PMAC: Personalized Multi-Agent Communication

Xiangrui Meng[1,2], **Ying Tan**[1,2,3,4 *]

[1] School of Intelligence Science and Technology, Peking University
[2] Key Laboratory of Machine Perceptron (MOE), Peking University
[3] Institute for Artificial Intelligence, Peking University
[4] National Key Laboratory of General Artificial Intelligence
mxxxr@stu.pku.edu.cn, ytan@pku.edu.cn

Abstract

Communication plays a crucial role in information sharing within the field of multi-agent reinforcement learning (MARL). However, how to transmit information that meets individual needs remains a long-standing challenge. Some existing work focus on using a common channel for information transfer, which limits the capability for local communication. Meanwhile, other work attempt to establish peer-to-peer communication topologies but suffer from quadratic complexity. In this paper, we propose Personalized Multi-Agent Communication (PMAC), which enables the formation of peer-to-peer communication topologies, personalized message sending, and personalized message receiving. All these modules in PMAC are performed using only multilayer perceptrons (MLPs) with linear computational complexity. Empirically, we show the strength of personalized communication in a variety of cooperative scenarios. Our approach exhibits competitive performance compared to existing methods while maintaining notable computational efficiency.

Introduction

Multi-agent reinforcement learning (MARL) (Busoniu, Babuska, and De Schutter 2008) has emerged as a powerful paradigm for modeling and solving complex problems in various domains, such as robotics, autonomous systems (Zhou et al. 2020), game AI (Berner et al. 2019), and social networks (Leibo et al. 2017). But coordination among agents is always challenging, especially when it lacks access to the environment state or other useful information. As communication serves as a common mechanism for sharing information, recently, multi-agent reinforcement learning with communication (Comm-MARL) has been receiving increasing attention. Previous work (Foerster et al. 2016; Sukhbaatar, Szlam, and Fergus 2016; Peng et al. 2017) focus on differential communication channels for agents to exchange information during both training and execution. However, these work equally value all agents and overlook the potential benefits of personalized communication.

Just as there are no two identical leaves in the world, human cooperation often involves personalized communication in which each one plays a distinct role. Typically, we send different messages to different receivers based on their specific needs, and we process messages differently depending on where they come from. Moreover, not everyone needs to participate in every communication, as some information may be irrelevant to certain individuals. Therefore, the consideration of personalized communication is of great practical significance.

In terms of communication messages, to the best of our knowledge, there has been no specific discussion on how to send different messages to meet the needs of the receivers. On the contrary, various approaches have been proposed to get weights of received messages, assigning the contributions of different senders. For example, TarMAC (Das et al. 2019) employs message signatures and utilizes soft attention (Vaswani et al. 2017) to assign weights to incoming messages. Similarly, DGN (Jiang et al. 2020) and MAGIC (Niu, Paleja, and Gombolay 2021) employ multi-head attention (Vaswani et al. 2017) and graph attention (Veličković et al. 2017), respectively, for communication message aggregation. However, these methods primarily focus on personalized message aggregation at the receiving end. Furthermore, the use of attention introduces computational complexity of $O(n^2)$, which is unbearable when dealing with a large number of agents.

In terms of communication topology, (Jiang and Lu 2018; Jiang et al. 2020; Ding, Huang, and Lu 2020) predefine communication range based on the observation field. However, nearby agents often have similar observations, which limits the information gain from communication. Moreover, this approach may not be suitable for certain environments with limited observation fields or where agents are not included in the observation definition. Considering building adaptive communication topologies through learning methods, IC3NET (Singh, Jain, and Sukhbaatar 2018) and Gated-ACML (Mao et al. 2020) employ gating networks to determine whether an agent needs to send messages to others at a given time step. DC2Net (Meng and Tan 2023) employs an independent group channel for information integration. SchedNet (Kim et al. 2019) trains a scheduler using DDPG (Lillicrap et al. 2015) to select the top-K most relevant agents for communication. These approaches use a public channel to receive messages, but they do not establish peer-to-peer communication topologies between agents, missing out on the potential

[*] Corresponding author.

for personalized communication.

To build personalized communication, recent work utilize graphs to model peer-to-peer communication topologies, as graphs provide a flexible representation of global relationships among agents. G2ANet (Liu et al. 2020) employs hard attention to model undirected communication topologies between agents. FlowComm (Du et al. 2021) and MAGIC (Niu, Paleja, and Gombolay 2021) use directed graphs for more detailed communication topology modeling. However, both G2ANet and MAGIC build graphs based on the hidden states of agent pairs, resulting in $O(n^2)$ computational complexity, and FlowComm employs coupling flow to model the adjacency of a directed graph, which is also computationally complex. Although using directed graphs to model communication relationships is an ideal solution, the high computational complexity is still an unsolved problem.

Overall, in terms of communication messages, personalization is restricted to the receiver side, while in terms of communication topologies, modeling personalized communication topologies by directed graphs is a potential way, but suffers from high computational complexity. To tackle these difficulties, in this paper, we propose *Personalized Multi-Agent Communication* (PMAC), which consists of three modules: personalized communication topology, personalized message sending, and personalized message receiving. Unlike existing approaches, PMAC uses simple multilayer perceptron (MLPs) and learns personalized communication in linear time, reducing computational costs. Experimental results demonstrate that our approach can achieve better information sharing through personalized communication while maintaining computational efficiency.

Problem Formulation

In this section, we first introduce preliminaries and notations of our work. Secondly, we present propositions for personalized multi-agent communication. Finally, we define the directed communication graph and communication matrix, which will be utilized in constructing the peer-to-peer communication topology.

Dec-POMDP. We consider a fully cooperative MARL task that can be modeled as a Decentralized Partially Observable Markov Decision Process (Dec-POMDP) (Oliehoek and Amato 2016). The Dec-POMDP consists of a tuple $\langle \mathcal{S}, \mathcal{A}, P, R, O, \Omega, \mathcal{N}, \gamma \rangle$. $\mathcal{N} = \{1, 2, ..., n\}$ is set of agents. $s \in \mathcal{S}$ is the true state of the environment. Each agent i has a partial observation $o_i \in \Omega$ according to the observation function $O(\mathbf{s}, i)$: $\mathcal{S} \times \mathcal{N} \rightarrow \Omega$ and chooses an action $a_i \in \mathcal{A}$ based on the o_i. A joint action $\mathbf{a}$ is formed by $\mathbf{a} = [a_i] \in \mathcal{A}^n$. After taking the joint action $\mathbf{a}$, the transition probability function $P : \mathcal{S} \times \mathcal{A}^n \times \mathcal{S} \rightarrow [0, 1]$ causes the transition on the environment and leads to the global reward $r \in R(s, \mathbf{a})$: $S \times \mathcal{A}^n \rightarrow \mathbb{R}$. The formal objective is to maximize the expected cumulative reward $\mathbb{E}_{s,\mathbf{a}}[\sum_{t=0}^{T} \gamma^t R(s, \mathbf{a}) | s_0 = s, \mathbf{a}_0 = \mathbf{a}]$, in which $\gamma \in [0, 1]$ is the discount factor. In this work, we adopt a *centralized training with decentralized execution* (CTDE) paradigm (Kraemer and Banerjee 2016), and further relax it by incorporating communication among agents. In the execution process, each agent can condition its actions on its local observation history and communication messages it received.

Proposition 1 (Personalized Communication). *The personalized communication in our work comprises three modules: personalized communication topology, personalized message sending, and personalized message receiving.*

- *Personalized communication topology: Each agent can choose one or more communication recipients, i.e. peer-to-peer communication.*
- *Personalized message sending: Agents send different messages depending on the receiver. For instance, different receivers may have distinct needs for information.*
- *Personalized message receiving: Agents process received messages differently depending on their source. For instance, messages from different senders may contain distinct information.*

Proposition 2 (Communication Session). *A communication session serves as the fundamental in PMAC, encompassing the communication topology, message sending, and message receiving (Fig. 1).*

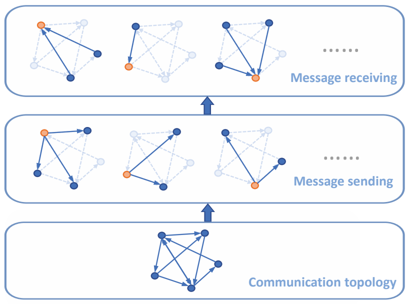


Figure 1: Basic steps in a communication session.

Definition 1 (Directed Communication Graph). *The directed communication graph is defined as a directed graph $G = (N, E)$, where N is the nodes set and E is the edges set. Each node corresponds to an agent, while the directed edge represents whether there is communication between one agent and another.*

Definition 2 (Communication Matrix). *The communication matrix is the adjacency matrix of the directed communication graph. For a directed communication graph with a node set $N = \{n_1, ..., n_n\}$, the communication matrix is a square $n \times n$ matrix* ***M*** *such that its element m_{comm}^{ij} is* ***one*** *when there is a communication from node n_i to node n_j, and* ***zero*** *when there is no communication.*

Methodology

In this section, we present *Personalized Multi-Agent Communication* (PMAC), which employs three modules to provide the agents with personalized communication. Its architecture consists of agents that condition their behavior on several communication sessions. Fig. 2 illustrates the complete setup for PMAC.

For each agent i, we first employ an encoding network that represents agents' hidden state at timestep t as $h_t^{i(1)}$ (Eq. 1), where (l) indicates the hidden state in the lth layer. There is an MLP to receive the current individual observation o_t^i as input at each time step, and an LSTM to encode the temporal state in the encoding network, where c_t^i and $h_t^{i(l)}$ are cell state and hidden state of the LSTM, respectively. We drop time t and layer (l) in the following notations for simplicity.

$$h_t^{i(1)} = \mathrm{LSTM}(\mathrm{MLP}(o_t^i), (c_{t-1}^i, h_{t-1}^{i(1)})) \tag{1}$$

We adopt the directed communication graph to model the personalized communication topologies. Unlike predefined communication topologies, we build the corresponding communication matrices adaptively through differential graph structure learning. Specifically, we assume that the communication matrix $\mathbf{M}_{comm}$ should follow the distribution $p(\mathbf{M}_{comm} \mid s)$. Since the global state s is often unavailable, some work (Lowe et al. 2017; Mao et al. 2020) utilize a concatenation of individual observations (or hidden states) to approximate it. Then the communication matrix follows the distribution $p(\mathbf{M}_{comm} \mid s) \approx p(\mathbf{M}_{comm} \mid o_1, \cdots, o_n) = p(\mathbf{M}_{comm} \mid h_1, \cdots, h_n)$. However, this approach can lead to a significant increase in the input dimensionality as the number of agents grows. To tackle this problem, some work (Liu et al. 2020; Niu, Paleja, and Gombolay 2021) consider building communication states for each pair of agents. They assume that the communication state between two agents is independent of others and utilize the following approximation: $p(m_{comm}^{ij} \mid s) \approx p(m_{comm}^{ij} \mid h_i, h_j) = p(m_{comm}^{ij} \mid h_1, ..., h_n), \mathbf{M}_{comm} = \left[m_{comm}^{ij}\right]_{n\times n}$. Nevertheless, this approach introduces another challenge in terms of $O(n^2)$ computational complexity, while building the directed communication graph. Consequently, it becomes a new challenge to build the directed communication graph more efficiently. Considering that the communication matrix is binary with a substantial number of zeros representing non-communication agent pairs, it is sparse and we introduce a low-rank approximation (Ye 2004) of the communication matrix to circumvent the quadric complexity, which introduces $2Kn$ times forward propagation with $O(n)$ computational complexity:

$$p(m_{comm}^{ij} \mid h_i, h_j) := \sum_{k=1}^{K} f_{\phi_1^k}(s_i \mid h_i) f_{\phi_2^k}(r_j \mid h_j) \tag{2}$$

where s_i and r_i are communication signals of the sender and receiver, respectively, while $f_{\phi_1^k}$ and $f_{\phi_2^k}$ are MLPs. We use reparameterization sampling from the distribution by gumble-softmax (Jang, Gu, and Poole 2016) to avoid the problem that the gradient cannot be backpropagated.

$$\hat{m}_{comm}^{ij} = \text{gumbel-softmax}\left(p(m_{comm}^{ij} \mid h_i, h_j)\right) \tag{3}$$

In message sending, existing work focuses on how the information is aggregated at the receiver side but ignores generating messages that meet the needs of different receivers, which is not only impractical but also could lead to invalid information transmission. Here, we use an MLP f_s to model the message sent to a particular recipient. Specifically, we add a one-hot ID of the receiver to the input of f_s to distinguish between different receivers, and the network learns the representation of messages for different IDs:

$$m_s^{ij} = f_s(\mathrm{concatenate}(h_t^i, \mathrm{ID}(j))), \quad \mathbf{M}_s = \left[m_s^{ij}\right]_{n\times n} \tag{4}$$

where m_s^{ij} is the message sent by agent i to agent j in terms of personalized message sending, $\mathbf{M}_s$ is the sending message matrix consists of m_s^{ij}. Although this approach allows for the generation of personalized communication messages for different receivers, unfortunately, it requires n^2 forward propagations of the network f_s, which again brings a significant computational burden. However, it can be observed that the network inputs are similar, as they consist of the same h_t^i. Therefore, we can simplify Eq. 4 according to Theorem 1 as follows:

Theorem 1. *Suppose $\boldsymbol{x} = \boldsymbol{x}_1 = \boldsymbol{x}_2 = ... = \boldsymbol{x}_n \in \mathbb{R}^d$, For a matrix $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^n \in \mathbb{R}^{n\times d}$, an identity matrix $\boldsymbol{I}_n = diag(1, 1, ..., 1) \in \mathbb{R}^{n\times n}$, weight matrices $\boldsymbol{W} \in \mathbb{R}^{(d+n)\times d}$, $\hat{\mathbf{W}} \in \mathbb{R}^{d\times d}$, and bias matrices $\boldsymbol{B} = \{\boldsymbol{b}_i\}_{i=1}^n \in \mathbb{R}^{n\times d}, \boldsymbol{b}_1 = \boldsymbol{b}_2 = ... = \boldsymbol{b}_n$, let $\boldsymbol{M} = [\boldsymbol{X}, \boldsymbol{I}]$. Then*

$$\boldsymbol{M} \cdot \boldsymbol{W} + \boldsymbol{B} = \{\boldsymbol{x} \cdot \hat{\boldsymbol{W}} + \boldsymbol{g}_i\}_{i=1}^n \tag{5}$$

where $\boldsymbol{g}_i, i = 1, ..., n$ are n unequal vectors belongs to $\mathbb{R}^d$.

Now, we can utilize a smaller network with reduced dimensions (parameterized by $\hat{\mathbf{W}}$ in Theorem 1) requiring only a single forward propagation and n vector summations to achieve personalized information sending. Then we can prune the messages that are not intended for communication based on the personalized communication topology $\mathbf{M}_{comm}$, resulting in the pruned sending message matrix $\mathbf{M}_s'$:

$$\mathbf{M}_s' = \mathbf{M}_{comm} \odot \mathbf{M}_s \tag{6}$$

where $\odot$ is the *Hadamard product*.

When an agent receives the messages sent by all other agents, the final step of the communication session is message receiving. Similar to the message sending, we use simple MLPs to get the personalized received messages:

$$m_r^{ij} = f_r(\mathrm{concatenate}({m'}_s^{ij}, \mathrm{ID}_j)), \quad \mathbf{M}_r = \left[m_r^{ij}\right]_{n\times n} \tag{7}$$

where m_r^{ij} is the message received by agent i from agent j, and it can be summed up to obtain the aggregated message of agent i: $m_r^i = \sum_{j=1}^n m_r^{ij}$. However, different from the message sending, since the networks no longer have similar inputs, it's hard to reduce the computational complexity while maintaining theoretical consistency. So here, we simplify the neural network f_r to an affine transformation g_r. According to the properties of affine transformations, we can swap the information reception and aggregation, thus reducing the forward propagation times of g_r from n to 1.

$$\sum_j g_r(\mathrm{concatenate}({m'}_s^{ij}, \mathrm{ID}_j)) = g_r(\sum_j \mathrm{concatenate}({m'}_s^{ij}, \mathrm{ID}_j)) \tag{8}$$

Finally, $m_r^i, i = 1, ..., n$ is considered as the output of the current communication session.

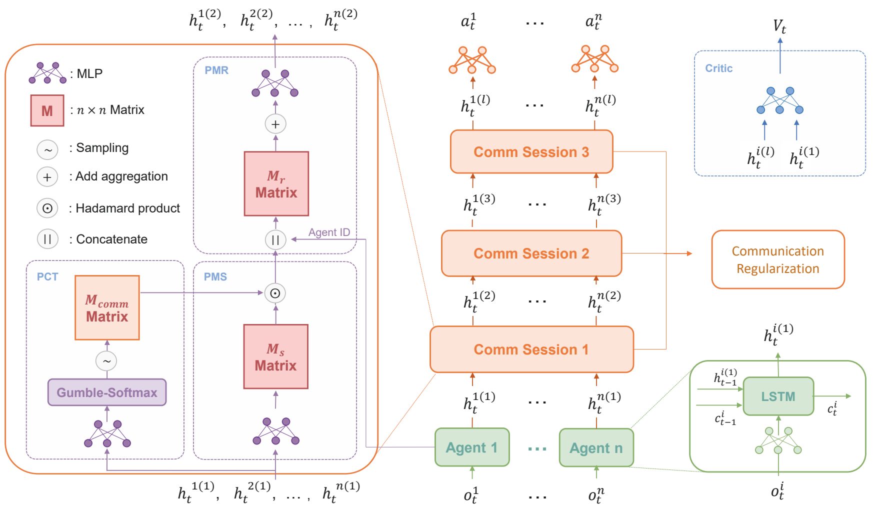


Figure 2: Architecture for PMAC.

Communication Regularization

Multiple rounds of communication can enhance information sharing, especially in complex tasks (Das et al. 2019; Jiang et al. 2020). Based on this idea, we stack multiple communication sessions to facilitate effective information sharing. Drawing inspiration from CNN models (Krizhevsky, Sutskever, and Hinton 2012; Szegedy et al. 2015) in image feature extraction, as the network layers go deeper, the size of the convolution kernel decreases to extract local features. We apply a similar concept to multiple communication sessions. As the communication session layer deepens, the number of communications decreases to capture more important relationships. We achieve this by limiting the number of communication agent pairs in the deeper communication session layers. Formally, we introduce a communication regularization to control the number of edges in the generated directed graph:

$$\mathcal{L}_{comm} = \sum_{l} log(l) \sum_{i} \sum_{j} I\{m_{comm}^{ij} \neq 0\} \tag{9}$$

where l indicates the lth communication session. We use actor-critic (Sutton and Barto 2018) to train our model, and finally, our optimization objective can be written as follows:

$$\begin{aligned}\nabla_{\theta_\pi}\mathcal{J}(\theta_\pi) =& \mathbb{E}_{\boldsymbol{o},\boldsymbol{a}}[\mathbb{E}_\pi\left[\nabla_{\theta_\pi}\log\pi(a_i|o_i)(R_i - V(o_i))\right] \\ &+ \beta\nabla_{\theta_V}(R_i - V(o_i))^2 + \eta\nabla_{\theta_{\pi_i}}\mathcal{L}_{comm}]\end{aligned} \tag{10}$$

Representational Complexity

The communication topology class representable with PMAC includes any peer-to-peer communication topology that can be formalized into a directed communication graph in our problem definition. (Fig. 3.) This expands upon the communication topologies that are predefined by other methods. Specifically, when the learned communication matrix is an identity matrix, there is no communication between agents and each agent performs independent learning. When the learned communication matrix is an all-ones matrix, there is a fully connected communication topology like DIAL (Foerster et al. 2016), CommNet (Sukhbaatar, Szlam, and Fergus 2016), and TarMAC (Das et al. 2019). It is also possible to represent more complex communication topologies, such as symmetric communication: IC3Net (Singh, Jain, and Sukhbaatar 2018), ATOC (Jiang and Lu 2018), G2ANet (Liu et al. 2020), DGN (Jiang et al. 2020), and asymmetric communication: SchedNet (Kim et al. 2019), I2C (Ding, Huang, and Lu 2020), MAGIC (Niu, Paleja, and Gombolay 2021).

$$\text{Receiver}$$

$$\text{Sender}\quad \underset{\text{Independent}}{\begin{bmatrix}1&0&0&0&0\\0&1&0&0&0\\0&0&1&0&0\\0&0&0&1&0\\0&0&0&0&1\end{bmatrix}} \underset{\text{Symmetric}}{\begin{bmatrix}1&0&1&1&0\\0&0&1&0&1\\1&1&1&1&0\\1&0&1&0&1\\0&1&0&1&1\end{bmatrix}} \underset{\text{Asymmetric}}{\begin{bmatrix}1&0&1&1&0\\0&0&1&0&1\\1&0&1&1&0\\0&1&1&0&1\\0&0&1&0&1\end{bmatrix}} \underset{\text{Full}}{\begin{bmatrix}1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\end{bmatrix}}$$

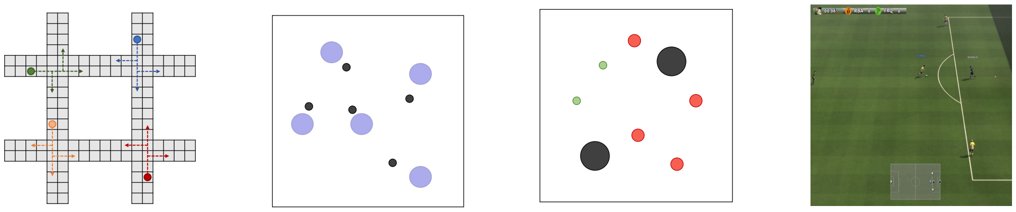
Figure 3: Examples of Representational Complexity.

In some algorithms, the communication process can be more complex and more than one round of communication may take place within a single time step. For this case, The stacked multiple communication sessions can characterize the features of this multi-round communication.

Experiments

We empirically evaluate PMAC on a variety of multi-agent cooperative tasks which are hard-level Traffic Junction (hard-TJ) (Sukhbaatar, Szlam, and Fergus 2016), Cooperative Navigation (CN), and Predator-Prey (PP) in Multi-agent Particle Environment (MPE) (Mordatch and Abbeel 2017; Lowe et al. 2017), and Google Research Football (GRF) (Kurach et al. 2020). (Fig. 4.).

- **TJ** We use the hard level TJ task on a 18×18 grid (see Fig. 4(a)). The total number of cars at any given time is limited to $N = 20$.



(a) TJ (b) CN (c) PP (d) GRF

Figure 4: Several multi-agent cooperative tasks involved in our experiments.

- **CN** The Cooperative Navigation task (see Fig. 4(b)) has N agents and landmarks ($N = 3$ by default). Here, we make this task harder by changing the number of agents to $N = 5$ and limiting the length of each episode to 20 time steps.
- **PP** The Predator-Prey task (see Fig. 4(c)) has N_1 preys and N_2 predators. Here, we make this task harder by setting the number of predators and prey to 4 and 2, respectively. The length T of each episode is 40 time steps.
- **GRF** We conducted experiments in the *3 versus 1 with Keeper* scenario from Football Academy (see Fig. 4(d)). We use the sparse reward that $+1$ when a goal is scored.

Results

We conducted a study on the hard-TJ task, running the five methods for 1,000,000 episodes and presenting the results in Table 1. In this scenario, where each agent has a limited observation range of size 1, communication plays a crucial role in avoiding collisions. As a communication method, PMAC not only shows its effectiveness in information sharing within this partially observable environment but also exhibits faster computational efficiency. Figure 5 shows the average time (in seconds) required to complete 1000 episodes. Under the condition of 20 agents in this task, other methods consume significant time due to the quadratic complexity of their modules. In contrast, PMAC reduces the computational complexity of each module using an equivalent or approximate method, achieving a success rate of 85.1% which outperforms the other four methods.

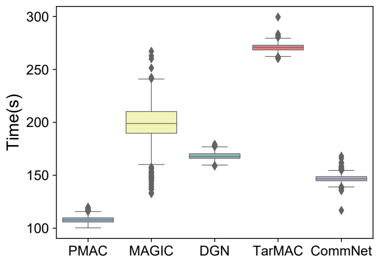
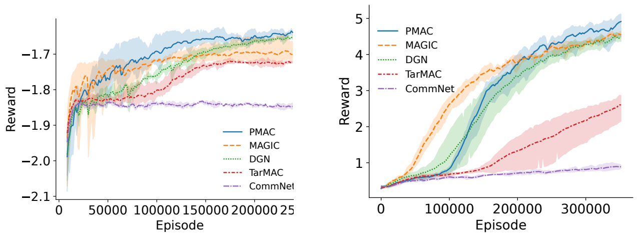


Figure 5: Comparison of computational efficiency of the five methods.

In the modified CN task, all agents are given a limited time of 20 time steps to occupy the target. This task imposes a higher demand on collaborative decision-making among the agents. They must learn to avoid making meaningless actions (such as wandering between two targets) and communicate with each other to prevent collisions or occupying the same target. The results in Fig. 6(a) indicate that PMAC achieves higher rewards by occupying more targets. PMAC's personalized communication topology allows agents to selectively communicate with each other, enhancing communication efficiency within the limited time step. In contrast, although MAGIC performs better in the early stage, it struggles to capture precise communication relationships in the later stage of learning due to the complex information aggregation by GAT.

In the modified PP task, as the prey is faster than the predator, the predators must learn to capture prey in a group of two to accomplish this task. This task places a high demand on coordination among the agents. The experimental results are shown in Fig. 6(b) in terms of mean reward, averaged over all agents and time steps. The three graph-based methods DGN, MAGIC, and PMAC demonstrate relatively better performance in this task. This can be attributed to the need for a grouping of agents during the pursuit, making peer-to-peer communication crucial for identifying partners. Although PMAC initially shows lower performance compared to both DGN and MAGIC, it gradually learns better collaboration strategies and eventually outperforms the other methods.



(a) CN (b) PP

Figure 6: Experimental results on CN and PP. Shaded regions are standard deviations over 3 runs.

The experimental results show the overall superiority of the PMAC (Fig. 7). Although this is a scenario with only three agents, collaboration among the agents is also critical. The ball carrier should strive to approach the goal and score, while the other two teammates need to create space to receive wide passes for better goal opportunities. Each agent has a personalized task, making personalized communication

	CommNet	TarMAC	DGN	MAGIC	PMAC
Success Rates (%)	60.6 ± 10.5	78.6 ± 2.9	79.1 ± 9.8	79.8 ± 7.7	$\mathbf{85.1} \pm \mathbf{6.6}$

Table 1: Success rates on hard TJ.

even more important in this scenario. It can be observed that the CommNet, which equally values all agents, achieves an average win rate of only 40%. The other two graph-based communication methods, MAGIC and DGN, due to the complexity of their graph-building process, struggle to adapt to the dynamic of the environment, resulting in high variance. TarMAC is comparable, but it still has a slower convergence rate compared to PMAC. This also demonstrates PMAC's strong ability to model personalized communication topologies to make proper team-decision. From the final success rates, PMAC achieves the best performance among the five methods.

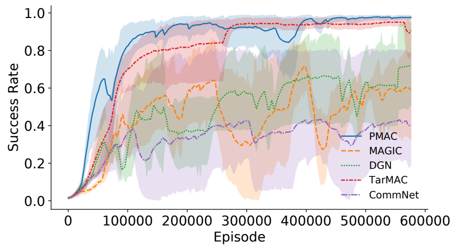


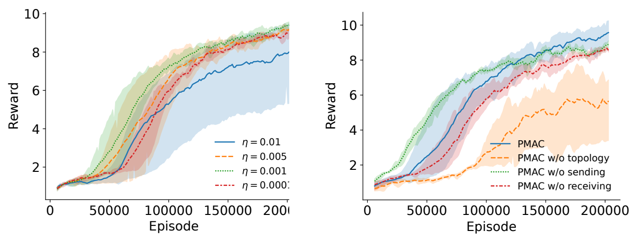
Figure 7: Experimental results on GRF. Shaded regions are standard deviations over 5 runs.

It is worth noting that in the GRF environment, agents have access to the global state. Generally, communication becomes more crucial when agents have limited visibility, as communication enables them to gain valuable information. However, our results demonstrate that even agents with a global view can significantly improve their performance by sharing information through PMAC. We think that although PMAC does not utilize attention, it implicitly assigns weights to different information through multiple communication sessions and personalized communication. As a result, each agent captures the most important information in the global state. In addition, as the actions of the agents are generated by the policy network, the parameters of the policy network inherently contain information about the agents' future decisions. Through communication, agents can also share their decision-making potential, which is another explanation of the favorable performance of PMAC in the environment with a global view.

Ablation Study

In this section, we present ablation study and calculations on predator-prey (PP) that complement our experimental results.

Ablation on Communication Regularization We compare four different coefficients η (0.01, 0.005, 0.001, 0.001) of communication regularization and evaluate how it affects the performance of PMAC. Figure 8(a) shows the learning curves in terms of rewards of different η. Notably, PMAC demonstrates faster convergence and achieves higher rewards when $\eta = 0.001$. It indicates that an excessive amount of communication can lead to the failure to capture local communication relationships, while insufficient communication can result in an ineffective transfer of information.



(a) Regularization coefficients (b) Modules in PMAC

Figure 8: Ablation study on PP. Shaded regions are standard deviations over 3 runs.

Ablation on Modules in PMAC Based on the results in Fig. 8(a), we select $\eta = 0.001$ as the coefficient for communication regularization and proceed to evaluate the impact of three modules: personalized communication topology, personalized message sending, and personalized message receiving.

- PMAC w/o topology: Fully connected communication topology.
- PMAC w/o sending: Sending same messages to different receivers.
- PMAC w/o receiving: Aggregating the received messages without considering the sender.

The results in Fig. 8(b) highlight the significant impact of personalized communication topology on overall performance. Specifically, the absence of personalized communication topology results in a notable drop in performance. Similarly, the absence of personalized message reception can also degrade performance. Conversely, PMAC without personalized message sending exhibits a faster convergence rate and lower variance, but its final reward falls short of PMAC. This can be attributed to that personalized message sending, despite posing challenges for training during the initial stages, has a greater potential to learn effective patterns of sending meaningful messages and ultimately receive higher rewards in the later stages.

Efficiency of Modules in PMAC We performed a computational efficiency analysis for the three modules, and the results are shown in Figure 9.

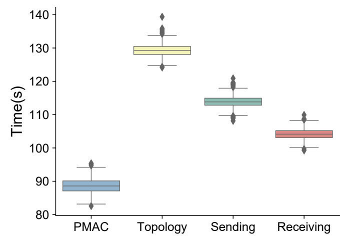


Figure 9: Efficiency of modules in PMAC on PP.

- Topology: Building personalized communication topology by quadratic complexity methods. (Without low-rank approximation.)
- Sending: Sending personalized messages by quadratic complexity methods. (Without function equivalence.)
- Receiving: Receiving personalized messages by quadratic complexity methods. (Without affine transformation approximation.)

Among these modules, the low-rank approximation of the communication topology contributes most to the computational efficiency of PMAC. Additionally, there is a decrease in computational efficiency when the equivalence and approximation of personalized message sending and receiving are not employed. By utilizing all these mechanisms to reduce computational complexity, PMAC achieves the highest level of computational efficiency.

Related Work

In Comm-MARL, **Communication topology** is crucial for information sharing. Early work consider fully connected communication topologies (Foerster et al. 2016; Sukhbaatar, Szlam, and Fergus 2016; Peng et al. 2017). However, these simple topologies not only perform poorly in complex environments, but also incur significant communication costs. To prune the communication messages, IC3NET (Singh, Jain, and Sukhbaatar 2018) and Gated-ACML (Mao et al. 2020) employ gating networks for each agent to determine whether it is engaged in communication. SchedNet (Kim et al. 2019) learns a scheduler module that enables agents to select a subset of top-K agents to communicate with. In ETCNET (Hu et al. 2021), the communication cost is penalized during training to optimize the gating module. DC2Net (Meng and Tan 2023) uses two independent channels for both individual and group level communication learning, effectively reducing communication costs. Another class of approaches focuses on communication with nearby (or observable) agents. In ATOC (Jiang and Lu 2018), the initiator selects collaborators within its observation field to establish communication. DGN (Jiang et al. 2020) introduces graph convolutional communication within the observable field. LSC (Sheng et al. 2022) clusters the communication group based on a predefined radius.

Considering **message sending**, the most common approach is to employ a simple neural network for message encoding (Foerster et al. 2016; Sukhbaatar, Szlam, and Fergus 2016; Peng et al. 2017). FCMNet (Wang and Sartoretti 2022) learns a multi-hop communication protocol based on recurrent neural networks. However, to the best of our knowledge, there has been no specific discussion on how to generate messages on the sender side for specific receivers. More work has focused on how to perform effective **message receiving** at the receiver side. Early approaches assume that the information from all agents has equal contributions (Sukhbaatar, Szlam, and Fergus 2016; Singh, Jain, and Sukhbaatar 2018). Other approaches concatenated messages for further processing (Foerster et al. 2016; Kim et al. 2019; Hu et al. 2021; Kim, Park, and Sung 2020). Currently, there is a growing focus on treating messages received from different senders in a personalized manner. Approaches such as attention mechanisms (Das et al. 2019; Liu et al. 2020; Ryu, Shin, and Park 2020; Niu, Paleja, and Gombolay 2021) are used to assign weights to the received messages, while RNNs (Peng et al. 2017; Jiang and Lu 2018; Wang and Sartoretti 2022) are employed to encode the sequential information of the received messages. These approaches enable agents to focus on the most relevant information during the message receiving process, but they also introduce higher computational complexity.

Recently, there has been a considerable amount of research utilizing graphs to model peer-to-peer communication topologies. MAGNet (Malysheva, Kudenko, and Shpilman 2019) trains a neural network to generate a complete graph and learn the relative importance of each edge. GCS (Ruan et al. 2022) proposes acyclicity and depth of the graph as signals for learning graph structures. DGN (Jiang et al. 2020) and CCOMA (Su, Adams, and Beling 2020) employ multi-layer graph convolution to adapt the dynamics in the multi-agent environment. G2ANet (Liu et al. 2020) utilizes hard attention to model undirected communication topologies among agents. FlowComm (Du et al. 2021) and MAGIC (Niu, Paleja, and Gombolay 2021) propose similar ideas to our work in using directed graphs to build multi-agent communication topologies. However, their approaches differ from ours in the following ways: (1) They do not consider sending personalized messages to different receivers. (2) They do not consider the local communication to enhance feature extraction, which is reflected in our work through the communication regularization. (3) FlowComm employs coupling flow to build directed communication graphs, while MAGIC builds the graph by considering the hidden states of each pair of agents, resulting in a computational complexity of $O(n^2)$. Both methods incur a significant computational costs.

Conclusion and Future Work

In this paper, we propose a low computational complexity approach PMAC to model personalized multi-agent communication. Three linear complexity modules are responsible for personalized communication topology, personalized message sending, and personalized message receiving, respectively. Experimental results in several cooperative tasks demonstrate the effectiveness of our method while maintaining high computational efficiency.

The personalized communication of PMAC is built on using IDs to identify different agents. While this represents a simple approach to personalized communication, it will be future work to introduce richer identification information to model more complex personalized communication.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant No. 62276008, 62250037, and 62076010), and partially supported by the National Key R&D of China (Grant #2022YFF0800601).

References

Berner, C.; Brockman, G.; Chan, B.; Cheung, V.; Dębiak, P.; Dennison, C.; Farhi, D.; Fischer, Q.; Hashme, S.; Hesse, C.; et al. 2019. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*.

Busoniu, L.; Babuska, R.; and De Schutter, B. 2008. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2): 156–172.

Das, A.; Gervet, T.; Romoff, J.; Batra, D.; Parikh, D.; Rabbat, M.; and Pineau, J. 2019. Tarmac: Targeted multi-agent communication. In *International Conference on Machine Learning*, 1538–1546. PMLR.

Ding, Z.; Huang, T.; and Lu, Z. 2020. Learning individually inferred communication for multi-agent cooperation. *Advances in Neural Information Processing Systems*, 33: 22069–22079.

Du, Y.; Liu, B.; Moens, V.; Liu, Z.; Ren, Z.; Wang, J.; Chen, X.; and Zhang, H. 2021. Learning correlated communication topology in multi-agent reinforcement learning. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, 456–464.

Foerster, J. N.; Assael, Y. M.; de Freitas, N.; and Whiteson, S. 2016. Learning to Communicate with Deep Multi-Agent Reinforcement Learning. *arXiv:1605.06676 [cs]*. ArXiv: 1605.06676.

Hu, G.; Zhu, Y.; Zhao, D.; Zhao, M.; and Hao, J. 2021. Event-triggered communication network with limited-bandwidth constraint for multi-agent reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*.

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

Jiang, J.; Dun, C.; Huang, T.; and Lu, Z. 2020. Graph Convolutional Reinforcement Learning. *arXiv:1810.09202 [cs, stat]*. ArXiv: 1810.09202.

Jiang, J.; and Lu, Z. 2018. Learning Attentional Communication for Multi-Agent Cooperation. *arXiv:1805.07733 [cs]*. ArXiv: 1805.07733.

Kim, D.; Moon, S.; Hostallero, D.; Kang, W. J.; Lee, T.; Son, K.; and Yi, Y. 2019. Learning to Schedule Communication in Multi-agent Reinforcement Learning. *arXiv:1902.01554 [cs]*. ArXiv: 1902.01554.

Kim, W.; Park, J.; and Sung, Y. 2020. Communication in multi-agent reinforcement learning: Intention sharing. In *International Conference on Learning Representations*.

Kraemer, L.; and Banerjee, B. 2016. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing*, 190: 82–94.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60: 84 – 90.

Kurach, K.; Raichuk, A.; Stańczyk, P.; Zając, M.; Bachem, O.; Espeholt, L.; Riquelme, C.; Vincent, D.; Michalski, M.; Bousquet, O.; et al. 2020. Google research football: A novel reinforcement learning environment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 4501–4510.

Leibo, J. Z.; Zambaldi, V.; Lanctot, M.; Marecki, J.; and Graepel, T. 2017. Multi-agent reinforcement learning in sequential social dilemmas. *arXiv preprint arXiv:1702.03037*.

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*.

Liu, Y.; Wang, W.; Hu, Y.; Hao, J.; Chen, X.; and Gao, Y. 2020. Multi-agent game abstraction via graph attention neural network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 7211–7218.

Lowe, R.; Wu, Y.; Tamar, A.; Harb, J.; Abbeel, P.; and Mordatch, I. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. *Neural Information Processing Systems (NIPS)*.

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

Mao, H.; Zhang, Z.; Xiao, Z.; Gong, Z.; and Ni, Y. 2020. Learning agent communication under limited bandwidth by message pruning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 5142–5149.

Meng, X.; and Tan, Y. 2023. Learning Group-Level Information Integration in Multi-Agent Communication. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, 2601–2603.

Mordatch, I.; and Abbeel, P. 2017. Emergence of Grounded Compositional Language in Multi-Agent Populations. *arXiv preprint arXiv:1703.04908*.

Niu, Y.; Paleja, R. R.; and Gombolay, M. C. 2021. Multi-Agent Graph-Attention Communication and Teaming. In *AAMAS*, 964–973.

Oliehoek, F. A.; and Amato, C. 2016. *A concise introduction to decentralized POMDPs*. Springer.

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

Ruan, J.; Du, Y.; Xiong, X.; Xing, D.; Li, X.; Meng, L.; Zhang, H.; Wang, J.; and Xu, B. 2022. GCS: Graph-Based Coordination Strategy for Multi-Agent Reinforcement Learning. *arXiv preprint arXiv:2201.06257*.

Ryu, H.; Shin, H.; and Park, J. 2020. Multi-agent actor-critic with hierarchical graph attention network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 7236–7243.

Sheng, J.; Wang, X.; Jin, B.; Yan, J.; Li, W.; Chang, T.-H.; Wang, J.; and Zha, H. 2022. Learning structured communication for multi-agent reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 36(2): 50.

Singh, A.; Jain, T.; and Sukhbaatar, S. 2018. Learning when to communicate at scale in multiagent cooperative and competitive tasks. *arXiv preprint arXiv:1812.09755*.

Su, J.; Adams, S.; and Beling, P. A. 2020. Counterfactual multi-agent reinforcement learning with graph convolution communication. *arXiv preprint arXiv:2004.00470*.

Sukhbaatar, S.; Szlam, A.; and Fergus, R. 2016. Learning Multiagent Communication with Backpropagation. *arXiv:1605.07736 [cs]*. ArXiv: 1605.07736.

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

Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; and Rabinovich, A. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1–9.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Wang, Y.; and Sartoretti, G. 2022. FCMNet: Full Communication Memory Net for Team-Level Cooperation in Multi-Agent Systems. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '22, 1355–1363. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450392136.

Ye, J. 2004. Generalized low rank approximations of matrices. In *Proceedings of the twenty-first international conference on Machine learning*, 112.

Zhou, M.; Luo, J.; Villella, J.; Yang, Y.; Rusu, D.; Miao, J.; Zhang, W.; Alban, M.; Fadakar, I.; Chen, Z.; et al. 2020. Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving. *arXiv preprint arXiv:2010.09776*.