication rate



(d) Performance across agents

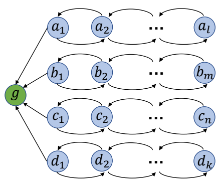
Figure 2: (a) The Listener-Speaker environment, where listeners are marked in black and speakers are marked in red. (b) Median test returns of different algorithms. (c) Communication rates under different listener-speaker pairs. (d) Median test returns with different number of listener-speaker pairs after 2M timesteps. The error bar represents the standard deviation across 5 random seeds.

timestep t. While the listener cannot observe anything l
its ID, the speaker knows its relative position to each listen
To succeed in such a task, the listener has to learn to figu
out from which speaker to ask for information, so that it c
move towards its paired speaker and won't be distracted
irrelevant information. Although this problem is seeming
simple, it poses a significant challenge to full communicati
algorithms even when n is small. We ignore NDQ in tl
environment as the behavior of the speaker is fixed.

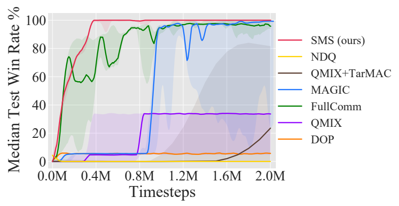
We set $n = 4$ and the map size to be 15×15. We a
set $t_{\text{selector}} = 0.2$M, which is the timestep when SMS
activated. As shown in Figure 2(b), the methods without communication, i.e., QMIX and DOP, fail in this task and perform worst, which indicates communication is necessary. TarMAC mitigates the redundancy to some extent, but its attention mechanism is inefficient, leading to its slow convergence speed. SMS behaves similarly to FullComm and MAGIC at the beginning but successfully outperforms all baselines to a large margin after enabling SMS at 0.2M steps. The ablation of SMS, i.e., FullComm, behaves almost the same as our method before $0.4M$ but deteriorates after that. We think it is because its policy overfits to the redundant messages of irrelevant speakers, converging to sub-optimal policy.

Figure 2(b) also reveals the effects of different sample sizes H (2, 4, 6) used to approximate the SMV on the performance. As the sample size grows, we have a little faster convergence speed, indicating a more accurate SMV estimation. A large sample size also costs more computational resources. We finally set the sample size $H = 2$ in the following experiments for a computational complexity and performance trade-off.

Communication Rate Analysis. To answer why SMS can effectively deal with redundancy, we plot the communication rate of SMS with the number of listener-speaker pairs ranging from 2 to 5 in Figure 2(c). Since listeners and speakers are one-to-one paired, for each listener there is only one speaker that can help it reach its destination. Thus the optimal communication rate is $1/n$. The learned SMS can almost achieve optimal communication rates in these scenarios, although the variance increases as N grows. Meanwhile, we can find from Figure 2(d) that SMS outperforms FullComm in all the scenarios, and the return of SMS changes nearly proportionally to the number of listener-speaker pairs, indicating SMS can effectively identify the paired speaker for each listener.



(a) Hallway



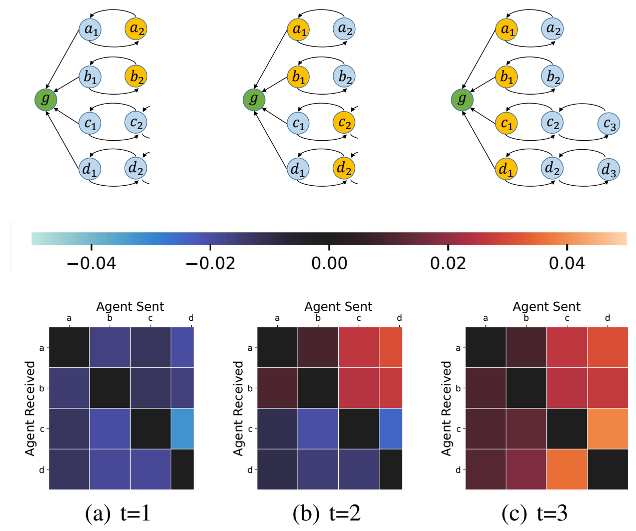
(b) Test median success rate

Figure 3: (a) Hallway, where agents aim to reach g simultaneously. (b) Test median success rates on Hallway.

4.3 Hallway

Hallway [Wang *et al.*, 2020b] (Figure 3(a)) is another cooperative environment under partial observability, with four agents a, b, c, and d randomly initialized at states a_1 to a_l, b_1 to b_m, c_1 to c_n, and d_1 to d_k, respectively. An agent can only observe its position and select actions from moving left, moving right, or keeping immobile at each timestep. An episode ends if any agent arrives at state g, and agents will win and get a shared reward of 1 only when they reach state g simultaneously, otherwise the reward is 0. The horizon is set to $\max(l, m, n, k) + 10$ to avoid infinite loops.

We set $l = m = 2$ and $n = k = 3$ to make the simultaneous arrival difficult. For SMS we set $t_{\text{selector}} = 0.3$M. The performance is shown in Figure 3(b), where we find QMIX and DOP fail again, indicating communication is a requirement to achieve effective coordination in this task. Other communication methods, i.e., TarMAC and NDQ, can measure the relative importance of messages but are not efficient enough. MAGIC can solve it but takes much more timesteps to converge than SMS. FullComm performs well, but the curve oscillates a lot and does not converge to the same level as SMS. The redundancy in the message can be validated by the



(a) t=1 (b) t=2 (c) t=3

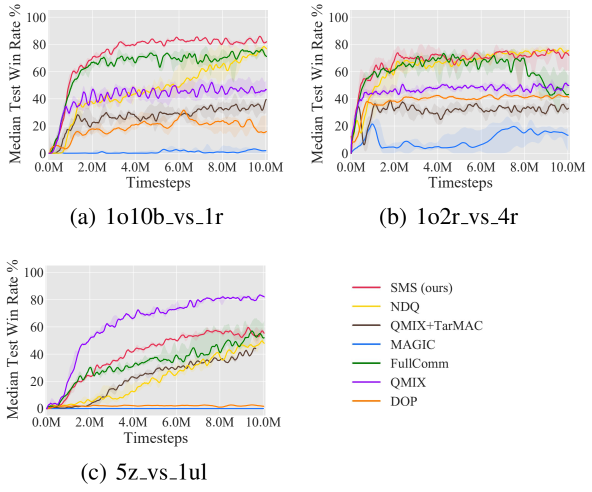
Figure 4: The SMV distributions learned by our method on Hallway. A blue square in the heat map means that the corresponding message has negative SMV and is pruned.

gap between the curves of SMS and FullComm.

The Communication Protocol Learned by SMS. We are particularly interested in how the learned SMS behaves at each timestep. To this end, we replay an episode in Figure 4 to demonstrate what happened at each timestep. The top row shows the snapshot of the global state under each timestep, while the corresponding output of the SMS is represented by a heat map in the bottom row. The ith row of the heat map is the output of the SMS of the ith agent, i.e., the predicted SMV for its teammates. At the first timestep ($t = 1$), the heat map is blue everywhere and all the SMVs are less than zero, indicating no communication among agents. This is because they are far from the state g, and all agents should move left regardless of the positions of their teammates. As the episode progresses ($t = 2$), some agents reach positions next to g, and this is where the coordination happens: agents (a and b) next to the goal g should wait for the ones (c and d) still two more steps away from g. This explains why the SMS of agents a and b have positive SMVs for other agents. In order to win the game when all agents are next to the goal state g ($t = 3$), they need to make sure that the other teammates are in the same position so that they can take actions simultaneously. This is also implied by the heat map. Besides, it is interesting to discover that agents tend to have a larger SMV for c and d than a and b. We believe this is because agents c and d are further from the state g on average, hence their information is more critical to the success of the whole team.

4.4 StarCraft II Micromanagement Benchmark

Finally, we verify whether SMS can handle more challenging scenarios on StarCraft Multi-Agent Challenge (SMAC) [Samvelyan *et al.*, 2019]. We test our method on three maps, including 1o10b_vs_1r and 1o2r_vs_4r, which are difficult due to partial observability of the environment, and 5z_vs_1ul,



(a) 1o10b_vs_1r (b) 1o2r_vs_4r (c) 5z_vs_1ul

Figure 5: Median test win rates on three SMAC maps.

where strong coordination is needed to succeed[3].

Since these tasks are more complex, we need more time to train the SMS before activation, thus we set $t_{\text{selector}} = 1.0$M. The results are shown in Figure 5. We find that 1o10b_vs_1r and 1o2r_vs_4r need strong communication to succeed. Methods without communication, such as QMIX and DOP, result in sub-optimal policies. SMS achieves a faster convergence speed than NDQ and higher asymptotic performance than other baselines, demonstrating that SMS can efficiently help agents identify helpful information among teammates. Scenario 5z_vs_1ul needs strong coordination. SMS outperforms all other baselines except QMIX as the problem of partial observability is not so severe on the map. MAGIC fails to maintain its decent performance in all the scenarios. We think it is largely due to the issues of credit assignment and sample efficiency are not properly handled.

5 Conclusions and Future Work

In this paper, we have shown the existence of redundancy in multi-agent communication and how it complicates the learning of communication protocol. Due to this issue, our work models multi-agent communication as cooperative games, and proposes Shapley Message Value (SMV) which extends Shapley value to quantitatively examine the value of each message. We also design a practical method to predict the value of messages based solely on local information.

The calculation of SMV is based on the individual critic, which implies that the quality of SMV highly relies on the accuracy of the individual critic. However, the current credit assignment methods do not work well and are not explainable enough [Wang *et al.*, 2020a], which may deteriorate the quality of SMV in more complex environments. A more precise method to calculate SMV and its theoretical support for multi-agent communication would be of great interest.

[3]The numbers of controlled agents on 1o10b_vs_1r, 1o2r_vs_4r and 5z_vs_1ul are 11, 3, and 5 respectively.

Acknowledgments

We would like to thank Quan He, Xue-Kun Jin, Yu Fan, Yun-Tian Zhang, and the anonymous reviewers for their helpful discussions and support. This work is supported by National Key Research and Development Program of China (2020AAA0107200), NSFC (61876077, 61876119), the Major Key Project of PCL (PCL2021A12), and Hikvision Open Fund (CCF-HIKVISION OF 20210001).

References

[Chalkiadakis *et al.*, 2011] Georgios Chalkiadakis, Edith Elkind, and Michael Wooldridge. Computational aspects of cooperative game theory. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 5(6):1–168, 2011.

[Das *et al.*, 2019] Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. TarMAC: Targeted multi-agent communication. In *ICML*, pages 1538–1546, 2019.

[Ding *et al.*, 2020] Ziluo Ding, Tiejun Huang, and Zongqing Lu. Learning individually inferred communication for multi-agent cooperation. In *NeurIPS*, pages 22069–22079, 2020.

[Fatima *et al.*, 2008] Shaheen S Fatima, Michael Wooldridge, and Nicholas R Jennings. A linear approximation method for the Shapley value. *Artificial Intelligence*, 172(14):1673–1699, 2008.

[Foerster *et al.*, 2018] Jakob N. Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *AAAI*, pages 2974–2982, 2018.

[Gronauer and Diepold, 2021] Sven Gronauer and Klaus Diepold. Multi-agent deep reinforcement learning: A survey. *Artificial Intelligence Review*, pages 1–49, 2021.

[Hu *et al.*, 2021] Jian Hu, Siyang Jiang, Seth Austin Harding, Haibin Wu, and Shih-wei Liao. RIIT: Rethinking the importance of implementation tricks in multi-agent reinforcement learning. *arXiv preprint arXiv:2102.03479*, 2021.

[Jang *et al.*, 2016] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2016.

[Kraemer and Banerjee, 2016] Landon Kraemer and Bikramjit Banerjee. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing*, 190:82–94, 2016.

[Lowe *et al.*, 2017] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *NIPS*, pages 6379–6390, 2017.

[Mao *et al.*, 2020] Hangyu Mao, Zhengchao Zhang, Zhen Xiao, Zhibo Gong, and Yan Ni. Learning agent communication under limited bandwidth by message pruning. In *AAAI*, pages 5142–5149, 2020.

[Niu *et al.*, 2021] Yaru Niu, Rohan Paleja, and Matthew Gombolay. Multi-agent graph-attention communication and teaming. In *AAMAS*, pages 964–973, 2021.

[Oliehoek and Amato, 2016] Frans A Oliehoek and Christopher Amato. *A Concise Introduction to Decentralized POMDPs*. Springer, 2016.

[Papoudakis *et al.*, 2021] Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V Albrecht. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. In *NeurIPS*, 2021.

[Rashid *et al.*, 2018] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *ICML*, pages 4295–4304, 2018.

[Samvelyan *et al.*, 2019] Mikayel Samvelyan, Tabish Rashid, Christian Schröder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philip H. S. Torr, Jakob N. Foerster, and Shimon Whiteson. The Starcraft multi-agent challenge. In *AAMAS*, pages 2186–2188, 2019.

[Singh *et al.*, 2019] Amanpreet Singh, Tushar Jain, and Sainbayar Sukhbaatar. Learning when to communicate at scale in multiagent cooperative and competitive tasks. In *ICLR*, 2019.

[Srivastava *et al.*, 2014] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[Wang *et al.*, 2020a] Jianhong Wang, Yuan Zhang, Tae-Kyun Kim, and Yunjie Gu. Shapley Q-value: A local reward approach to solve global reward games. In *AAAI*, pages 7285–7292, 2020.

[Wang *et al.*, 2020b] Tonghan Wang, Jianhao Wang, Chongyi Zheng, and Chongjie Zhang. Learning nearly decomposable value functions via communication minimization. In *ICLR*, 2020.

[Wang *et al.*, 2021] Yihan Wang, Beining Han, Tonghan Wang, Heng Dong, and Chongjie Zhang. DOP: Off-policy multi-agent decomposed policy gradients. In *ICLR*, 2021.