# Context-aware Communication for Multi-agent Reinforcement Learning

Xinran Li
The Hong Kong University of Science and Technology
Hong Kong, China
xinran.li@connect.ust.hk

Jun Zhang
The Hong Kong University of Science and Technology
Hong Kong, China
eejzhang@ust.hk

## ABSTRACT

Effective communication protocols in multi-agent reinforcement learning (MARL) are critical to fostering cooperation and enhancing team performance. To leverage communication, many previous works have proposed to compress local information into a single message and broadcast it to all reachable agents. This simplistic messaging mechanism, however, may fail to provide adequate, critical, and relevant information to individual agents, especially in severely bandwidth-limited scenarios. This motivates us to develop context-aware communication schemes for MARL, aiming to deliver personalized messages to different agents. Our communication protocol, named CACOM, consists of two stages. In the first stage, agents exchange coarse representations in a broadcast fashion, providing context for the second stage. Following this, agents utilize attention mechanisms in the second stage to selectively generate messages personalized for the receivers. Furthermore, we employ the learned step size quantization (LSQ) technique for message quantization to reduce the communication overhead. To evaluate the effectiveness of CACOM, we integrate it with both actor-critic and value-based MARL algorithms. Empirical results on cooperative benchmark tasks demonstrate that CACOM provides evident performance gains over baselines under communication-constrained scenarios. The code is publicly available at https://github.com/LXXXXR/CACOM.

## KEYWORDS

Reinforcement Learning; Multi-agent Systems; Communication

## 1 INTRODUCTION

Cooperative multi-agent reinforcement learning (MARL) has recently emerged as an exciting research avenue due to its applicability to real-world scenarios. Many of these applications naturally call for multi-agent frameworks, such as resource management [32], package delivery [26], disaster rescue [22], and robots control [29]. Despite the considerable success of seminal works [18, 24, 33] in generalizing reinforcement learning algorithms to multi-agent cases

under the centralized training and decentralized execution (CTDE) paradigm [13, 19], further improvement in MARL is hindered by a few prominent obstacles, such as non-stationarity and partial observability [21, 35].

Subsequent research efforts [23, 28] aim to tackle the non-stationary and partially-observable issues in MARL by incorporating communication into the framework. By integrating local information (observations or histories) with messages from other agents, an agent can gain a more comprehensive understanding of the environment, leading to improved decision-making. Prior methods, focusing on improving the MARL task performance, mostly adopt continuous messages and broadcasting mechanisms, thus inducing substantial communication overhead.Communication channels typically face severe bandwidth constraints [10], making it difficult to directly deploy previous approaches to real-world multi-agent systems (MASs). Therefore, it is crucial to develop MARL communication protocols that can utilize the limited bandwidth more efficiently.

In this work, we consider an MAS with a limited communication budget. Ideally, a communication protocol that operates well under a tight budget shall only allow transmitting information that is useful for the target receiver, i.e., the message generation shall be context-aware. This makes the broadcast communication protocol a suboptimal choice, as different agents likely need different information. But the question remains, i.e., how to identify the pieces of information worth transmitting between two particular agents? Note that MARL aims to solve a decision-making problem at its core, therefore it is up to the receivers to determine what information is relevant and helpful for decision making. This motivates us to develop a receiver-centric communication protocol, in which the receiver initiates the communication by providing the sender with the "context" before the sender can send context-aware messages. We compare ideas of the traditional broadcasting scheme and context-aware communication (CACOM) using an illustrative example in Figure 1. In the first stage of our proposed CACOM, each agent broadcasts a very short context message conveying coarse local information. In the second stage, agents selectively send longer, personalized messages to specific peers based on the context messages they received in the first stage. It is noteworthy that similar communication strategies among humans have been successfully utilized in the fields of marketing [1] and healthcare [14], where they are termed as customized communication.

To aid personalized message generation, we treat the entities in agents' observation as tokens and employ an attention-based architecture to encode features and generate personalized messages. This approach enables the network to learn which parts of the features are most important for the intended receiver. Furthermore, we quantize the messages through the LSQ technique [5] to make
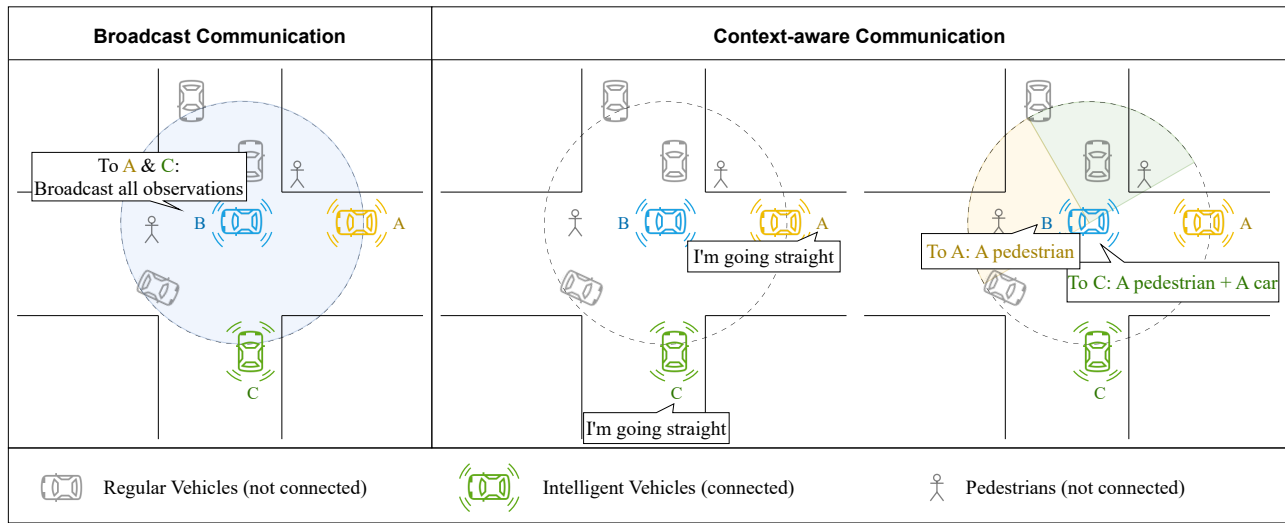
**Figure 1: An illustrative example in a busy traffic junction. Under the broadcasting communication scheme, without knowledge of agent A and C's intentions, agent B needs to broadcast all its observations, which will result in heavy communication overhead. In contrast, when adopting context-aware communication, agent A and C first convey their local information in short context messages. Then agent B generates personalized messages for A and C based on the context messages from the previous stage. In this way, more context-relevant messages are provided for decision making with much lower communication overhead.**

the communication blocks compatible to digital communication systems. Since CACOM is agnostic to specific MARL algorithms, we combine it with MADDPG [18] and QMIX [24] to evaluate its efficacy on cooperative multi-agent benchmarks in communication-constrained scenarios. Specifically, we conduct experiments on two scenarios in multi-agent particle environment (MPE) [18] and four maps in the StarCraft multi-agent challenge (SMAC) [25]. The results showcase the superior performance of CACOM in comparison with baselines. Our contributions are summarized as follows:

- We investigate a realistic multi-agent system with limited communication resources and propose a context-aware communication protocol in MARL, namely CACOM, to efficiently utilize the limited communication budgets.
- We leverage various attention blocks to generate personalized messages based on the senders' and receivers' local information, and we incorporate learned step size quantization (LSQ) to ensure digital communication while keeping the overall network differentiable.
- Through adequate experimentation on cooperative multi-agent benchmarks with limited communication budgets, CACOM has been demonstrated to be effective and outperform traditional broadcast communication protocols.

## 2 RELATED WORK

In recent years, there has been a surge of interest in the field of MARL, particularly in the context of CTDE [13]. Numerous algorithms have been developed to support cooperative MARL, including QMIX [24], MAPPO [33], and MADDPG [18]. To mitigate the impact of partially observablility and promote cooperation, communication has been incorporated into MASs [23, 28]. Notably, by

applying attention mechanisms and graph neural networks (GNNs), TarMAC [2], DICG [15] and DGN [11] learn local embeddings and broadcast the messages to all the reachable agents. Subsequent methods propose to further improve the performance from two aspects: the sender side and the receiver side. From the sender side, MAIC [34] and ToM2C [31] utilize teammate modeling to generate incentive messages based on identities of the receivers. The other line of research aims to develop more delicate aggregation schemes at the receiver side to utilize the received messages more efficiently, leading to algorithms such as G2A [17] and MASIS [7]. However, most of these works focus more on enhancing the MARL task performance, with less emphasis on controlling the communication costs, which could result in prohibitive communication overhead in real-world systems.

There have been methods, e.g., ATOC [12], IC3 [27] and I2C [4], trying to reduce the communication overhead by applying local gating mechanisms to dynamically prune the communication links among agents. However, since the messages are continuous, it remains unknown whether these methods can work well in systems with extremely limited communication budgets. In contrast, DIAL [6] and vector quantization-based methods [16] generate discrete messages directly, while NDQ [30] and TMC [37] also designs messages in a compact way. Nevertheless, the expressiveness of discrete messages will be limited by the communication budget with broadcasting communication schemes. Furthermore, ETC [9], VBC [36] and MBC [8] have proposed event-triggered communication to reduce the communication frequency and address the communication constraints. This line of research focuses on optimizing when transmissions shall occur, thus is orthogonal to our

work and can potentially be combined with our work to achieve better performance with lower communication budgets.

In this work, we explicitly account for the effect of limited communication budgets in MASs. We utilize the LSQ technique to ensure that messages are discrete and therefore applicable to communication-constrained scenarios. Moreover, we leverage the context-aware communication protocol to generate optional links and personalized messages, enabling more efficient use of communication resources. Compared with previously proposed incentive messages [31, 34] based on identities instead of specific context, CACOM can better address environment dynamics and generate more compact messages for receivers. Comprehensive comparisons with existing methods will illustrate the effects of limited communication resources, as well as the importance of sending personalized messages to agents.

## 3  BACKGROUND

**Decentralized Partially Observable Markov Decision Process (Dec-POMDP):**  In this work, we consider a fully cooperative partially observable multi-agent task, which can be modeled as a decentralized partially observable Markov decision process (Dec-POMDP) [20]. The Dec-POMDP is defined by a tuple $\mathcal{M} = \langle S, A, P, R, \Omega, O, n, \gamma \rangle$, where $n$ denotes the number of agents and $\gamma \in (0, 1]$ is the discount factor that balances the trade-off between immediate and long-term rewards.

At each timestep $t$, when the environment state is $s \in S$, each agent $i$ receives a local observation $o_i \in \Omega$ drawn from the observation function $O(s, i)$. Before taking any actions, agents communicate with each others and exchange local information. Subsequently, each agent $i$ follows its local policy $\pi_i$ to select an action $a_i \in A$ based on its local information and information obtained from communication. Individual action decisions will then form a joint action $\boldsymbol{a} \in A^n$, which results in a state transition to the next state $s' \sim P(s'|s, \boldsymbol{a})$ and a global reward $r = R(s, \boldsymbol{a})$. Each agent keeps a local action-observation history denoted as $h_i \in (\Omega \times A)$. The team objective is to learn the policies such that the expectation of discounted accumulated reward $G_t = \sum_t \gamma^t r^t$ is maximized.

**Centralized Training and Decentralized Execution (CTDE):** The CTDE [13, 19] paradigm in MARL, which combines the advantages of centralized training for improved coordination and decentralized execution for enhanced scalability, has emerged as a popular approach for solving complex problems in MASs. Recently, much research effort has been made to generalize single-agent RL algorithms to multi-agent cases under the CTDE paradigm, resulting in a variety of concrete MARL algorithms. In this work, we build our proposed communication blocks on top of both actor-critic algorithm MADDPG [18] and value-based algorithm QMIX [24] for different environments, aiming to showcase the feasibility of further improving the learning performance of these algorithms when communication is allowed. Training objectives and details of these MARL algorithms are provided in Supplementary A.

**Learned Step Size Quantization (LSQ):** When developing communication blocks for MARL algorithms, the desideratum is to have discrete messages, which can be transmitted in real-world digital communication systems. At the same time, we also want to ensure the end-to-end differentiable property of the overall network
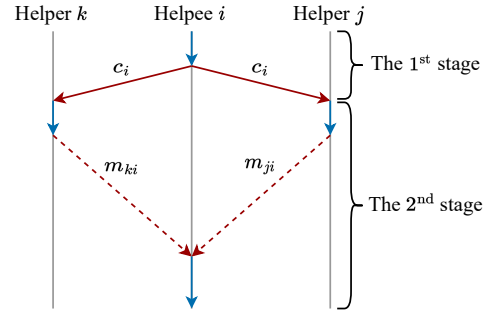


**Figure 2: Illustration of the CACOM protocol from the helpee agent $i$'s perspective. The blue arrows denote local processing and the red arrows denote communication. At each timestep, agent $i$ first broadcasts a context message $c_i$ to all its peers agent $k$ and $j$. Then, after local processing, agents $k$ and $j$ decide whether to reply and what to send, respectively.**

architecture. To meet such requirements without impeding the performance, we migrate quantization techniques in low-precision networks into MARL network architecture design. In particular, we choose LSQ [5] due to its simplicity of implementation and its ability to maintain high accuracy in the low-bit precision networks across a wide range of network architectures on the benchmark dataset ImageNet [3]. In the LSQ algorithm, the quantizer step size is treated as a learnable parameter rather than a fixed value, which allows more flexibility.

## 4  CONTEXT-AWARE COMMUNICATION FOR MARL

In this section, we first introduce CACOM, the two-stage communication protocol we designed to enable context-aware communication in MARL. We then detail the network architecture to accommodate CACOM. Lastly, we provide an overview of the training objective and the techniques we employed for training. For the sake of readability, we only discuss the implementation on top of the QMIX algorithm in this section, but it can be easily generalized to the MADDPG algorithm. Pseudo codes for CACOM can be found in Supplementary B.

### 4.1  Communication Protocol Design

The objective of MARL is for agents to choose actions locally so that the joint action will lead to a better team reward. From an agent's perspective, the communication messages it receives can augment its local information, thereby assisting action decisions. To ease the explanation, we call this agent receiving information the *helpee* and the agent providing information the *helper*. Note that in our proposed method, agents are homogeneous, therefore each agent behaves both as a helpee and a helper at the same time.

Thinking from a helpee's perspective, to improve the overall performance, different helpees may need different information to make more sensible decisions. If the helpers send messages in a broadcast fashion, to accommodate the various need for different helpees, this single message has to contain ample information, inevitably
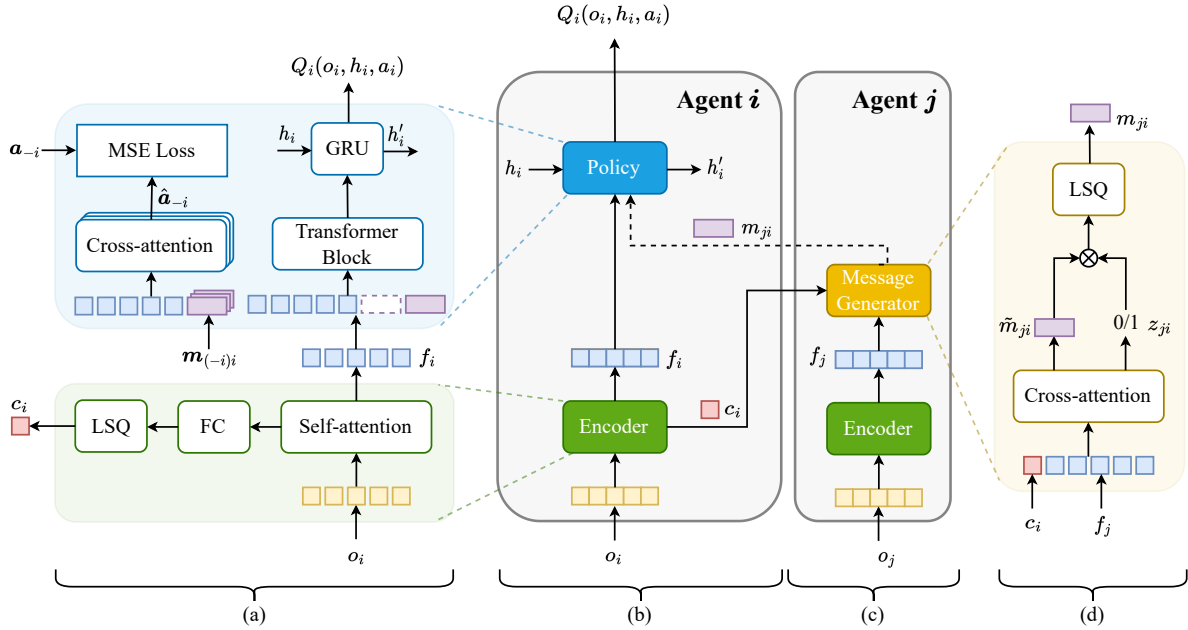
**Figure 3: Network architecture for CACOM. (a) Helpee's feature encoder and policy network. (b) Overall architecture for a helpee agent $i$. (c) Overall architecture for a helper agent $j$. (d) Helper's message generator.**

inducing heavy communication overhead. At the same time, the helpees will end up being flooded with a massive amount of irrelevant information [17], which can even impede the decision making process. To this end, we propose a two-stage communication protocol, namely CACOM, in which the messages are personalized to the helpees' need.

The proposed CACOM is demonstrated in Figure 2. In the first stage, the helpee, agent $i$, broadcasts a short *context message* $c_i$ to all its peers in the system. $c_i$ is a coarse representation of the local information, ideally indicating the available information at agent $i$. Upon receiving the messages from the first stage, a potential helper agent $j$ first decides locally whether it can help the helpee. If agent $j$ decides not to be the helper of agent $i$, then no message will be sent from agent $j$ to agent $i$ in the second stage. Otherwise, agent $j$ will act as a helper and generate a *personalized message* $m_{ji}$ based on the context message $c_i$ from the helpee as well as agent $j$'s local information. After receiving all the messages from the second stage, the helpee $i$ aggregates the information and chooses an action to execute.

## 4.2 Agent Network Design

To implement CACOM, we designed the agent network architecture as depicted in Figure 3. It is composed of three blocks: an encoder, a message generator and a policy network. In particular, the encoder at the helpee agent $i$ takes local observation $o_i$ as input, and outputs the local feature $f_i$ and the context message $c_i$ for the first communication stage. Upon receiving $c_i$, the message generator at the helper agent $j$ determines locally whether to assist agent $i$ and generates the personalized message $m_{ji}$ for the second communication stage if needed. After the second communication stage, the policy network at the helpee agent $i$ aggregates the received

personalized messages $\boldsymbol{m}_{(-i)i}$ and the local feature $f_i$ to calculate the local value.

**Attention Architecture to Generate Personalized Messages:** As discussed before, we aspire for the message generator at the helper side to produce a personalized message for the helpee based on the received context information. This is achieved by the attention-based blocks which encode entities from the observation respectively and generate personalized messages.

At the helpee agent $i$'s side, the encoder treats the observation as tokens of entities, where entities can be ego states, enemy states, teammate states or environment states. By applying self-attention to tokens of entities, the local encoder generates a local feature $f_i \in \mathbb{R}^{m \times d_f}$ ($m$ denotes the number of entities and $d_f$ denotes the encoding dimension for each entity) and a context message $c_i$ for broadcasting. At the helper agent $j$'s side, the message generator utilizes the cross-attention mechanism and takes the received message $c_i$ and the local feature as input and generates the personalized message $m_{ji}$:

$$\tilde{m}_{ji} = \text{softmax}\left(\frac{q_i^T k_j}{\sqrt{d_k}}\right) v_j, \tag{1}$$

$$m_{ji} = \text{LSQ}\left(\tilde{m}_{ji} \cdot z_{ji}\right), \tag{2}$$

where $k_j = W_k f_j \in \mathbb{R}^{d_k \times m}$, $q_i = W_q c_i \in \mathbb{R}^{d_k \times 1}$, $v_j = W_v f_j \in \mathbb{R}^{d_m \times m}$, $d_m$ denotes the message dimension for the second stage communication and the softmax$(\cdot)$ is calculated along the $m$ dimension. $z_{ji}$ is a one-bit value denoting whether to generate a personalized message to $i$, and we delay further details for Equation 2 later in this section.

After receiving context-aware messages from all the helpers, the helpee agent $i$ uses a transformer block to aggregate the received messages $\boldsymbol{m}_{(-i)i}$ and its own local feature $f_i$. This is followed by a gated recurrent unit (GRU) block, which considers historical information $h_i$ and outputs a local value:

$$Q_i(o_i, h_i, \cdot; \theta) = \mathcal{F}\left(f_i, h_i, \boldsymbol{m}_{(-i)i}; \theta\right), \tag{3}$$

where $\mathcal{F}$ denotes the policy function.

**Quantization to Reduce Communication Overhead:** As practical systems adopt digital communication, it is essential to use a differentiable quantizer to digitize the messages. To this end, we adopt LSQ as the quantizer for both communication stages, which offers a simple implementation and high accuracy for gradient estimation. Following the LSQ algorithm, we have

$$m_{ji} = \lfloor \text{clip}(\tilde{m}_{ji} \cdot z_{ji}, -2^{b-1}, 2^{b-1} - 1) \rceil \cdot s_\theta, \tag{4}$$

where $\text{clip}(\cdot, r_1, r_2)$ returns the input with the values below $r_1$ set to $r_1$ and the values above $r_2$ set to $r_2$, $\lfloor \cdot \rceil$ rounds input to the nearest integer, $b$ denotes number of bits and $s_\theta$ denotes the learnable step size. This approach allows for digital communication while preserving end-to-end differential properties.

**Gating Mechanism to Prune Unnecessary Links:** To further reduce the communication overhead and eliminate unnecessary communication links, we utilize the gating mechanism [12] to prune the links for the second communication stage dynamically. The context messages from the first communication stage can naturally serve as indications on how to prune the links in the second communication stage. Specifically, we train a local binary classifier at the potential helper agent $j$'s side to determine whether to send a message at the second stage to a particular helpee $i$, based on the helper's local feature $f_j$ and the context message from the helpee $c_i$.

$$z_{ji} = \mathbb{1}\left[\mathcal{G}(f_j, c_i; \theta_{\mathcal{G}}) > 0.5\right], \tag{5}$$

$$\mathcal{G}(f_j, c_i; \theta_{\mathcal{G}}) = \text{sigmoid}\left(\text{FC}\left(\frac{q_i'^T k_j'}{\sqrt{d_k}}\right)\right), \tag{6}$$

where $k_j' = W_k' f_j \in \mathbb{R}^{d_k \times m}$, $q_i' = W_q' c_i \in \mathbb{R}^{d_k}$, $\mathbb{1}[\cdot]$ is the indicator function, $\text{FC}(\cdot)$ is the fully-connected (FC) layer and $\theta_{\mathcal{G}}$ is the parameters for the gate.

We train the binary classifier in a self-supervised manner, leveraging the centralized value function to generate pseudo labels $y_{ji}$ for local gates during training. In particular, we compare the global value when $z_{ji} = 1$ with the one when $z_{ji} = 0$:

$$y_{ji} = \mathbb{1}\left[Q_{\text{tot}}(\boldsymbol{o}, \boldsymbol{h}, \boldsymbol{a}_{-i}, a_i^0; \theta) - Q_{\text{tot}}(\boldsymbol{o}, \boldsymbol{h}, \boldsymbol{a}_{-i}, a_i^1; \theta) > T\right], \tag{7}$$

where

$$a_i^0 = \arg\max_a Q_i^0(o_i, h_i, a), \tag{8}$$

$$a_i^1 = \arg\max_a Q_i^1(o_i, h_i, a), \tag{9}$$

$$Q_i^0(o_i, h_i, \cdot; \theta) = \mathcal{F}\left(f_i, h_i, \boldsymbol{m}_{(-i)i}; \theta\right)|_{m_{ji}=0}, \tag{10}$$

$$Q_i^1(o_i, h_i, \cdot; \theta) = \mathcal{F}\left(f_i, h_i, \boldsymbol{m}_{(-i)i}; \theta\right)|_{m_{ji}=1}, \tag{11}$$

where $T$ is a threshold.

In this way, we prune the communication links that fail to contribute to the global value.

### 4.3 Training Details

Following the QMIX algorithm, parameters $\theta$ are updated by the TD loss:

$$\mathcal{L}_{\text{TD}}(\theta) = \mathbb{E}_{(\boldsymbol{o}, \boldsymbol{h}, \boldsymbol{a}, r, \boldsymbol{o}', \boldsymbol{h}') \sim \mathcal{D}}\left[(y - Q_{\text{tot}}(\boldsymbol{o}, \boldsymbol{h}, \boldsymbol{a}; \theta))^2\right], \tag{12}$$

$$y = r + \gamma \max_{\boldsymbol{a}'} Q_{\text{tot}}(\boldsymbol{o}', \boldsymbol{h}', \boldsymbol{a}'; \theta^-), \tag{13}$$

where $y$ is the target and $\theta^-$ is the parameters for the target network.

Moreover, since the two-stage communication blocks may enlarge the policy space, making the optimization even more challenging, we introduce an auxiliary loss to prevent the communication from collapsing (i.e., not transmitting any helpful information). We design the auxiliary task to predict the helpers' local values at the helpee's side utilizing the local feature and received personalized messages. Specifically, in addition to the TD loss, we minimize an auxiliary loss given as the mean squared error (MSE) of predictions:

$$\mathcal{L}(\theta_{\mathcal{P}}) = \mathbb{E}_{(h_j, f_i, m_{ji}) \sim \mathcal{D}}\left[(Q_j(h_j, a; \theta) - \mathcal{P}(f_i, m_{ji}; \theta_{\mathcal{P}}))^2\right], \tag{14}$$

where $\theta_{\mathcal{P}}$ denotes the parameters for the communication blocks and the predictor, and $\mathcal{P}$ denotes the predictor function. Note that this auxiliary task is only used to regularize the parameters training in the communication blocks, and the predictions will not be utilized to assist helpee's action decisions.

As previously mentioned, we train the local gate in a self-supervised manner, with synthetic labels generated during training. The local gate $\mathcal{G}$ is trained alternately with the rest of the network. To prevent all communication links from being cut off at the beginning, we disable gate training during the early stages. The loss function for the local gates is defined as follows:

$$\begin{aligned}\mathcal{L}(\theta_{\mathcal{G}}) = \mathbb{E}_{(c_i, f_j) \sim \mathcal{D}}\big[&y_{ji} \cdot \log(\mathcal{G}(f_j, c_i; \theta_{\mathcal{G}}))\\ & - (1 - y_{ji}) \cdot \log(1 - \mathcal{G}(f_j, c_i; \theta_{\mathcal{G}}))\big],\end{aligned} \tag{15}$$

where $y_{ji}$ is generated following Equation 7.

To improve the sample efficiency as well as lower the model complexity, we adopt parameter sharing among agents.

## 5 EXPERIMENTS

In this section, we evaluate our proposed CACOM protocol on three multi-agent cooperative benchmark tasks shown in Figure 4: predator-prey (PP) and cooperation navigation (CN) in MPE [18], and SMAC [25]. As CACOM is agnostic to specific MARL algorithms, we build the proposed communication blocks on top of both value-based algorithms and actor-critic algorithms to test out the applicability. In particular, we follow the literature and use the MADDPG algorithm for the continuous tasks of MPE, and the QMIX algorithm for the discrete task SMAC.

Furthermore, real-world communication systems cannot send continuous signals, so we allocate different communication budgets (in the number of bits per link per timestep) for each scenario based on the observation and action spaces. For CACOM, we use the communication overhead from both stages combined together as the overall communication overhead. We then test out the average rewards or success rates to compare different algorithms. For evaluation, all experiments are averaged with five random seeds and the shaded areas refer to the 95% confidence interval. Details of the
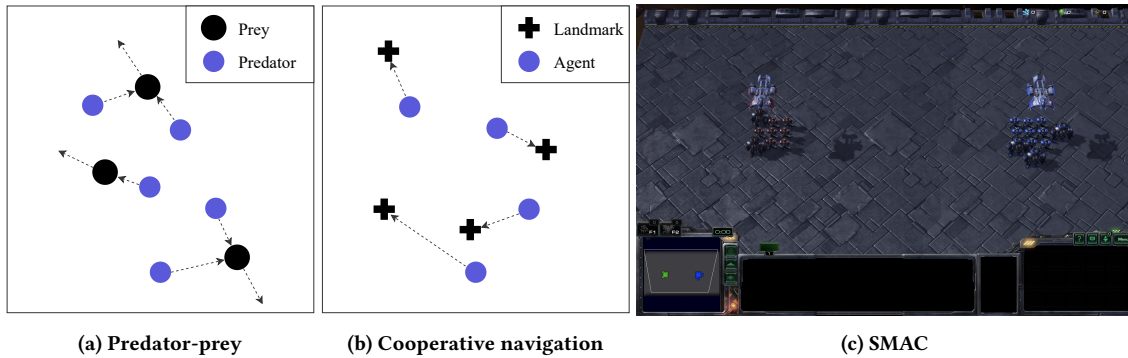
(a) Predator-prey · (b) Cooperative navigation · (c) SMAC

**Figure 4: Multi-agent environments.**

network architectures and training hyperparameters are given in Supplementary B.

## 5.1 MPE

PP is a cooperative MARL benchmark task where the goal for agents (predators) is to capture the moving preys. We set the number of predators to be 10 and the number of preys to be 4 in this experiment. CN requires the agents to occupy stationary landmarks. In this experiment, we set both the number of agents and the number of landmarks to be 8. The world for both scenarios is a continuous space (as opposed to a grid world). The communication budget is set to 24 bits per link per timestep in PP and 32 bits per link per timestep in CN for all algorithms. We include more details on the environmental settings in Supplementary C.

**MARL Algorithms and Baselines:** For this environment, we build CACOM on top of the MADDPG algorithm and then evaluate it against MADDPG, TarMAC + LSQ, I2C + LSQ, and MAIC + LSQ. The baselines, except MADDPG, do not generate discrete messages, and directly pruning the messages would result in unsatisfactory outcomes. To address this issue, we augment TarMAC, I2C, and MAIC with LSQ blocks for fair comparison. In addition, we introduce extra FC layers before and after the communication for MAIC and I2C to enable them to operate within the communication budget. The hyperparameters in both baselines and CACOM were tuned to obtain the best performance.

**Table 1: Gate Pruning Results for MPE**

| scenarios | second-stage links pruned ratio | communication budget occupied ratio |
|---|---|---|
| PP | 36.33(±12.73)% | 75.78(±8.48)% |
| CN | 25.88(±15.81)% | 80.59(±11.85)% |

**Results:** We show the performance of CACOM and baselines on MPE benchmarks in Figure 5 and the corresponding gate pruning results in Table 1. It can be observed from Figure 5 that CACOM outperforms all the baselines in both tasks, which shows the superiority of context-aware communication under low communication budgets.

In both tasks, most communication-enhanced methods exceed the performance of MADDPG, even under severe communication constraints. Compared to the performance in the corresponding original papers [2, 4, 34], where continuous messages are transmitted, we only observe modest performance degradation in these baseline communication methods. This shows great potential in applying quantization methods in MARL to reduce communication overhead. However, in some cases (e.g., I2C in Figure 5b), we see that inappropriate communication protocol design may harm the overall performance by providing messages that fail to assist action decisions at the helpee's side. The comparison between CACOM and other communication methods indicates the superior performance of context-aware communication. And this could attribute to the high flexibility of message generation in such a non-broadcast communication protocol and personalized messages based on the helpee's context.

**Visualization:** In Figure 6, we visualize the policies alongside the corresponding communication overhead in two instances of PP and CN, respectively. We see that for instance shown in the Figure 6a, agents divide into subgroups and surround the preys. The communication overhead remains roughly the same because the observations are constantly changing as the episode progress. In Figure 6b, at the early stage of the episode, each agent's target landmark is unclear, therefore more communication links are established among agents. At a later stage, when each agent already occupies a landmark, the communication overhead drops due to the decreasing need. Overall, it shows that the policies learnt by CACOM demonstrate strong cooperation among agents, and the local gate can dynamically prune the unnecessary links according to the context.

## 5.2 SMAC

We further evaluate CACOM on SMAC [25, 34] across four super hard maps: MMM3, 1c3s5z_vs1c3s6z, 27m_vs_30m and corridor. The communication budget is set to be $2 \times$ n_actions per link per timestep for each scenario.

**MARL Algorithms and Baselines:** For this environment, we build CACOM on top of the QMIX algorithm and evaluate it against QMIX, NDQ, TMC and MAIC + LSQ. Similarly to the experiments in MPE, we augment MAIC with LSQ blocks for quantization. Since NDQ and TMC already consider the communication bandwidth and generate messages in compact ways, we directly adopt the original
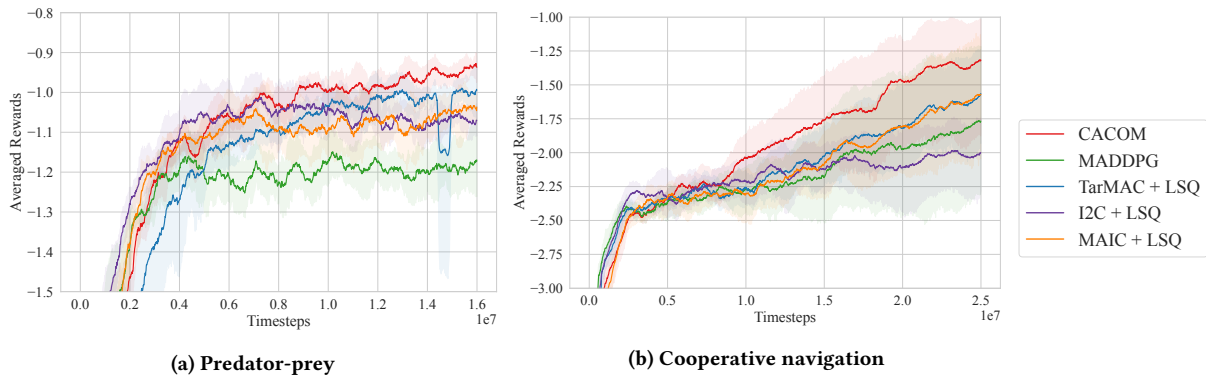
(a) Predator-prey

(b) Cooperative navigation

**Figure 5: Performance comparison with baselines on MPE benchmarks.**
Performance comparison with baselines on MPE benchmarks, where CACOM achieves the best performance.



(a) Predator-prey
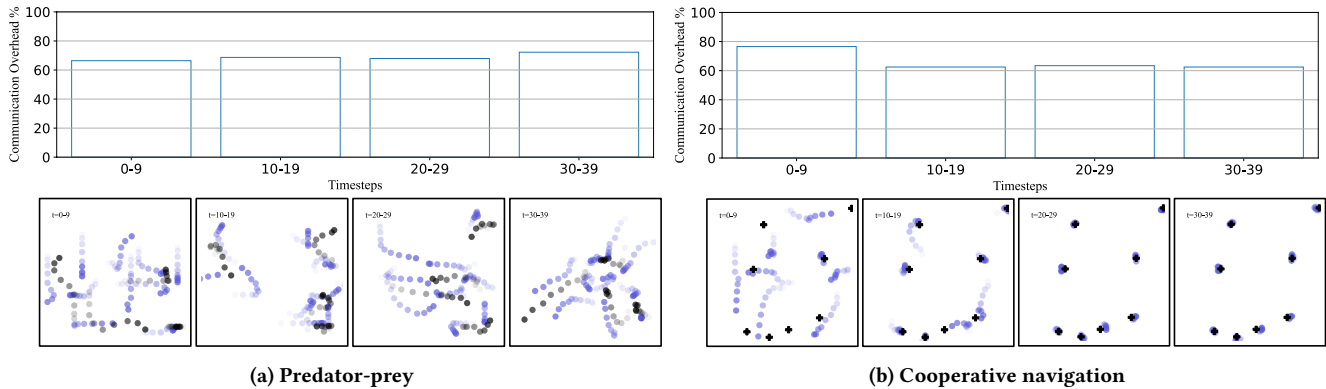
(b) Cooperative navigation

**Figure 6: Visualization of policies in MPE. On the top, we show the communication overhead versus timesteps, in which the communication overhead measures the percentage of the utilized communication budgets. On the bottom, we stack ten timesteps in one figure. The markers correspond to Figure 4, and darker dots correspond to more recent timesteps.**

implementation. We use the hyperparameters in PyMARL [25] for both baselines and CACOM.

**Table 2: Gate Pruning Results for SMAC**

| scenarios | second-stage links pruned ratio | communication budget occupied ratio |
|---|---|---|
| MMM3 | 60.52($\pm$13.23)% | 58.59($\pm$6.19)% |
| 1c3s5z_vs_1c3s6z | 57.36($\pm$18.10)% | 56.98($\pm$10.18)% |
| 27m_vs_30m | 49.96($\pm$4.66)% | 66.69($\pm$2.07)% |
| corridor | 23.05($\pm$4.60)% | 84.63($\pm$2.04)% |

**Results:** We show the performance of CACOM and baselines on SMAC benchmarks in Figure 7 and the corresponding gate pruning results in Table 2. Overall, CACOM consistently outperfoms the baselines. Compared to the MPE, the observations in SMAC are more sophisticated therefore exacerbate the consequences of limited bandwidth. As a result, most baseline communication protocols perform even worse than the communication-free method QMIX under severe bandwidth constraints (e.g., Figure 7b and 7c). In

contrast, CACOM still manages to obtain evident performance gains. This highlights its ability to leverage the limited bandwidth efficiently by transmitting only the most relevant information.

**The Effect of Messages Length:** In Figure 8, we demonstrate how the number of bits for the context message and personalized message influence the performance. From the red curve, we see that increasing context message bits nearly never impedes the training because a longer context message enables better personalized message generation. However, increasing the message size for the first stage may result in heavy communication overhead since the first-stage communication operates in a broadcast fashion.

On the other hand, a very small personalized message size can result in worse performance compared to the case without communication, as the green curve suggests. This may be because personalized messages that are supposed to assist the receivers' decision making fail to provide helpful information under such a low communication budget. And this also implies that CACOM may be further improved when we combine it with more delicate design at the receiver side [7].

**Ablation Studies:** To verify the effectiveness of different design choices in CACOM, we conduct ablation studies on MMM3 and
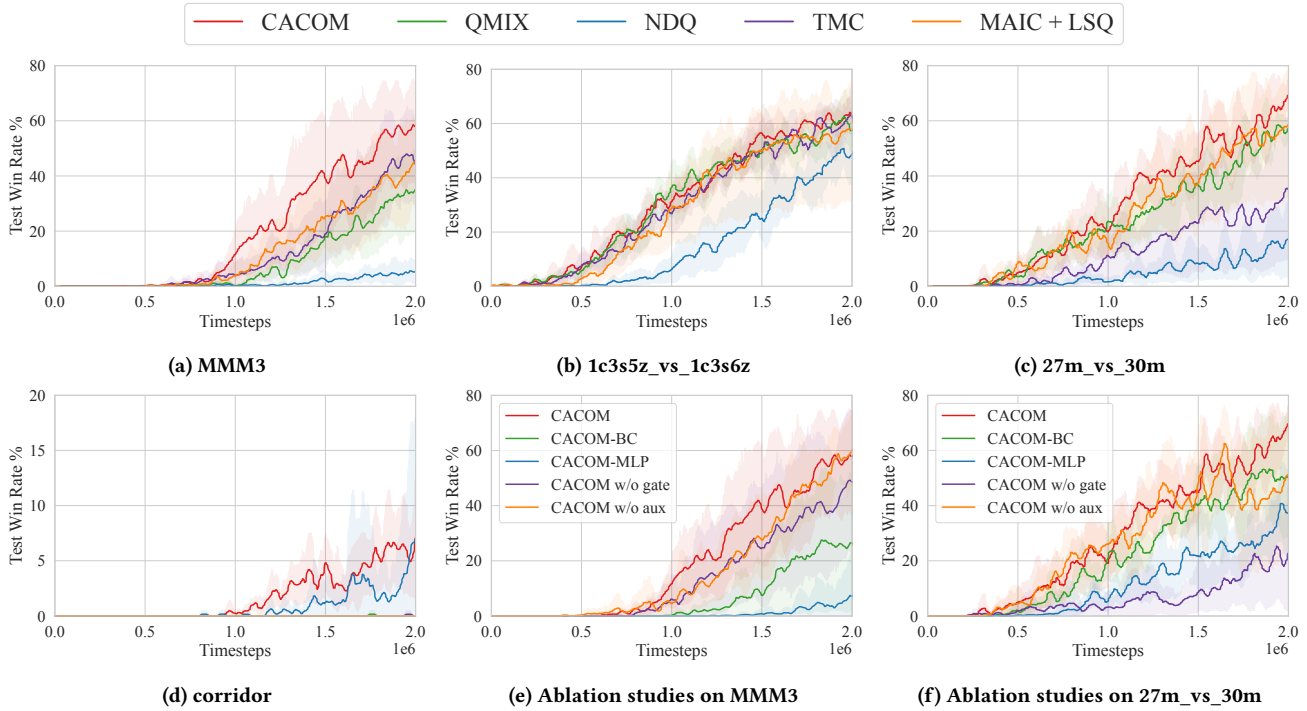
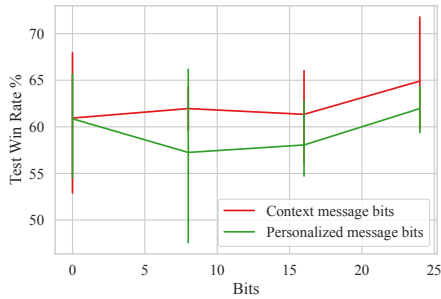Figure 7: Performance comparison with baselines on SMAC benchmarks.



Figure 8: Message size analysis. The **green** curve shows the influence of personalized message bits while the context message is fixed as 8 bits and the **red** curve shows the influence of context message bits while the personalized message is fixed as 24 bits.

27m_vs_30m scenarios, which are illustrated in Figure 7e and 7f. We keep the CACOM network architecture but replace the context-aware communication with broadcasting communication and denote it as CACOM-BC. We replace the attention blocks in the network with multi-layer perceptron (MLP) blocks and denote it as CACOM-MLP. We remove the gating mechanism in CACOM and call it CACOM w/o gate. We remove the auxiliary task and call it CACOM w/o aux. Figure 7e and 7f show that CACOM outperforms CACOM-BC, indicating the better expressiveness achieved by context-aware communication. The unsatisfactory performance

of CACOM-MLP implies that the success of CACOM shall also attribute to the network design. Without the attention mechanism to help identifying the related information, it is difficult for the helpers to learn to generate the personalized messages. Furthermore, the gating mechanism improves the performance by filtering out the unrelated information and the auxiliary task assists the training by stabilizing the training in some scenarios.

**Limitations:** While our method has demonstrated its effectiveness in the above experiments, we have observed that the results of gate-pruning exhibit a relatively large variance (especially in MPE). This suggests that the gate training process may be subject to instability.

## 6 CONCLUSIONS

In this paper, we investigated context-aware communication in communication-constrained MARL. In an endeavor to optimize the limited communication budget, we proposed CACOM as a context-aware communication protocol, advocating a shift from sender-centric broadcasting communication schemes to receiver-centric personalized communication. Particularly, we utilized various attention-based blocks to discern the information beneficial to receivers and employed LSQ blocks to quantize the messages to be transmitted. Empirical results underscore the effectiveness of CACOM in cooperative MARL under low communication budgets. For future research, it is interesting to further enhance the context-aware communication protocols by investigating link scheduling and utilizing historical communication messages to reduce communication overhead.

# REFERENCES

[1] Asim Ansari and Carl F Mela. 2003. E-customization. *Journal of marketing research* 40, 2 (2003), 131–145.

[2] Abhishek Das, Théophile Gervet, Joshua Romoff, Dhruv Batra, Devi Parikh, Mike Rabbat, and Joelle Pineau. 2019. Tarmac: Targeted multi-agent communication. In *International Conference on Machine Learning*. PMLR, 1538–1546.

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 248–255.

[4] Ziluo Ding, Tiejun Huang, and Zongqing Lu. 2020. Learning individually inferred communication for multi-agent cooperation. *Advances in Neural Information Processing Systems* 33 (2020), 22069–22079.

[5] Steven K Esser, Jeffrey L McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S Modha. 2020. Learned Step Size Quantization. In *International Conference on Learning Representations*.

[6] Jakob Foerster, Ioannis Alexandros Assael, Nando De Freitas, and Shimon Whiteson. 2016. Learning to communicate with deep multi-agent reinforcement learning. *Advances in neural information processing systems* 29 (2016).

[7] Cong Guan, Feng Chen, Lei Yuan, Chenghe Wang, Hao Yin, Zongzhang Zhang, and Yang Yu. 2022. Efficient Multi-agent Communication via Self-supervised Information Aggregation. *Advances in Neural Information Processing Systems* 35 (2022), 1020–1033.

[8] Shuai Han, Mehdi Dastani, and Shihan Wang. 2023. Model-based Sparse Communication in Multi-agent Reinforcement Learning. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*. 439–447.

[9] Guangzheng Hu, Yuanheng Zhu, Dongbin Zhao, Mengchen Zhao, and Jianye Hao. 2021. Event-triggered communication network with limited-bandwidth constraint for multi-agent reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems* (2021).

[10] Chicheng Huang, Huaqing Li, Dawen Xia, and Li Xiao. 2016. Distributed consensus of multi-agent systems over general directed networks with limited bandwidth communication. *Neurocomputing* 174 (2016), 681–688.

[11] Jiechuan Jiang, Chen Dun, Tiejun Huang, and Zongqing Lu. 2019. Graph Convolutional Reinforcement Learning. In *International Conference on Learning Representations*.

[12] Jiechuan Jiang and Zongqing Lu. 2018. Learning attentional communication for multi-agent cooperation. *Advances in neural information processing systems* 31 (2018).

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

[14] Matthew W Kreuter, David W Farrell, Laura R Olevitch, and Laura K Brennan. 2013. *Tailoring health messages: Customizing communication with computer technology*. Routledge.

[15] Sheng Li, Jayesh K Gupta, Peter Morales, Ross Allen, and Mykel J Kochenderfer. 2021. Deep Implicit Coordination Graphs for Multi-agent Reinforcement Learning. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*. 764–772.

[16] Dianbo Liu, Alex Lamb, Xu Ji, Pascal Notsawo, Mike Mozer, Yoshua Bengio, and Kenji Kawaguchi. 2022. Adaptive Discrete Communication Bottlenecks with Dynamic Vector Quantization. arXiv:2202.01334 [cs.LG]

[17] Yong Liu, Weixun Wang, Yujing Hu, Jianye Hao, Xingguo Chen, and Yang Gao. 2020. Multi-agent game abstraction via graph attention neural network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 7211–7218.

[18] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems* 30 (2017).

[19] Xueguang Lyu, Yuchen Xiao, Brett Daley, and Christopher Amato. 2021. Contrasting centralized and decentralized critics in multi-agent reinforcement learning. *arXiv preprint arXiv:2102.04402* (2021).

[20] Frans A Oliehoek and Christopher Amato. 2016. *A concise introduction to decentralized POMDPs*. Springer.

[21] Afshin Oroojlooy and Davood Hajinezhad. 2022. A review of cooperative multi-agent deep reinforcement learning. *Applied Intelligence* (2022), 1–46.

[22] James Parker, Ernesto Nunes, Julio Godoy, and Maria Gini. 2016. Exploiting spatial locality and heterogeneity of agents for search and rescue teamwork. *Journal of Field Robotics* 33, 7 (2016), 877–900.

[23] Peng Peng, Ying Wen, Yaodong Yang, Quan Yuan, Zhenkun Tang, Haitao Long, and Jun Wang. 2017. Multiagent bidirectionally-coordinated nets: Emergence of human-level coordination in learning to play starcraft combat games. *arXiv preprint arXiv:1703.10069* (2017).

[24] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2020. Monotonic value function factorisation for deep multi-agent reinforcement learning. *The Journal of Machine Learning Research* 21, 1 (2020), 7234–7284.

[25] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim G. J. Rudner, Chia-Man Hung, Philiph H. S. Torr, Jakob Foerster, and Shimon Whiteson. 2019. The StarCraft Multi-Agent Challenge. *CoRR* abs/1902.04043 (2019).

[26] Sven Seuken and Shlomo Zilberstein. 2007. Improved memory-bounded dynamic programming for decentralized POMDPs. In *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence*. 344–351.

[27] Amanpreet Singh, Tushar Jain, and Sainbayar Sukhbaatar. 2019. Individualized controlled continuous communication model for multiagent cooperative and competitive tasks. In *International conference on learning representations*.

[28] Sainbayar Sukhbaatar, Rob Fergus, et al. 2016. Learning multiagent communication with backpropagation. *Advances in neural information processing systems* 29 (2016).

[29] Gokul Swamy, Siddharth Reddy, Sergey Levine, and Anca D Dragan. 2020. Scaled autonomy: Enabling human operators to control robot fleets. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 5942–5948.

[30] Tonghan Wang, Jianhao Wang, Chongyi Zheng, and Chongjie Zhang. 2019. Learning nearly decomposable value functions via communication minimization. *arXiv preprint arXiv:1910.05366* (2019).

[31] Yuanfei Wang, Fangwei Zhong, Jing Xu, and Yizhou Wang. 2021. Tom2c: Target-oriented multi-agent communication and cooperation with theory of mind. *arXiv preprint arXiv:2111.09189* (2021).

[32] Wang Ying and Sang Dayong. 2005. Multi-agent framework for third party logistics in E-commerce. *Expert Systems with Applications* 29, 2 (2005), 431–436.

[33] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. 2021. The surprising effectiveness of ppo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955* (2021).

[34] Lei Yuan, Jianhao Wang, Fuxiang Zhang, Chenghe Wang, Zongzhang Zhang, Yang Yu, and Chongjie Zhang. 2022. Multi-agent incentive communication via decentralized teammate modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 9466–9474.

[35] Lei Yuan, Ziqian Zhang, Lihe Li, Cong Guan, and Yang Yu. 2023. A Survey of Progress on Cooperative Multi-agent Reinforcement Learning in Open Environment. *arXiv preprint arXiv:2312.01058* (2023).

[36] Sai Qian Zhang, Qi Zhang, and Jieyu Lin. 2019. Efficient communication in multi-agent reinforcement learning via variance based control. *Advances in Neural Information Processing Systems* 32 (2019).

[37] Sai Qian Zhang, Qi Zhang, and Jieyu Lin. 2020. Succinct and robust multi-agent communication with temporal message control. *Advances in Neural Information Processing Systems* 33 (2020), 17271–17282.