ReSCOM: Reward-Shaped Curriculum for Efficient Multi-Agent Communication Learning

Xinghai Wei
Beijing University of Posts and Telecommunications
Beijing, China
junjuntvt@bupt.edu.cn

Tingting Yuan*
University of Göttingen
Göttingen, Germany
tingting.yuan@cs.uni-goettingen.de

Jie Yuan
Beijing University of Posts and Telecommunications
Beijing, China
yuanjie@bupt.edu.cn

Dongxiao Liu
Beijing University of Posts and Telecommunications
Beijing, China
liudongxiao@bupt.edu.cn

Xiaoming Fu
University of Göttingen
Göttingen, Germany
fu@cs.uni-goettingen.de

ABSTRACT

Communication enhances collaboration among artificial intelligence agents. Given the conflicts between limited communication resources and communication needs, learning effective communication strategies is essential. We observe that incorporating learning to communicate can complicate mastering primary tasks. This is due to the uncertainty in information acquisition during the learning process, which can lead to an unstable environment for primary tasks. In this paper, we introduce ReSCOM, an efficient joint learning framework that combines learning-to-communicate with primary tasks. ReSCOM progressively adjusts the learning emphasis through reward-shaped curriculums, allowing agents to shift their focus from primary tasks and basic communication tasks (e.g., how to encode) to advanced communication strategies (e.g., determining when it is worthwhile to communicate). This approach minimizes the impact on the learning efficiency of primary tasks while simultaneously facilitating communication learning. We evaluate ReSCOM against state-of-the-art methods across various tasks, and experimental results demonstrate its effectiveness.

KEYWORDS

Learn to communicate; curriculum learning; reward shaping; agent collaboration

ACM Reference Format:
Xinghai Wei, Tingting Yuan, Jie Yuan, Dongxiao Liu, and Xiaoming Fu. 2025. ReSCOM: Reward-Shaped Curriculum for Efficient Multi-Agent Communication Learning. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 9 pages.

*Corresponding author.

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

1 INTRODUCTION

Decentralized intelligence has proven to offer significant advantages across various modern applications, including smart grid control [4, 31, 38], wireless communication [17], autonomous driving [16], and network management [39]. However, agents often encounter difficulties when tackling complex tasks independently due to partial and limited information. To overcome these challenges, emerging applications require the use of collective intelligence (CI) to facilitate cooperation among agents [5, 27]. By integrating the capabilities of multiple devices, CI enables agents to achieve shared intentions and joint objectives.

Communication among agents is considered crucial for promoting cooperation and building CI [6, 24, 30]. Through information sharing, agents can acquire observations and actions from others to perceive the environment more effectively, leading to improved decision-making and more stable cooperation [28]. However, the ideal fully communicative environment often does not exist due to limited environmental resources. For instance, in vehicular networks, constant information exchange between vehicles rapidly depletes limited bandwidth resources. More importantly, not every message can provide useful information. Redundant information could make communication barely help and even jeopardize the learning process [8, 14, 19, 41]. Therefore, agents are expected to learn appropriate communication strategies. In recent years, various methods have been developed to optimize agent communication, enabling agents to decide how, when, and with whom to communicate. This process, referred to as learning to communicate (L2C), employs several components for information compression [32] and message filtering [22, 29].

While L2C can foster cooperation among agents, it also presents a complex learning task requiring the optimization of multiple components. Without a suitable learning framework, the uncertainty introduced during the L2C process can hinder the agent's exploration and negatively impact the learning of the primary task [24, 40]. Unfortunately, existing L2C schemes almost exclusively employ the end-to-end differentiable training paradigm, where gradients are propagated across the entire model, and the agent's policy network as well as all L2C components are trained simultaneously. This may results in slow convergence and suboptimal performance when the training samples are limited [36, 42].

In this paper, we introduce ReSCOM, an efficient joint learning framework that integrates L2C with multi-agent reinforcement learning (MARL) as primary tasks. ReSCOM features well-designed reward-shaped curriculums specifically tailored for L2C. It guides the learning process from basic communication tasks, such as learning to encode observations, to mastering communication strategies, such as determining when to communicate. Our principal contributions are summarized as follows:

- To the best of our knowledge, this is the first work to design a curriculum learning approach for L2C in multi-agent systems. By introducing a curriculum for L2C, ReSCOM reduces the impact of L2C on the learning of primary tasks by 16.3% to 21.9% compared to the baselines.
- We design three reward-shaping methods specifically for L2C in MARL, including one discrete curriculum and two continuous curricula. It enables effective L2C training compared to training all L2C components from scratch. Besides, bandwidth consumption during training remains stable, and with continuous curricula, it can actually decrease when the number of agents exceeds 10 in our scenarios.
- We implement ReSCOM and test its performance in various multi-agent cooperative tasks. Experimental results demonstrate that ReSCOM accelerates the convergence of L2C components and significantly enhances performance compared to state-of-the-art L2C methods.

2 RELATED WORK

2.1 Learn to communicate

Efficient communication is widely recognized as essential for fostering cooperation among agents. DIAL [9] is considered the first L2C scheme via backpropagation. IC3Net [29] introduces a gating mechanism that takes the agent's hidden state to output a binary action to indicate whether the agent should initiate communication in this step. This mechanism has also been utilized in other works such as ATOC [14] and GACML [22]. TMC [41] applies a temporal smoothing technique to encourage agents to reduce the transmission of temporally correlated messages and send out new messages only when the current message contains relatively new information compared to the previously sent message. SchedNet [15] addresses the issue of shared communication mediums by employing a weight-based scheduling mechanism to decide which agents can use the communication medium to broadcast their messages during a given time interval. TarMAC [7] adopts a targeted multi-round communication architecture, and employs a signature-based soft attention mechanism to achieve effective message aggregation. I2C [8] learns one-to-one communication to reduce information redundancy, where each agent learns prior knowledge via causal inference to determine whom to communicate with.

2.2 Curriculum learning in MARL

Curriculum learning is a training strategy for machine learning that trains from easier data (tasks) to harder data (tasks), imitating human curricula [12, 34, 35]. This approach has been extensively used in MARL to tackle challenges such as sparse rewards and scalability. The essence of curriculum learning lies in the design of a suitable curriculum scheduler that dynamically adjusts the learning difficulty during the training process. To achieve this, a common strategy is to adjust the environment settings. For instance, EPC [20] and DyMA-CL [33] utilize entity progression technique to incrementally increase the number of agents being trained simultaneously [20, 33], while VACL [3], CD-DDPG [11] and Genet [37] employ task expansion that directly escalates the complexity of the task. An alternative method, referred to as reward-shaped curriculum learning, involves dynamically modifying the reward signals to steer agents toward more efficient exploration and improved strategy development [10]. This method operates under the assumption that not all experiences carry equal importance throughout the training process. Similar to human learning, when an agent attempts to learn a new skill, they may not remember every aspect of the learning process, but rather focus on the moments that offer critical insight allowing them to advance their understanding of the skill [2, 23].

3 PRELIMINARIES

3.1 MARL with communication

In this paper, We consider the on-demand communication scenario, where agents receive messages from other agents for decision-making only when necessary. We model MARL with communication as a partially observable markov decision process [1, 13, 25], defined as a tuple $(N, S, A, M, O, P, \Omega, R, \gamma)$, which includes the number of agents N, the space of global states S, the set of action spaces $A = \{A^1, \ldots, A^N\}$, the set of message spaces $M = \{M^1, \ldots, M^N\}$ and the set of observation spaces $O = \{O^1, \ldots, O^N\}$. At time step t, an agent i observes a partial view o_t^i of the underlying global state s_t following the observation function $\Omega : S \to O$, and generate a message m_t^i. When agent i needs to communicate for cooperation, it broadcasts a request to the entire network. All agents receiving this request forward their message to agent i. The agent then chooses an action $a_t^i \in A^i$ based on its own policy π_{θ^i} parameterized by θ^i. Given the joint actions of all N agents $a_t = \{a_t^1, \ldots, a_t^N\}$, the transition function $P : S \times A \to S$ maps the current state s_t and the set of agent actions a_t to a distribution over the next state s_{t+1}. Finally, each agent receives an individual reward $r_t^i \in R(s_t, a_t)$ where $R : S \times A \to \mathbb{R}$, and the joint rewards are represented as $r_t = \{r_t^1, \ldots, r_t^N\}$. In this paper, we consider a fully cooperative environment in which the objective is to maximize the total expected return of all agents, expressed as:

$$\underset{\pi: S \to A \times C}{maximize} \mathbb{E}_{s_t \sim P(s_{t-1}), a_t, m_t \sim \pi}[\sum_{i \in N} \sum_{t \in T} \gamma^t r_t^i], \tag{1}$$

where T is the finite time horizon, γ is the discount factor, and $\pi = \{\pi_{\theta^1}, \ldots, \pi_{\theta^N}\}$ is the set of agent policy. Note that in MARL with communication, agents' policies are responsible not only for action decisions but also for a series of communication-related decisions.

3.2 Centralized Learning with Decentralized Execution

Centralized learning with decentralized execution (CLDE) is a commonly used architecture in MARL [21], where a centralized critic is responsible for updating the decentralized policies during training. The role of the critic is to provide more accurate feedback to the

individual actors with a limited observation range. The objective of critic is to learn the action-value function $Q^{\phi}(o_t, a_t)$ by minimizing the loss:

$$\mathcal{L}(\phi) = \mathbb{E}_{o_t,a_t}[(y - Q^{\phi}(o_t, a_t))^2], \tag{2}$$

where $y = r_t + \gamma Q^{\phi}(o_{t+1}, a')$, a' is given by actor with the subsequent observation o_{t+1}, and ϕ is the parameter associated with Q. In this case, each agent's policy π_{θ^i} can be updated using the following equation:

$$\nabla_{\theta^i} \mathcal{J}(\pi_{\theta^i}) = \mathbb{E}_{o_t,a_t}[\nabla_{\theta^i} \pi_{\theta^i}(o_t^i) \nabla_{a_t^i} Q^{\phi}(o_t, a_t)|_{a_t^i = \pi_{\theta^i}(o_t^i)}]. \tag{3}$$

3.3 L2C in MARL

L2C entails learning a series of sub-tasks, each corresponding the optimization of a set of L2C components. In this paper, we primarily focus on the following two key learning sub-tasks:

(1) **Learning How to Communicate**: This task focuses on how to make messages effectively represent environmental information, thereby better assisting agents in decision-making. This concept is highly relevant in the real world; for instance, humans learn a common language to exchange information, and vehicles use specific communication protocols to achieve coordination [18, 26]. Typically, this task is achieved through an encoder and an aggregator. The encoder extracts the agent's observations into messages, while the aggregator combines multiple messages from other agents to assist agents in decision-making. The objective of learning how to communicate is to maximize the average reward by learning effective representations of environmental information, which aligns with the optimization goal defined in Eq. (1).

(2) **Learning When to Communicate**: Learning when to communicate can reduce communication overhead and prevent redundant information from interfering with agent decision-making, thereby enhancing performance. This can be accomplished by pruning out unnecessary messages, meaning agents initiate communication only when necessary [22, 29]. The objective of learning when to communicate is to minimize communication overhead while maintaining system performance. To achieve this goal, a revised reward function $R_c(s_t, a_t, g_t)$ is introduced, denoted as:

$$r_{c,t}^i = r_t^i(s_t, a_t^i) - c g_t^i, \tag{4}$$

where $r_{c,t}^i \in R_c(s_t, a_t, g_t)$ is the revised individual reward for agent i, c represents the communication cost that is usually related to the bandwidth consumption generated by communication, $g_t = \{g_t^1, \ldots, g_t^N\}$ is the joint actions of agents' L2C components, and $g_t^i \in \{0, 1\}$ denotes whether agent i initiates communication at time step t. Overall, cg_t^i is the penalty term. This penalty term only takes effect when $g_t^i = 1$, i.e., when the agent i initiates communication. The objective of learning when to communicate can be regarded as the maximizing of the expected accumulative reward $r_{c,t}$, i.e.,

$$\underset{\pi: S \to A \times C}{maximize}\, \mathbb{E}_{s_t \sim P(s_{t-1}), a_t, m_t, g_t \sim \pi}[\sum_{i \in N} \sum_{t \in T} \gamma^t r_{c,t}^i]. \tag{5}$$

4 METHOD

We propose ReSCOM, a communication architecture with progressive learning capabilities. ReSCOM is an extension of the MADDPG architecture, which employs the centralized learning and decentralized execution paradigm. Our design can be broken down into two domains: the L2C component that learns how (encoder and aggregator) and when to communicate (gate), as illustrated in Fig. 1, and the curriculum scheduler that guides agents to develop the correct communication strategy step-by-step, which we will discuss later. Since the optimization objectives for learning how to communicate and learning when to communicate are different, two critic networks, denoted as Critic I and Critic II, are introduced to train the L2C components related to these two sub-tasks, where Critic I is responsible for updating agent's encoder and aggregator, and Critic II is responsible for updating agent's gate. Additionally, Critic I is also responsible for optimizing the primary task, i.e., updating the agent's actor.

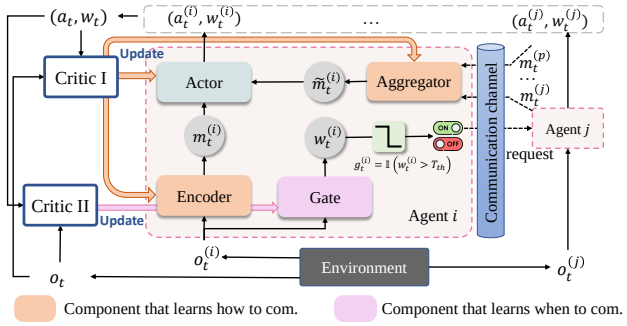


Figure 1: Overview of ReSCOM

4.1 Structure of Agent

As shown in Fig. 1, each agent comprises the following four parts:

(1) **Message Encoder**: Extracts agent i's local observation into a message that encodes both local observation and action intention [14, 26], denoted as $m_t^i = \mu_{enc}^i(o_t^i; \theta_{enc}^i)$, where μ_{enc}^i is the encoding function parameterized by θ_{enc}^i.

(2) **Aggregator**: Aggregates multiple messages into a consolidated message $\tilde{m}_t^i$, denoted as $\tilde{m}_t^i = \mu_g^i(m_t^{i-}; \theta_g^i)$, where $m_t^{i-} = \{m_t^1, \ldots, m_t^{i-1}, m_t^{i+1}, \ldots m_t^N\}$ represents the set of messages from all $n-1$ agents expect agent i, and μ_g^i is the aggregation function parameterized by θ_g^i.

(3) **Actor**: Makes action decisions based on local message m_t^i and aggregated message $\tilde{m}_t^i$, denoted as $a_t^i = \mu_{as}^i(m_t^i, \tilde{m}_t^i; \theta_{as}^i)$, where μ_{as}^i is the message-action function parameterized by θ_{as}^i.

(4) **Gate**: Determines the necessity of communication in the current time step based on the agent's local observation, represented as $w_t^i = \mu_{wg}^i(o_t^i; \theta_{wg}^i)$, where μ_{wg}^i is the observation-weight function parameterized by θ_{wg}^i.

The workflow of these four modules is as follows: At every time step t, based on local observations o_t^i, agent i utilizes the message encoder and the gate to get a local message m_t^i and a weight w_t^i, respectively. Then, based on w_t^i, agent i determines the necessity of communication, represented as:

$$g_t^i = \mathbb{I}(w_t^i > T_{th}), \tag{6}$$

where $\mathbb{I}$ is the indicator function, and T_{th} represents the communication threshold. If $w_t^i > T_{th}$, then $g_t^i = 1$, and agent i decides to communicate and broadcasts a request to the entire network. All agents receiving this request will response their local messages

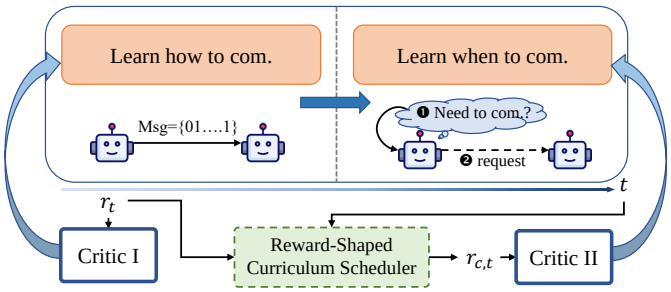


Figure 2: Curriculum scheduler in ReSCOM

to agent i, which will then use the aggregator to generate the aggregated message $\tilde{m}_t^i$. Conversely, if agent i decides that communication is not necessary, the aggregated message $\tilde{m}_t^i$ will be set to a zero vector. Utilizing the above information, the actor is responsible for making action decisions.

4.2 Curriculum Scheduler

The four components mentioned above exhibit strong dependence. For example, when the policy of the gate changes, the distribution of incoming messages is altered, directly affecting the actor and aggregator. Consequently, the actor must adjust to these changes, which subsequently influences the encoder. Furthermore, the updates of the encoder and actor trigger the update of the gate again. As a result, simultaneous training of these components is particularly challenging and leads to reduced training efficiency.

Inspired by curriculum learning [34], our insight is to decompose the simultaneous training of multiple components to reduce learning complexity by dividing the training process into multiple stages, where the agent focuses on optimizing different components at each stage. We split the L2C components in ReSCOM into two parts: the first part includes the encoder, and aggregator, which learn how to communicate; the second part includes the gate, which learns when to communicate, as depicted in Fig. 1. We believe that learning how to communicate should precede learning when to communicate. This is because an agent can only evaluate the benefits of communication to decide whether to communicate after mastering the representation of meaningful environment information.

A reward-shaped curriculum scheduler is introduced to achieve the above goal, as shown in Fig. 2. This component receives the global reward r_t as input and generates the revised reward $r_{c,t}$ as the feedback signal for Critic II. By continuously adjusting the revised reward $r_{c,t}$ throughout the training process, ReSCOM dynamically refines the tasks the agent focuses on, facilitating a transition from learning how to communicate to learning when to communicate. This is achieved by modifying the definition of $r_{c,t}^i$ via setting λ, which can be expressed as follows:

$$r_{c,t}^i = r_t^i(s_t, a_t^i) - \lambda(t) c g_t^i, \tag{7}$$

where $\lambda(t)$ is the scheduling function defined to map the training step number t to a scalar $\lambda \in (0, 1]$, representing the difficulty of learning at step t. Specifically, $\lambda(t)$ should be monotonic and non-decreasing, starting at $\lambda(0) \geq 0$ and ending at $\lambda(\tau) = 1$, where τ denotes the step at which the function first reaches 1. Any function meeting these criteria can serve as $\lambda(t)$ function. The intuitive interpretation of Eq. (7) is as follows: During the initial stages of training, the communication penalty term is relatively small. Consequently, agents tend to adopt a full communication strategy to rapidly optimize their encoders and aggregators, achieving learning how to communicate. As the training period extends, the communication penalty term incrementally increases, prompting the agents to learn when to communicate to reduce unnecessary communication overhead, thus optimizing the gate.

We consider the following two methods to set λ:

(1) **Discrete Scheduler**: The training process is divided into two distinct phases. During the first phase, an easier curriculum is applied, where agents adopt a full communication strategy to learn how to communicate. After a fixed number of training steps τ, the second phase begins, during which agents start learning when to communicate. Following the definition of $\lambda(t)$ above, discrete scheduling can be expressed as:

$$\lambda_d(t) = \begin{cases} 0, & \text{if } t < \tau, \\ 1, & \text{if } t \geq \tau. \end{cases}$$

Essentially, the design of the discrete scheduler can be viewed as "delayed learning of when to communicate". Hence, in the subsequent text, we use ReSCOM (delay) to refer to ReSCOM with the discrete curriculum scheduler.

(2) **Continuous Scheduler**: In contrast to the discrete scheduler that separates the training phase into several distinct parts, the continuous scheduler employs a continuous scheduling function to progressively amplify the communication penalty, shifting the focus of optimization from learning how to communicate to learning when to communicate gradually. We give two examples as follows:

- *Linear function*, denoted as $\lambda_l(t) = \min(1, \lambda_0 + \frac{1-\lambda_0}{\tau} * t)$.
- *Root-p function*, denoted as $\lambda_r(t) = \min(1, (\frac{1-\lambda_0^p}{\tau} * t + \lambda_0^p)^{1/p})$.

Here, $\lambda_0 \in [0, 1)$ is the initial difficulty of learning task at $t = 0$, and p is a hyperparameter. We refer to the ReSCOM using the two aforementioned functions as ReSCOM (linear) and ReSCOM (root-p), respectively. Specifically, in this paper, we set p to 2. The impact of various $\lambda(t)$ function settings on learning will be explored in future work.

The above scheduling functions are illustrated in Fig. 3 with $\tau = 100$ and $\lambda_0 = 0$ as an example. Note that training without the curriculum scheduler (denoted as "baseline") is also regarded as a special case, whose $\lambda(t)$ value is always 1.

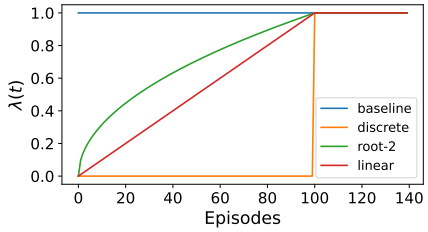


Figure 3: Visualization of different scheduling functions.

4.3 Training

For training ReSCOM, an experience reply buffer is required. The experience reply buffer $\mathcal{R}$ contains tuples $< o_t, a_t, r_t, r_{c,t}, o_{t+1}, w_t >$

recording the experiences of all agents. Leveraging two critic networks, i.e., Critic I and Critic II, ReSCOM learns two action-value functions $Q^{\phi_I}(o_t, a_t) = \mathbb{E}_{o_t,a_t}[R(o_t, a_t)]$ and $Q^{\phi_{II}}(o_t, a_t, w_t) = \mathbb{E}_{o_t,a_t,w_t}[R_c(o_t, a_t, g_t)|_{g_t=\mathbb{I}(w_t>T_{th})}]$ for updating L2C components as well as the actors.

Update ϕ_I and ϕ_{II}: Leverage Eq. (2), the weights ϕ_I and ϕ_{II} are updated by minimizing $\mathcal{L}(\phi_I)$ and $\mathcal{L}(\phi_{II})$, respectively:

$$\mathcal{L}(\phi_I) = \mathbb{E}_{o_t,a_t,r_t,o_{o+1}\sim\mathcal{R}}[r_t + \gamma Q^{\phi_I}(o_{t+1}, a') - Q^{\phi_I}(o_t, a_t)]^2, \quad (8)$$

$$\mathcal{L}(\phi_{II}) = \mathbb{E}_{o_t,a_t,r_{c,t},w_t,o_{o+1}\sim\mathcal{R}}[r_{c,t} + \gamma Q^{\phi_{II}}(o_{t+1}, a', w') - Q^{\phi_{II}}(o_t, a_t, w_t)]^2, \quad (9)$$

where a' and w' are given by actors and gates with the subsequent observations o_{t+1}, respectively.

Update θ^i_{as}: The parameters of agent i's action selector θ^i_{as} are updated using the deterministic policy gradient, expressed as:

$$\nabla_{\theta^i_{as}}\mathcal{J}(\theta^i_{as}) = \mathbb{E}_{o_t,a_t\sim\mathcal{R}}[\nabla_{\theta^i_{as}}\mu^i_{as}(m^i_t, \tilde{m}^i_t) \nabla_{a^i_t}Q^{\phi_I}(o_t, a_t)|_{a^i_t=\mu^i_{as}(m^i_t,\tilde{m}^i_t)}]. \quad (10)$$

Update θ^i_{enc} and θ^i_g: By applying the chain rule, the aggregator and encoder can be optimized jointly using back-propagation, expressed as:

$$\nabla_{\theta^i_g}\mathcal{J}(\theta^i_g) = \mathbb{E}_{o_t,a_t\sim\mathcal{R}}[\nabla_{\theta^i_g}\mu^i_g(m^{i-}_t) \nabla_{\tilde{m}^i_t}\mu^i_{as}(m^i_t, \tilde{m}^i_t)|_{\tilde{m}^{(i)}_t=\mu^i_g(m^{i-}_t)} \nabla_{a^i_t}Q^{\phi_I}(o_t, a_t)|_{a^i_t=\mu^i_{as}(m^i_t,\tilde{m}^i_t)}], \quad (11)$$

$$\nabla_{\theta^i_{enc}}\mathcal{J}(\theta^i_{enc}) = \mathbb{E}_{o_t,a_t\sim\mathcal{R}}[\nabla_{\theta^i_{enc}}\mu^i_{enc}(o^i_t) \nabla_{m^i_t}\mu^i_{as}(m^i_t, \tilde{m}^i_t)|_{m^i_t=\mu^i_{enc}(o^i_t)} \nabla_{a^i_t}Q^{\phi_I}(o_t, a_t)|_{a^i_t=\mu^i_{as}(m^i_t,\tilde{m}^i_t)}]. \quad (12)$$

Update θ^i_{wg}: The gradient for updating the parameters of θ^i_{wg} can be written as:

$$\nabla_{\theta^i_{wg}}\mathcal{J}(\theta^i_{wg}) = \mathbb{E}_{o_t,a_t,w_t\sim\mathcal{R}}[\nabla_{\theta^i_{wg}}\mu^i_{wg}(o^i_t) \nabla_{w^i_t}Q^{\phi_{II}}(o_t, a_t, w_t)|_{w^i_t=\mu^i_{wg}(o^i_t)}]. \quad (13)$$

5 EVALUATION

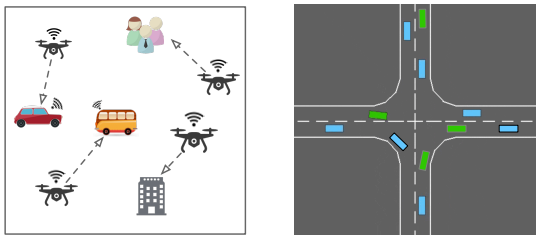
In this section, we first compare ReSCOM with existing L2C schemes across multiple collaborative tasks. Subsequently, we deep dive into the reason why ReSCOM is effective. Finally, we conduct the ablation study.

5.1 Environments

We evaluate ReSCOM in the following environments:

Drone coverage [1]: In this task, several drones work together to navigate towards stationary landmarks while avoiding collisions. Each drone operates as an agent with limited observation that contains the relative position of k nearest landmarks and other agents. The agents exhibit heterogeneity in their observation abilities, represented by the vector $K = \{k_i\}_{i=1}^n$, where k_i denotes the observation range of the i-th agent. Agents are rewarded based on their proximity to the landmarks, penalized for collisions, and receive bonuses for covering landmarks. The performance of ReSCOM is evaluated in scenarios with four agents and ten agents. In the four-agent scenario, the observation ranges are set to [0, 0, 2, 4]. In the ten-agent scenario, the observation ranges are set to [0, 0, 0, 0, 1, 1, 2, 2, 4, 4].

Highway [2]: In this task, multiple vehicles try to quickly traverse an intersection to reach their respective destinations while avoiding collisions. We simulate four vehicles entering the intersection from different directions, with a random initial position on the road. Similar to the previous scenario, the agents obtain different observation abilities, represented as $K = [1, 1, 1, 4]$, where $k_i = n, k_i \in K$ means that agent i can observe the position and speed of other n nearest vehicles. The agents receive a punishment if a collision occurs and a bonus if successfully reach the destination in time.



(a) Drone coverage (b) Highway

Figure 4: Multi-agent environment.

5.2 Baselines

We compare ReSCOM with the following methods:

- GACML [22]: GACML employs a gate component similar to the one used in this paper to filter out redundant information. The benefits of communication are estimated using the Q-network, and GACML prunes the messages that contribute smaller Q-values than the threshold T_G. In our experiments, we set T_G as 0.2.
- SchedNet [15]: SchedNet is designed for scenarios involving shared communication mediums. In each time step, a fixed number of agents (l) can broadcast messages, while the other agents can receive these messages to aid in decision-making. In our experiments, we set $l = \frac{1}{2}N$, meaning that in each time step, half of the agents can broadcast messages.
- TarMAC [7]: TarMAC focus on learning how to communicate. Its original version employs a full communication strategy. We consider a variant, TarMAC+IC3Net, where IC3Net [29] uses the gate mechanism to determine when to communicate. This gate is a simple network containing a softmax layer for two actions, i.e., communicate or not.

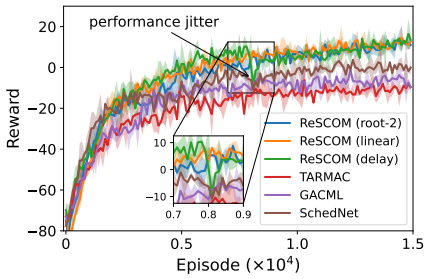
Note that our approach focuses on learning how to communicate and when to communicate. Therefore, approaches that consider learning whom to communicate with, such as ATOC [14] and SOG [28], are not considered for comparison.

5.3 Hyperparameters setting

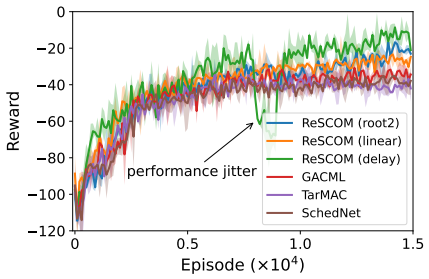
We implement ReSCOM as an extension of MADDPG [21]. All agents share the same parameters. We use an Adam optimizer with a learning rate of 0.005. The discount factor for reward, γ, is 0.95. For the soft update of target networks, we set $\xi = 0.01$. We use a

[1] https://github.com/openai/multiagent-particle-envs.

[2] https://github.com/eleurent/highway-env.

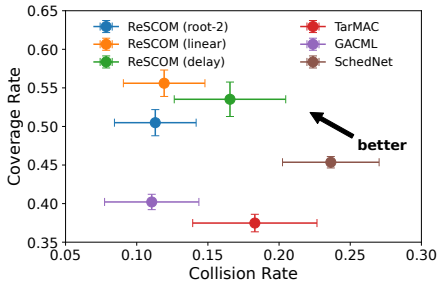


(a) 4 Agents

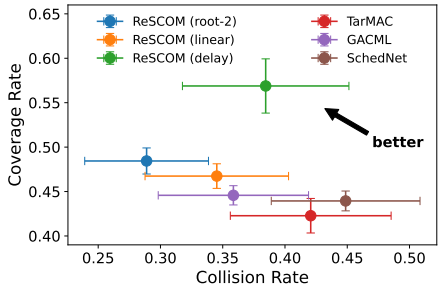


(b) 10 Agents

Figure 5: ReSCOM vs. baselines.

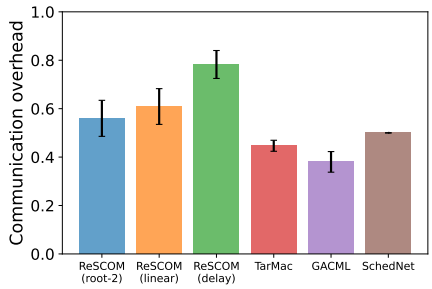


(a) 4 Agents

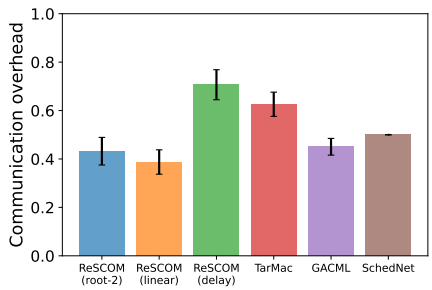


(b) 10 Agents

Figure 6: Performance breakdown.



(a) 4 Agents



(b) 10 Agents

Figure 7: Communication overhead

three-layer multilayer perceptron (MLP) with 64 units to implement the actor, the critic, and the gate. All the above neural networks use ReLU as activation functions. The aggregator is implemented with an attention-based unit, with the number of heads set to 8 and a dropout rate of 0. The capacity of the replay buffer is 10^5, and we take a minibatch of 1024 to update the network parameters. The cost of communication c is set to 0.2, the communication threshold T_{th} is set to 0.5, and the dimensions of messages, outputted by the encoder and the aggregator, are 12 floating-point values. For the three different types of ReSCOM, we set τ to 8000 and λ_0 to 0. The choice of the τ value will be discussed later. We initialize the parameters of neural networks with random initialization, and train our models on an AMD Ryzen 7 7840H CPU. All experiments are repeated for 5 times.

5.4 Results

We compare ReSCOM with the baseline across multiple cooperative tasks, demonstrating both their performance and communication overhead. Note that our concern is on the performance of the primary task. Therefore, the y-axis 'rewards' in all experiments represent the optimization objective defined in Eq. (1).

5.4.1 Drone Coverage. Fig. 5(a) and Fig. 5(b) present the learning curve over 150,000 episodes in terms of reward in the drone coverage environment with four agents and ten agents, respectively. Firstly, in the scenario with four agents, three types of ReSCOM surpass baselines and achieve comparable performance upon convergence. A notable performance jitter is observed in ReSCOM (delay) during curriculum transitions, while the learning process of ReSCOM with the continuous curriculum scheduler remains more stable. This jitter occurs because the gate employs an almost random exploration strategy when the second learning phase starts. Consequently, some valuable messages are overlooked, leading to a temporary decline in performance. However, after a period of adaptation, the gate rapidly learns a feasible communication strategy, and ReSCOM (delay) resumes an upward reward trend, ultimately attaining satisfactory performance. Secondly, in the scenario with ten agents, ReSCOM (delay) achieves the best performance. ReSCOM (root-2) and ReSCOM (linear) struggle to handle this task successfully since it still requires simultaneous learning of how and when to communicate during the early stage of training, which impedes learning efficiency. In contrast, the design of the discrete curriculum scheduler enables ReSCOM (delay) to thoroughly decompose the L2C process, where it solely learns how to communicate in the first curriculum. This suggests that decomposing the task of L2C into multiple sub-tasks and progressively learning them is effective, particularly in more complex environments.

In Fig. 6, we decompose the performance of ReSCOM into two metrics, coverage rate and collision rate, to analyze the advantages of ReSCOM over other baselines. Firstly, in the scenario with four agents, take ReSCOM (linear) for example, it significantly improves coverage rate by 22.73%, 48.66%, and 36.81% compared to SchedNet, TarMAC, and GACML; besides, ReSCOM (linear) has lower collision rate than the baseline by 69.12% and 34.83% compared to SchedNet and TarMAC. Compared with GACML, ReSCOM (linear) compromises a slight performance drop in collision rate (approximately 5.83%), but it improves a much obvious coverage rate and exhibits a greater gain in reward. Secondly, in the scenario with ten agents, ReSCOM (root-2) and ReSCOM (linear) outperform the baselines in both coverage rate and collision rate, while ReSCOM (delay) compromises a little in collision rate but gets a large improvement ($1.32 \sim 1.39\times$) in coverage rate.

Table 1: Performance comparison on highway environment. The best results are highlighted in bold.

	ReSCOM (root-2)	ReSCOM (linear)	ReSCOM (delay)	SchedNet	TarMAC	GACML
Mean reward	$\mathbf{7.33_{\pm 0.56}}$	$7.21_{\pm 0.49}$	$7.05_{\pm 0.54}$	$5.88_{\pm 0.53}$	$6.39_{\pm 0.48}$	$6.07_{\pm 0.61}$
Collision rate (%)	$10.92_{\pm 2.07}$	$\mathbf{10.67_{\pm 2.41}}$	$11.35_{\pm 1.19}$	$12.25_{\pm 2.24}$	$10.95_{\pm 1.34}$	$13.41_{\pm 2.59}$
Arrive rate (%)	$\mathbf{52.04_{\pm 2.23}}$	$51.02_{\pm 2.56}$	$51.62_{\pm 3.26}$	$35.39_{\pm 1.87}$	$36.54_{\pm 1.49}$	$38.27_{\pm 2.54}$
Bandwidth overhead (%)	$72.56_{\pm 3.31}$	$69.54_{\pm 3.24}$	$82.91_{\pm 2.52}$	$\mathbf{50.00_{\pm 0.00}}$	$75.89_{\pm 2.77}$	$79.15_{\pm 2.94}$

Fig. 7 shows the communication overhead during the training process. We follow the definition in [41], where the communication overhead during period T is expressed as $\frac{\sum_{t=1}^{T}\sum_{i=1}^{n} g_t^i}{nT}$, which is the average number of agents that conduct communication within a single time step. We show the results in Fig. 7. Since SchedNet permits half of the agents to broadcast messages at each time step, its communication overhead remains constant at 0.5. In the scenario with four agents, all three ReSCOM variants exhibit higher communication overhead than the baseline, with ReSCOM (delay) incurring the highest communication overhead since it applies the full communication strategy in the initial stage. However, in the scenario with ten agents, ReSCOM (linear) demonstrates the lowest communication overhead. Remember that its performance consistently surpasses the baselines as mentioned above. This indicates that the effectiveness of ReSCOM is not derived from more frequent communication, but rather from its ability to learn a more efficient communication strategy.

5.4.2 Highway. We show the evaluation results on highway environment in Table 1, with the optimal values for each metric highlighted in bold. We see that the three ReSCOM variants significantly outperform the baselines overall. For instance, ReSCOM (root-2) improves mean rewards by 24.66%, 14.71%, and 20.76% compared to SchedNet, TarMAC, and GACML, respectively. When we further decompose the performance of ReSCOM (root-2) into collision rate and arrival rate, we find that it significantly increases the arrival rate ($1.36 \sim 1.47\times$) while maintaining an acceptable collision rate, leading to a substantial improvement in overall performance. In terms of communication overhead, SchedNet incurs the least overhead, while ReSCOM (delay) incurs the most. Besides, compared to TarMAC and GACML, ReSCOM utilizing a continuous curriculum scheduler demonstrates a reduction in communication overhead.

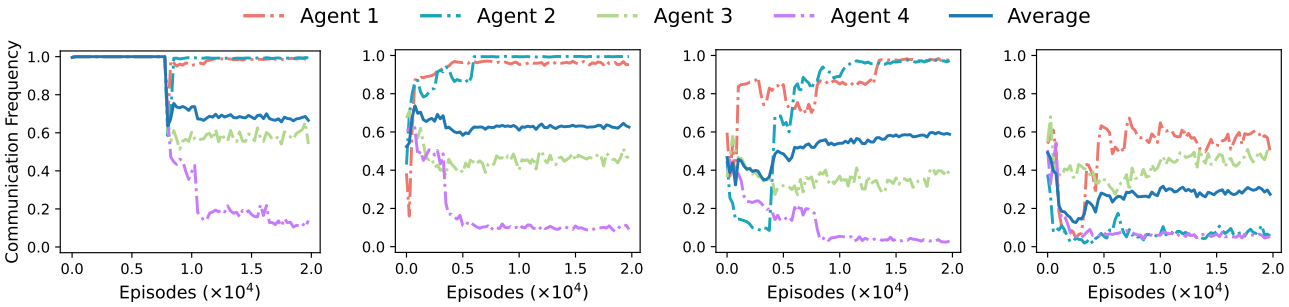
5.5 Why ReSCOM is effective?

To demonstrate the impact of the curriculum scheduler on the L2C process, we visualized the communication patterns of agents during training. In this experiment, we use the drone coverage environment with four agents as an example, with the observation ranges set as [0, 0, 2, 4]. We record the communication frequency of the agents in each episode during training. Note that due to the zero observation range of agents 1 and 2, they must communicate with other agents to perceive their surroundings and make decisions. Thus, upon convergence, these two agents are expected to initiate communication with nearly 100% probability at each time step. As illustrated in Fig. 8, when applying the curriculum scheduler, both agent 1 and agent 2 converge to the anticipated state. Besides, agent 4 communicates minimally due to its ability to observe the states of all landmarks and other agents in the environment, thus communication will not bring any gain in reward but introduce unnecessary overhead. Agent 3 communicates at a certain frequency, predominantly at the beginning of each episode. This is because, at the start of each episode, agents need to determine destination landmarks and avoid conflicts, requiring more communication to reach a consensus. In contrast, ReSCOM without the curriculum scheduler fails to reach the desired state, as shown in 8(d). Particularly, agent 2's communication frequency rapidly plummets to nearly 0% early in the training, indicating that it does not learn the correct communication strategy.

To further understand this phenomenon, we analyze the reward gain from communication. Reward gain from communication is defined as $r(s_t, a_{c,t}|_{g_t=\mathbb{I}(w_t>T_{th})}) - r(s_t, a_{n,t}|_{g_t=\{0\}_{j=1}^{n}})$, where $a_{c,t}$ and $a_{n,t}$ are the actions taken with and without communication, respectively. Here, we take ReSCOM (linear) as an example. As depicted in Fig. 9(a), the reward gain from communication increases as training proceeds during the early stages. This suggests that agents progressively learn to better utilize the additional environmental information provided by communication for decision-making. Besides, the cost of communication increases gradually over time (see the orange line in Fig. 9(a)). Initially, the benefits of communication outweigh its costs, encouraging agents to engage in communication more frequently and facilitating rapid learning of how to communicate. As communication costs rise, agents become more selective to minimize unnecessary communication overhead, shifting their focus to learning when to communicate. This also explains why, in Fig. 8(b), the average communication frequency of agents initially increases and then decreases during the training process. The same logic also applies to ReSCOM (root-2) and ReSCOM (delay). In contrast, without the curriculum scheduler, as shown in Fig. 9(b), the cost of communication greatly outweighs its benefits in the early stage of training, namely $\mathbb{E}_{s_t,a_{c,t},a_{n,t}}[r(s_t, a_{c,t}|_{g_t=\mathbb{I}(w_t>T_{th})}) - r(s_t, a_{n,t}|_{g_t=\{0\}_{j=1}^{n}})] < c$, leading agents to be reluctant to communicate. This phenomenon can be observed in Fig. 8(d). In the early stages of training, the average communication frequency of the agents rapidly declines to less than 20%. Since agents are reluctant to communicate, they can not effectively perceive the environment to optimize their policy and the L2C components that learn how to communicate. Consequently, the reward gain from communication remains low, which in turn discouraging the agents from engaging in communication, and ultimately leading to low learning efficiency.

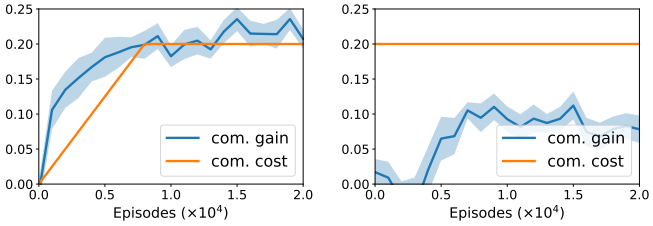
5.6 Ablation study

Firstly, we investigate the impact of the curriculum scheduler on ReSCOM's overall performance. Table 2 presents the performance of ReSCOM with and without the curriculum scheduler across three



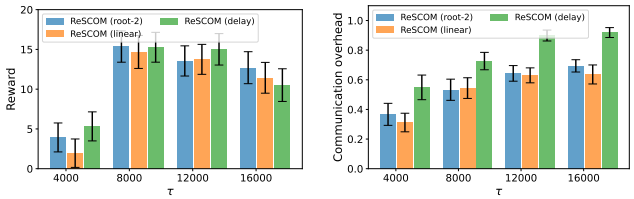
(a) ReSCOM (delay) (b) ReSCOM (linear) (c) ReSCOM (root-2) (d) ReSCOM w/o. curriculum scheduler

Figure 8: Visualizing the impact of the curriculum scheduler in the communication patterns of agents.



(a) ReSCOM (linear) (b) ReSCOM w/o. curriculum scheduler

Figure 9: Illustration of communication gain



(a) Overall performance (b) Communication overhead

Figure 10: Ablation study results for different values of hyper-parameter τ on drone coverage with four agents

Table 2: Ablation study results on various environment.

Method	Drone (4 agents)	Drone (10 agents)	Highway
ReSCOM	$14.68_{\pm 2.07}$	$-13.03_{\pm 4.01}$	$7.33_{\pm 0.56}$
Baseline	$-6.92_{\pm 2.58}$	$-32.65_{\pm 5.88}$	$6.01_{\pm 0.58}$

experimental scenarios. Here, we select the best-performing ones from three different curriculum schedulers for comparison, and the ReSCOM without curriculum scheduler serves as the baseline. The experimental results demonstrate that the introduction of the curriculum scheduler substantially enhances the performance, underscoring the effectiveness of the proposed approach.

Secondly, we examine the performance of ReSCOM across different values of the hyper-parameter τ using the drone environment with four agents as an example (see Fig. 10). Similar results are observed in other scenarios. As shown in Fig. 10(a), as τ increases, the mean reward first increases and then decreases, reaching its peak at $\tau = 8000$. This is reasonable because a larger τ implies a slower escalation of communication penalties, encouraging agents to communicate more frequently in the early stages of training, thereby allowing them more time to focus on learning how to communicate. However, an excessively large τ can lead to slower convergence, negatively impacting performance. Additionally, Fig. 10(b) shows that the communication overhead increases with τ. Essentially, τ regulates the pace at which curriculum difficulty intensifies. An optimal τ is expected to achieve high final performance with a rapid learning rate, while also reducing communication overhead during training. Based on the above observations, we set $\tau = 8000$ in our experiments.

6 CONCLUSION AND DISCUSSION

In this paper, we introduce ReSCOM, an efficient joint learning framework that integrates L2C with MARL. ReSCOM progressively adjusts the learning emphasis through reward-shaped curriculums, allowing agents to shift their focus from primary tasks and basic communication tasks (e.g., how to encode) to advanced communication strategies (e.g., determining when it is worthwhile to communicate). We design effective reward-shaped curriculums for L2C, including one discrete and two continuous curricula. Experimental results demonstrate that ReSCOM outperforms existing methods across multiple cooperative tasks, showing a performance improvement by 16.3%-21.9%.

Discussion Note that the effectiveness of ReSCOM relies on the setting of the curriculum scheduling functions, particularly the hyperparameter τ. The optimal τ setting often varies across different environments and usually requires expert knowledge about the task environment and its complexity. Achieving adaptive curriculum scheduling in curriculum learning remains a key challenge in this field. We present this as an open issue and look forward to addressing it in future work.

ACKNOWLEDGMENTS

This work has been partly funded by Horizon Europe CODECO project (Grant No. 101092696) and DAAD PPP (Grant No. 57702286).

REFERENCES

[1] Christopher Amato, Girish Chowdhary, Alborz Geramifard, N. Kemal Üre, and Mykel J. Kochenderfer. 2013. Decentralized control of partially observable Markov decision processes. In *52nd IEEE Conference on Decision and Control*. 2398–2405.

[2] Mihai Anca, Jonathan D Thomas, Dabal Pedamonti, Mark Hansen, and Matthew Studley. 2023. Achieving goals using reward shaping and curriculum learning. In *Proceedings of the Future Technologies Conference*. Springer, 316–331.

[3] Jiayu Chen, Yuanxin Zhang, Yuanfan Xu, Huimin Ma, Huazhong Yang, Jiaming Song, Yu Wang, and Yi Wu. 2021. Variational automatic curriculum learning for sparse-reward cooperative multi-agent problems. *Advances in Neural Information Processing Systems* 34 (2021), 9681–9693.

[4] Hwei-Ming Chung, Sabita Maharjan, Yan Zhang, and Frank Eliassen. 2020. Distributed deep reinforcement learning for intelligent load scheduling in residential smart grid. *IEEE Transactions on Industrial Informatics* (2020).

[5] Allan Dafoe, Yoram Bachrach, et al. 2021. Cooperative AI: machines must learn to find common ground. *Nature* 593, 7857 (2021), 33–36.

[6] Allan Dafoe, Edward Hughes, et al. 2020. Open problems in cooperative AI. In *NeurIPS*.

[7] 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.

[8] 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.

[9] 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).

[10] Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. 2017. Automated curriculum learning for neural networks. In *international conference on machine learning*. Pmlr, 1311–1320.

[11] Niko A Grupen, Daniel D Lee, and Bart Selman. 2021. Multi-agent curricula and emergent implicit signaling. *arXiv preprint arXiv:2106.11156* (2021).

[12] Guy Hacohen and Daphna Weinshall. 2019. On the power of curriculum learning in training deep networks. In *International conference on machine learning*. PMLR, 2535–2544.

[13] Timothy M. Hansen, Edwin K. P. Chong, Siddharth Suryanarayanan, Anthony A. Maciejewski, and Howard Jay Siegel. 2018. A Partially Observable Markov Decision Process Approach to Residential Home Energy Management. *IEEE Transactions on Smart Grid* 9, 2 (2018), 1271–1281.

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

[15] Daewoo Kim, Sangwoo Moon, David Hostallero, Wan Ju Kang, Taeyoung Lee, Kyunghwan Son, and Yung Yi. 2019. Learning to schedule communication in multi-agent reinforcement learning. *arXiv preprint arXiv:1902.01554* (2019).

[16] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. 2021. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems* (2021).

[17] Khaled B. Letaief, Yuanming Shi, Jianmin Lu, and Jianhua Lu. 2022. Edge Artificial Intelligence for 6G: Vision, Enabling Technologies, and Applications. *IEEE Journal on Selected Areas in Communications* 40, 1 (2022), 5–36.

[18] Toru Lin, Jacob Huh, Christopher Stauffer, Ser Nam Lim, and Phillip Isola. 2021. Learning to ground multi-agent communication with autoencoders. *Advances in Neural Information Processing Systems* 34 (2021), 15230–15242.

[19] Bo Liu, Qiang Liu, Peter Stone, Animesh Garg, Yuke Zhu, and Anima Anandkumar. 2021. Coach-player multi-agent reinforcement learning for dynamic team composition. In *International Conference on Machine Learning*. PMLR, 6860–6870.

[20] Qian Long, Zihan Zhou, Abhibav Gupta, Fei Fang, Yi Wu, and Xiaolong Wang. 2020. Evolutionary population curriculum for scaling multi-agent reinforcement learning. *arXiv preprint arXiv:2003.10423* (2020).

[21] 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).

[22] Hangyu Mao, Zhengchao Zhang, Zhen Xiao, Zhibo Gong, and Yan Ni. 2020. Learning agent communication under limited bandwidth by message pruning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 5142–5149.

[23] Siddharth Mysore, Robert Platt, and Kate Saenko. 2019. Reward-guided curriculum for robust reinforcement learning. In *Workshop on Multi-task and Lifelong Reinforcement Learning at ICML*.

[24] Parinaz Naghizadeh, Maria Gorlatova, Andrew S Lan, and Mung Chiang. 2019. Hurts to be too early: Benefits and drawbacks of communication in multi-agent learning. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 622–630.

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

[26] Murtaza Rangwala and Ryan Williams. 2020. Learning multi-agent communication through structured attentive reasoning. *Advances in Neural Information Processing Systems* 33 (2020), 10088–10098.

[27] Stefano Savazzi, Monica Nicoli, Mehdi Bennis, Sanaz Kianoush, and Luca Barbieri. 2021. Opportunities of Federated Learning in Connected, Cooperative, and Automated Industrial Systems. *IEEE Communications Magazine* 59, 2 (2021), 16–21.

[28] Jianzhun Shao, Zhiqiang Lou, Hongchang Zhang, Yuhang Jiang, Shuncheng He, and Xiangyang Ji. 2022. Self-organized group for cooperative multi-agent reinforcement learning. *Advances in Neural Information Processing Systems* 35 (2022), 5711–5723.

[29] Amanpreet Singh, Tushar Jain, and Sainbayar Sukhbaatar. 2018. Learning when to communicate at scale in multiagent cooperative and competitive tasks. *arXiv preprint arXiv:1812.09755* (2018).

[30] Sainbayar Sukhbaatar, arthur szlam, and Rob Fergus. 2016. Learning Multiagent Communication with Backpropagation. In *Advances in Neural Information Processing Systems (NIPS)*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (Eds.), Vol. 29. Curran Associates, Inc.

[31] Xianzhuo Sun and Jing Qiu. 2021. Two-Stage Volt/Var Control in Active Distribution Networks With Multi-Agent Deep Reinforcement Learning Method. *IEEE Transactions on Smart Grid* 12, 4 (2021), 2903–2912.

[32] Rundong Wang, Xu He, Runsheng Yu, Wei Qiu, Bo An, and Zinovi Rabinovich. 2020. Learning efficient multi-agent communication: An information bottleneck approach. In *International Conference on Machine Learning*. PMLR, 9908–9918.

[33] Weixun Wang, Tianpei Yang, Yong Liu, Jianye Hao, Xiaotian Hao, Yujing Hu, Yingfeng Chen, Changjie Fan, and Yang Gao. 2020. From few to more: Large-scale dynamic multiagent curriculum learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 7293–7300.

[34] Xin Wang, Yudong Chen, and Wenwu Zhu. 2021. A survey on curriculum learning. *IEEE transactions on pattern analysis and machine intelligence* 44, 9 (2021), 4555–4576.

[35] Yiru Wang, Weihao Gan, Jie Yang, Wei Wu, and Junjie Yan. 2019. Dynamic curriculum learning for imbalanced data classification. In *Proceedings of the IEEE/CVF international conference on computer vision*. 5017–5026.

[36] Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime Carbonell. 2019. Characterizing and avoiding negative transfer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11293–11302.

[37] Zhengxu Xia, Yajie Zhou, Francis Y Yan, and Junchen Jiang. 2022. Genet: automatic curriculum generation for learning adaptation in networking. In *Proceedings of the ACM SIGCOMM 2022 Conference*. 397–413.

[38] Yujian Ye, Yi Tang, Huiyu Wang, Xiao-Ping Zhang, and Goran Strbac. 2021. A Scalable Privacy-Preserving Multi-Agent Deep Reinforcement Learning Approach for Large-Scale Peer-to-Peer Transactive Energy Trading. *IEEE Transactions on Smart Grid* 12, 6 (2021), 5185–5200.

[39] Tingting Yuan, Wilson da Rocha Neto, Christian Esteve Rothenberg, Katia Obraczka, Chadi Barakat, and Thierry Turletti. 2020. Dynamic Controller Assignment in Software Defined Internet of Vehicles through Multi-Agent Deep Reinforcement Learning. *IEEE Transactions on Network and Service Management* (2020).

[40] Tingting Yuan, Jie Yuan, Dongxiao Liu, and Xiaoming Fu. 2022. Rethinking the Role of Communication in Collective Intelligence: Benefits and Costs. In *Proceedings of ACM SIGCOMM 2022 Workshops*. ACM, ACM, Amsterdam, the Netherlands.

[41] 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.

[42] Wen Zhang, Lingfei Deng, Lei Zhang, and Dongrui Wu. 2022. A survey on negative transfer. *IEEE/CAA Journal of Automatica Sinica* 10, 2 (2022), 305–329.