electronics

Article

ACUTE: Attentional Communication Framework for Multi-Agent Reinforcement Learning in Partially Communicable Scenarios

Chengzhang Zhao [1,2], Jidong Zhao [2], Zhekai Du [2] and Ke Lu [1,2,*]

[1] Yangtze Delta Region Institute (Huzhou), University of Electronic Science and Technology of China, Huzhou 314099, China
[2] School of Computer Science and Engineering, University of Electronic and Technology of China, Chengdu 611731, China
* Correspondence: kel@uestc.edu.cn

Abstract: Multi-agent reinforcement learning (MARL) aims to study the behavior of multiple agents in a shared environment. Existing communication-based MARL methods seldom consider the case of communication interference. However, such situations are not rare in real-world inter-agent communication. The majority of previous MARL methods struggle to design effective communication techniques for better cooperation between agents without considering communication reliability or channel capacity constraints. In addition, these models are typically not ready to be extended to large-scale multi-agent systems. To address these issues, in this paper, we propose a method named the **A**ttentional **C**omm**U**nica**T**ion Fram**E**work (ACUTE), which enables efficient communication between agents in a dynamic environment and improves the effectiveness of decision-making by using the most useful information from other agents. Specifically, we introduce an attention mechanism for the feature extraction of information during communication which determines the importance of messages received by agents. We evaluate the performance of our approach under different channel capacity constraints. Experimental results show that our model can efficiently exploit messages transmitted in unreliable channels for higher returns when compared to existing methods and can be applied to large-scale multi-agent systems.

Keywords: multi-agent reinforcement learning; partial communication; attention mechanism; large-scale systems

Citation: Zhao, C.; Zhao, J.; Du, Z.; Lu, K. ACUTE: Attentional Communication Framework for Multi-Agent Reinforcement Learning in Partially Communicable Scenarios. *Electronics* **2022**, *11*, 4204. https://doi.org/10.3390/electronics11244204

Academic Editor: Fernando De la Prieta

Received: 18 November 2022
Accepted: 13 December 2022
Published: 16 December 2022

1. Introduction

Multi-agent reinforcement learning (MARL) algorithm leverages reinforcement learning (RL) techniques to simultaneously train multiple agents in an interactive environment, where each agent regards the other agents as part of the environment. MARL has been applied successfully in many fields [1]. However, cooperation between agents is difficult to learn due to the complexity and uncertainty of the learning process. The complexity of MARL lies in the fact that each agent makes decisions that affect the environment differently. The difficulty is that each agent cannot know what the other agents are doing. This uncertainty presents a significant challenge for cooperation between agents [2]. Moreover, with an increase in the agents' scale, it is increasingly difficult to manage the interactions between all agents, because the number of possible interactions grows exponentially.

There are generally two learning paradigms widely adopted in early MARL studies. One is independent learning (IL), which allows each agent to learn its own strategy independently and has achieved good performance in some cooperative tasks. However, it ignores the connection between agents and aggravates non-stationary learning [3,4], leading to poor performance in some adversarial tasks [5]. Well-known methods in this category include IQL (independent Q-learning) [6] and IPPO (independent PPO) [7]. The

second learning paradigm is centralized learning, which considers the multi-agent system as a single system and solves the non-stationary issue of the environment. However, it implicitly assumes global communication and cannot be applied in scenarios with local communication and large-scale state-action space.

The current mainstream MARL algorithms typically solve the above issues by making a trade-off between fully independent and centered paradigms, which can be roughly categorized into collaboration-based MARL learning and communication-based learning. Specifically, the former usually belongs to the centralized training decentralized execution [8] (CTDE) paradigm. CTDE strategies combine the ideas of multi-agent learning and reinforcement learning, where agents have access to the global state during training and rely only on their partial observations during execution. CTDE strategies have drawn significant attention from researchers and been regarded as a landmark in MARL studies. Some representative CTDE methods such as QMIX [9], MADDPG [10], and MAPPO [11] have achieved considerable success in many MARL scenarios. The CTDE strategy can partially alleviate the environmental non-stationarity problem through centralized training. However, collaboration between the agents is still tricky because each agent can only obtain its local observation in the execution stage [12]. To solve this problem, many researchers have proposed communication-based multi-agent reinforcement learning methods to enable information sharing between agents during execution.

Although some communication-based approaches such as CommNet [13] and ATOC [14] explicitly leverage information interaction in the execution stage and achieve empirical success in the laboratory environment, they pay little attention to the unreliability of the channel and bandwidth limitation factor in reality. However, in many MARL applications, especially UAV swarms (unmanned aerial vehicle), these issues have to be considered. UAV swarms are widely used in modern combat [15] and are known for their large-scale number, low individual production cost, and resistance to communication interference. Inspired by this, we aim to study a multi-agent communication paradigm under the above constraints that can help UAV swarms efficiently accomplish global awareness and decision-making by their limited communication capabilities under interference.

In this paper, we build a model for the communication between agents with unreliable channels and bandwidth constraints and develop a communication mechanism between agents in this scenario, termed **A**ttentional **C**omm**U**nica**T**ion Fram**E**work (ACUTE). Specifically, the agents send messages to other agents by broadcasting, and the communication message is in the form of discrete values. Only a random fraction of the agents can receive this message, and the information that each agent can receive in each round of communication is limited by the maximum capacity of a fixed channel. In addition, we assume a dynamical environment where the number of agents varies over time. To handle this challenging setting, we design a generalizable message feature encoder module, which receives broadcast messages from a variable number of other agents and outputs a fixed-size encoded message. An attention mechanism is introduced to excavate the most important information. As a result, the encoder can extract features from the messages randomly received by each agent and efficiently leverage the useful information for decision-making, which is formulated as a part of the input of the action selector. Meanwhile, each agent broadcasts the encoding obtained by a local observation module to other agents that are able to receive the message. The features from the message feature encoder module and the local observation encoder module constitute the complete input of the action selector, which can be learned by the deep Q-learning or an actor-critic-based approach. Our approach is trained and tested in some scenarios in ma-gym [16] and LBF [17,18]. We also tested the model's performance in the above scenarios in the case of large-scale multi-agents and the effect of channel capacity size on the cooperative performance of the agents.

In summary, the contributions of our work can be summarized as follows:

1. To the best of our knowledge, we are among the first attempts to model partial communication scenarios for MARL under unreliable channels and bandwidth constraints,

which is easy to measure the level of communication interference and apply to various multi-agent environments without complex modifications to the environment.
2. To address these issues, we propose an attentional communication framework which efficiently extracts useful features from the randomly received messages for decision-making with unstable channels.
3. Extensive experimental results in various environments show that our approach achieves better performance than existing MARL algorithms and can be extended to large-scale multi-agent systems.

The remainder of this paper is organized as follows: In Section 2, we introduce the related work about communication-based MARL algorithms. In Section 3, our approach, i.e., ACUTE, is described in detail. In Section 4, we compare the performance of ACUTE and other algorithms under various settings and make a conclusion in Section 5.

2. Related Work

The study of communication-based MARL algorithms is an active area in MARL. Communication channels can be divided into two types: discrete-based and continuous-based channels. The discrete channel model treats a message as an action generated by the policy network. As a result, the action space is expanded, and a few actions are specifically designed for communication. In contrast, in a continuous channel, the message generated by an agent is used directly as an input to another agent's network, rather than simply generating an action of communication, so that the gradients flow can go through agents via the communication channel [14].

The first communication mechanism introduced in MADRL (multi-agent deep reinforcement learning) is DIAL [19], which integrates the learning of communication and policy in deep Q-networks and enables gradients to flow across agents in continuous communication channel, increasing the agent's perception of the environment, which alleviates the problem of a non-stationary environment better than IQL. However, the communication model of DIAL is too simple and can only select predefined messages (usually a real value) to be transmitted. Its performance is limited by the small amount of information to be transmitted, which makes it only capable of solving a few simple tasks. The bandwidth for communication is significantly wasted compared to the cost of establishing the connection in application.

CommNet is the first communication model for transmitting information based on a continuum channel where information is delivered by broadcasting following the CTCE framework, and the algorithm receives local observations of all agents as input and then outputs the decisions of all agents. However, it is poorly scalable because it is only a massive single feed-forward network for all agents. It performs poorly in environments with many agents because the number of interactions between agents grows exponentially as the number of agents increases. Moreover, the fully connected communication topology is not applicable in many application scenarios, such as unreliable communication channels.

The attention mechanism is a way to focus on a few key pieces of information by filtering them from massive information. Many successful applications have been developed in computer vision [20,21], natural language processing [22], and reinforcement learning [23].

MAAC [24] shares an attention module among centrally computed critics and uses attention to select relevant information to estimate critics, which is scalable and more effective in complex multi-intelligent systems. Nevertheless, centrally trained critic makes it difficult to expand to larger multi-agent systems.

ATOC expects the agent to learn the communication model, i.e., the agent itself decides at any moment whether it needs to communicate with other agents and with which agents. It introduces an attention mechanism to build its communication model where the agent uses its local observations to decide whether it needs to initiate communication with other agents within its field of view and with which agents. The topology of this communication method for communicating with neighbors is a tree topology [25], and this neighboring structure effectively reduces the communication cost and makes it easier to

be applied to large-scale MARL systems. However, ATOC only simulates the situation of limited communication bandwidth by a simple policy to select up to m agents to join the communication group, and it only considers the case of channel reliability. In addition, being a CTDE framework algorithm also makes it difficult to extend to scenarios where the number of agents is significant.

Mean field [26] receives the mean action of neighboring agents to help make the decision, and it is suitable for making algorithms scale to many agents. However, some important information that contributes to cooperative decision-making is lost in the averaging procedure. Similar to DIAL, it also has the problem of wasting communication bandwidth because only the action information of the agent is delivered in each timestep.

POOL [27] solves the large-scale multi-agent coordination problem by introducing a pheromone communication framework in reinforcement learning and using the pheromone mechanism in the ant colony algorithm to transfer information between agents. However, it is difficult for real-world agents to leave information in the environment. Therefore, the application of POOL is restricted.

Generally, existing communication-based approaches focus on designing efficient communication protocols to achieve SOTA performance in ideal experimental environments. However, they ignore the unreliable channel and bandwidth limitations in real-world scenarios and the wastage of channel resources caused by transmitting only little information per communication. In addition, the centralized training fashion makes these methods difficult to be trained on large-scale multi-agent systems. Our work is the first attempt towards a partially communicable MARL scenario, and the level of interference of the communication is easily measurable.

3. Background

3.1. Decentralized-Partially Observable Markov Decision Process (DEC-POMDP)

The problem that our framework tries to solve can be formulated as a DEC-POMDP [8] game, which is a partially observable multi-agent extension of the Markov decision process. It can be defined by a tuple $\mathcal{I}, \mathcal{S}, \{\mathcal{A}_i\}, \mathcal{P}, \{\mathcal{R}_i\}, \{\Omega_i\}, \mathcal{O}, \mathcal{T}, \gamma >$, where

- $\mathcal{I}$ is a finite set of N agents;
- $\mathcal{S}$ is a set of states;
- $\mathcal{A}_i$ is a set of available actions for agent i;
- Ω_i is a set of observations for agent i;
- T is the time horizon for the game;
- $\mathcal{R}_i$ is the reward function for agent i, and γ is the discount factor for individual rewards;
- $\mathcal{P}(s' \mid s, a)$ denotes the probability that agents took actions a in state s transitioning to state s';
- $\mathcal{O}(o \mid s, a)$ denotes the probability of agents obtaining the observations o.

3.2. Deep Q-Networks (DQN)

Q-learning [28] is one of the classic value-based methods in reinforcement learning, and DQN extends it with neural networks, which have promoted the development of RL and achieved success in many areas such as Atari Games [29] and Go [30]. Specifically, for an agent i at each timestep t, it receives the observation $\omega_t \in \Omega$, chooses an action $a_t \in \mathcal{A}$ according to the policy π, obtains a reward r_t, and transitions to the next state s_{t+1}. The objective is to maximize the total expected discounted reward $R_i = \sum_{t=0}^{T} \gamma^t r_t$. DQN uses neural networks to approximate the action-value function $Q^\pi(s, a) = \mathbb{E}_s[R_t \mid s_t = s, a_t = a]$, which can be recursively rewritten as

$$Q_i^\pi(\omega, a) = \mathbb{E}_{\omega'}\left[r(\omega, a) + \gamma \mathbb{E}_{a' \sim \pi}\left[Q^\pi\left(\omega', a'\right)\right]\right]. \quad (1)$$

The optimal value can be obtained by minimizing the loss:

$$\mathcal{L}_i(\theta) = \mathbb{E}_{\omega,a,r,\omega'}\left[\left(Q_i(\omega,a;\theta) - y\right)^2\right], \tag{2}$$

where $y = r + \gamma \max_{a'} Q_i(\omega', a'; \theta)$. DQN uses a replay buffer to store agent information in each step and randomly extracts it from it to optimize the Q-network. To reduce overestimation, double-DQN [31] decomposes the max operation in the target into the action Q-network and action Q-network.

4. Proposed Method

In this section, we introduce ACUTE in detail. We develop a concrete implementation based on a value-based approach. However, it can also be combined with other MARL algorithms. Generally, our method consists of three parts: an observation encoder, a message feature encoder, and an action selector, which are depicted in the top-left, top-right, and bottom-right parts of Figure 1, respectively. All agents share the same set of parameters.

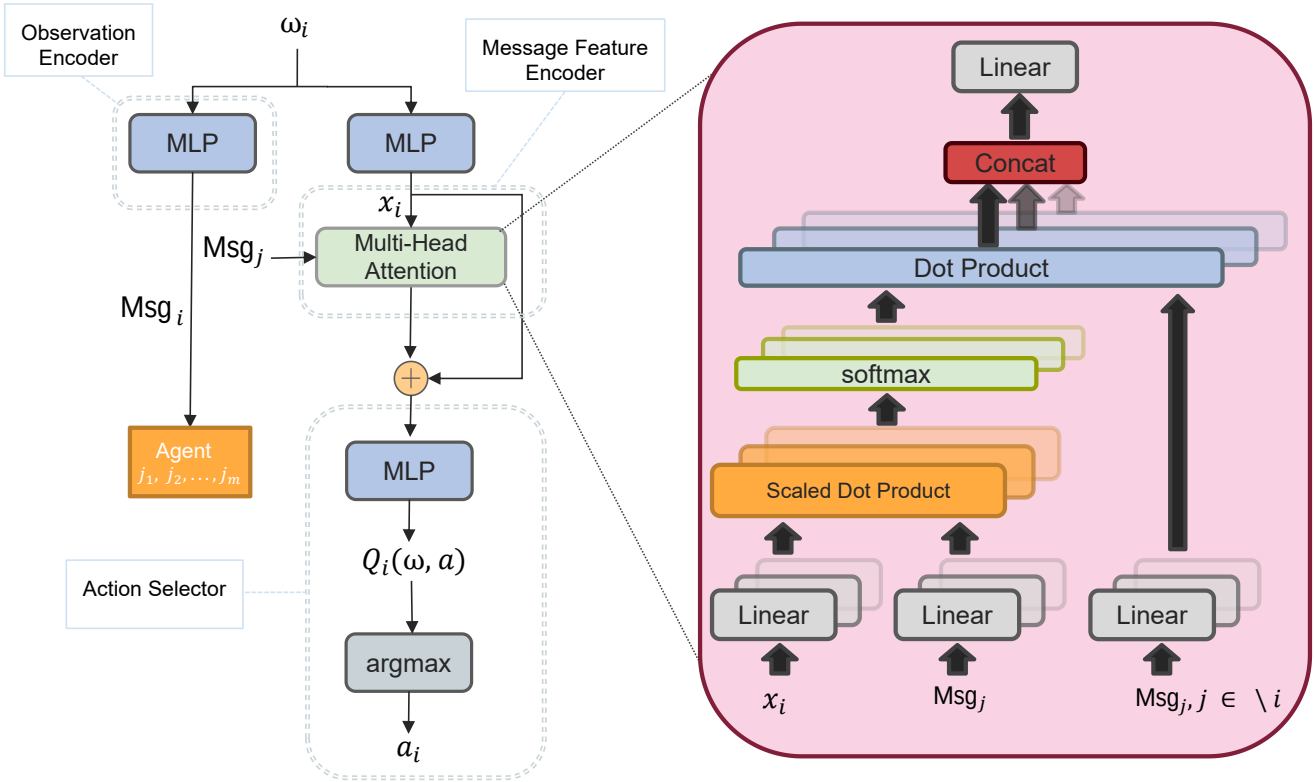


Figure 1. Network architecture for agent i. The observation encoder on the left encodes observation ω_i to Msg_i and broadcasts it. The right part shows how policy network receives the messages from other agents and aggregates them with ω_i by the multi-head attention unit.

Based on the IL framework, ACUTE does not require a global state s_t during the training and execution phases.

4.1. Communication Framework

Considering unreliable channels and bandwidth limitations, we design a passive communication model for this scenario. Specifically, the agent aims to receive as many messages from other agents as possible to establish a more global environmental awareness, thus achieving better synergy. In this way, the communication bandwidth can reach its maximum efficiency. For the unreliable channel condition, we choose a passive approach to

receive messages broadcast by other agents. Compared with peer-to-peer communication, the broadcast–receive asynchronous communication model is well suited to the unreliable channel conditions and can deliver information to as many agents as possible. For the communication bandwidth constraints, the agent is restricted to receiving at most C messages at each timestep. In our implementation, at each timestep t, we randomly pick the encoded local observations $y_j^t = \{y_{u_1}^t, y_{u_2}^t, \ldots, y_{u_3}^t\}(u_k \neq i, k \in [1..C])$ of C agents for each agent i as the input to the multi-head attention module.

Each agent i will receive a local observation ω_i^t at timestep t. The observation encoder takes the agent's local observations ω_i^t and self attribute $attr(agent_i)$ (including the agent number, location, etc.) as input. It then passes through a linear layer l_1(the MLP in an observation encoder) to obtain a compressed encoded observation message Msg_i^t, which contains the agent's local observations and its attributes, denoted as $\mathrm{Msg}_i^t = l_1(\omega_i^t, attr(agent_i))$. Then, we use $\backslash i$ to denote the set of all agents except i and index it with j (timestep t is omitted below). Each agent i broadcasts its observation message Msg_i and randomly receives C (maximum channel capacity) encoded messages Msg_j from other agents. Afterwards, these messages are stored in their message memory $\mathbf{M}_i$, which is initialized to a matrix of C empty messages at the beginning of each timestep. All the messages stored in $\mathbf{M}_i$ will be concatenated to the multi-head attention network. Figure 1 shows the overall framework of ACUTE for each agent i.

In contrast to MeanField [26] and CommNet [13], which integrate the shared information of agents using the average action and arithmetic mean of their surrounding neighbors, our attention module can efficiently find the information that promotes the agents' cooperation and integrate it into the inputs of the agents.

4.2. Attention Module

After receiving messages from adjacent agents, we expect the agent to explore useful information from the messages adaptively. Intuitively, the agent will query the messages for information relevant to its observations. One naive strategy to distinguish useful information could be the adoption of some predefined rules, such as giving higher weights to messages sent by close agents. However, this could be too poorly scalable. In this paper, we encourage the agents to find which messages are relevant in a learnable way. To this end, we introduce a multi-head attention mechanism in the message feature encoder module and merge it into the estimation of the Q-value function. Attentional mechanisms can be viewed as differentiable key–value memory models [21,32]. The attention module can better find the relationship between its observations and the received disordered messages than linear and convolutional layers [33].

During our work, we tried various input–output modes for the attention module and chose the most effective one. Specifically, we used the self-attention mechanism, which integrates the Msg_j and Msg_i (ignore t) to obtain a message matrix with $C+1$ messages size as the input of multi-headed self-attention module. It then flattens the output as the input of the action selector. However, we empirically found that it is difficult to converge in our case. We conjecture that the attention module needs to downscale the input to be able to extract effective features of the aggregated messages. In addition, we employed a residual structure which may also have an important impact on the model performance.

In our implementation, the attention module is implemented by the MLP. Concretely, at each timestep t, the attention module receives two inputs: an observation $x_i = l_2(\omega_i)$ of the agent after a linear layer l_2 transformation and C messages Msg_j received randomly from other agents. The output z_i of the attention layer is a weighted sum of the messages that agent i received:

$$z_i = \sum_{j \in \mathbf{u}} \alpha_j v_j = \sum_{j \in \mathbf{u}} \alpha_j h(V y_j). \tag{3}$$

The attention weights α_j are generated by the query–key–value system, where x_i is the "query" and y_j is the "key" and "value". By comparing the similarity of x_i and

each y_j, passing these embeddings into softmax to obtain the weight of each message, all "query", "key", and "value" are transformed by linear layers W_q, W_k, and W_v, respectively. Moreover, they matched according to the dimensionality of these matrices. To perceive the different input features, we use multiple attention heads [34]. Each head has a separate set of parameters (W_q, W_k, W_v), and the outputs of all heads are concatenated together and size-transformed by a matrix V. It will be added to the output x_i of the linear layer l_2 as the input of the action selector. The residual connection avoids the gradient vanishing problem and improves the convergence speed of the network during the training.

4.3. Training and Implementation Details

All agents are trained together in ACUTE by minimizing a joint regression loss function due to shared parameters. Compared to DQN, ACUTE adds an attention module to the Q-network to handle messages received from other agents. In addition to partial observation of the current environment, our action selector receives messages that have been processed through the attention module. For each agent i, the messages received by it are noted as $msgs$, and we update the action-value function $Q(\omega, msgs, a; \theta)$ as:

$$\mathcal{L}_i(\theta) = \mathbb{E}_{\omega,\mathrm{msgs},a,r,\omega',\mathrm{msgs}'}\left[\left(Q_i(\omega_i, \mathrm{msgs}_i, a; \theta) - y\right)^2\right], \tag{4}$$

where $y = r + \gamma \max_{a'} \bar{Q}_i(\omega', \mathrm{msgs}'_i, a'; \theta')$. The complete algorithm is presented in Algorithm 1. Specifically, the algorithm consists of two parts: in the first part, each agent encodes and broadcasts the local observations to other ones. In the second part, each agent receives the messages from other ones and selects actions based on its local observations as well as communication messages it received.

Algorithm 1 Deep Q-learning with ACUTE extension

Input: $\mathcal{D}$:replay buffer; θ, θ': current Q-network and target Q-network; N_e: number of epochs; T:maximum timestep of one game episode; M: memory to store messages with maximum C; N_{cap}: capacity for replay buffer
Output: Optimal parameters θ for the trained model
Initialize replay buffer $\mathcal{D}$ to capacity N_{cap}
Initialize messages memory for each agent i with all zero matrix of shape $C * size(\text{message})$,
Initialize θ with random parameters, initialize $\theta' = \theta$
for epoch e = 1 to N_e **do**
  for step t = 1 to T **do**
    obtain observation ω_i and transform ω_i into y_i for each agent i
    broadcast y_i to all available agents for each agent i
    randomly receive C messages and store in m_i for each agent i
    Select Action a_i for each agent by $a_i = \arg\max_a Q(\omega_i, \mathrm{msgs}_i, a; \theta)$
    Execute actions taken by agents and observe reward r_i and next observation ω'_i for each agent i
    Store transition $(\omega_i, a_i, r_i, m_i, \omega'_i, m'_i)$ to $\mathcal{D}$ for each agent i
  end for
  Sample random mini-batch of transitions from $\mathcal{D}$
  Perform a gradient descent step on loss according to Equation (2) to update θ
  Copy parameters from θ to θ' per certain steps
end for

4.4. Limitations

However, some limitations should be noted. When the interference level of communication channel changes dynamically, ACUTE can only utilize a less efficient bandwidth as it receives a determined number of messages at each timestep. For example, in t_1 the agent receives eight messages and in t_2 it only receives three messages; our model can only set up at the lower one, which means that five messages in t_1 need to be discarded. Moreover,

if the quality of the communication channel is good and the number of agents is extremely large, then many messages are redundant because the agent simply expects to learn about the information that it is interested in. We leave it to be solved in our future work.

5. Experiment

In this section, we separately set up experiments with two different task types of environments: cooperation confrontation and purely cooperation. We test the specific performance of ACUTE and the comparison with other methods. Besides we also show the effect of communication channel capacity constraints on ACUTE capability.

5.1. Hyperparameters, Base Settings and Detailed Information of Environments

We use three-layer MLP to implement the double Q-Network in all the experiments with Adam [35] optimizer and the hidden layer (second layer) has 64 units. It is also possible to implement it with RNN [36], but there is no significant difference in performance in our experiments. The implementation of two MARL algorithms, DQN and QMIX, and the multi-process benchmark framework are referred to EPyMARL [37]. For MFQ, our implementation refers to the open-source code provided by the authors. The deep learning frameworks we use are Pytorch-1.12.0 and CUDA-11.6. The learning rate of the optimizer lr is 5×10^{-4}. The discount factor of reward γ is 0.99. The capacity of the replay buffer is 5000, and the batch size is 32. To accelerate the sampling, we employ eight processes at a time to interact with the environment to obtain replay data. The agent chooses actions according to the epsilon-greedy policy, with an epsilon starting value of 1 and a final value of 0.05 for 2 or 5 million steps. The detailed information of environments used in experiment is shown in Table 1. For partially observable environments, the state size is the sum of the size of the observation space for all agents.

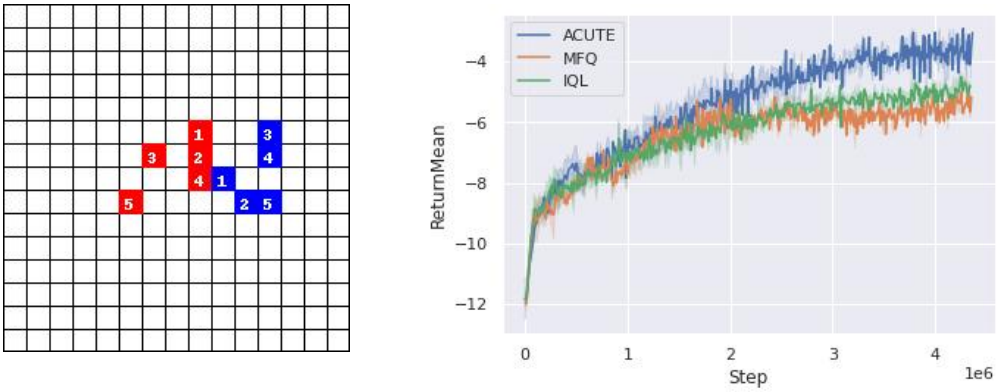
Table 1. Detailed Information of Environments in Experiment.

Environment	Observation Space	State Space	Agent Amount	Observation Type
Combat	120	600	5	part
8 × 8-2p-2f	12	12	2	full
5s-19 × 19-8p-5f	78	1482	19	part
Switch	2	4	2	part

5.2. Cooperation Confrontation Task

5.2.1. Description of Combat Environment

We chose the "Combat" environment in ma-gym as a cooperation confrontation scenario. Two opposing teams battle in a 15 × 15 grid as shown in Figure 2a, each team has five agents, and their initial position are sampled uniformly around the team center in a 5 × 5 square around. Each timestep, an agent can move in one of the four directions (up, down, left, and right) or attack the enemy within shooting range. After an attack, agents need a cool-down time during which they cannot attack. Our model controls one of the teams, and predefined rules control the other team. They attack the nearest enemy agent at their attack range, and the agents in the team share the observation range. This visual sharing setting provides an advantage to the team controlled by the rules. If a team of agents loses the game or reaches the time limit, they will receive a -1 reward. In addition, it receives a reward of -0.1 multiple, the sum of the enemy agents' health to encourage attacks on the enemy side. Each agent has a starting life value of 3 points, and it will die when the life value is less than zero. When one team's agents are killed within 40 timesteps, the other team will win. Otherwise, it is a tie.



(a) Combat environment (b) Result of combat environment

Figure 2. Sample and experiment results in ma-gym: Combat.

5.2.2. Result of Experiment in Combat Environment

We compared three value-based methods in combat environments: ACUTE, MeanField-Q (MFQ) [26], and IQL [6]. Figure 2b shows the results for each algorithm after 4 million training steps. It is evident that ACUTE receives higher average rewards (blue line) than the other two methods. As we can see, these three algorithms have similar performances in the beginning. However, ACUTE relies on information sharing to obtain better results in the end, which indicates that our framework can effectively exploit helpful information from disordered messages.

We evaluate ACUTE and other methods by running 100 test episodes. The mean test reward and win rate are illustrated in Table 2.

Table 2. Mean reward and win-rate per episode on ma-gym: Combat.

	ACUTE	**MFQ**	**IQL**
mean reward	−3.43	−5.12	−4.84
mean win rate	0.77	0.43	0.52

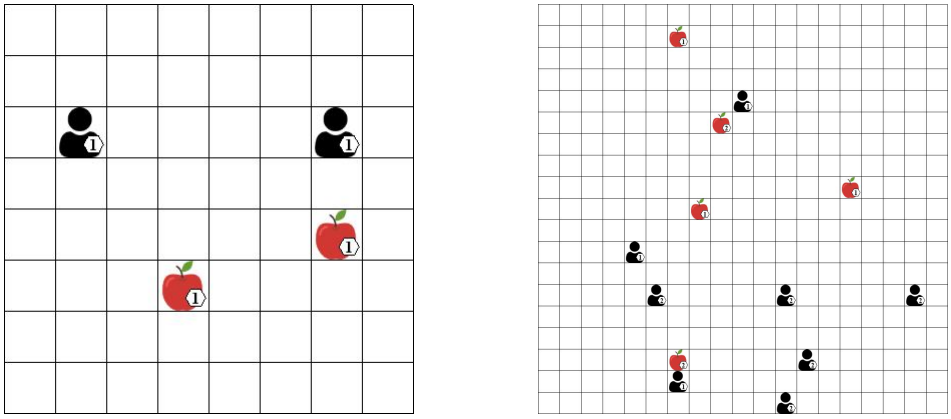
5.3. Pure Cooperation Task

5.3.1. Description of Level-Based Foraging Environment

Level-based foraging (LBF) environment is a mixed cooperative–competitive grid game that focuses on the agents' cooperation. Agents must navigate the environment and collect food items randomly scattered in a grid world. In the beginning, each agent is assigned a level, and each food item has its level. Agents can move in four directions and attempt to collect food items placed next to them. Only when the sum of the levels of the participating agents is equal to or greater than the level of the food item will the food collection be successful. Finally, agents are awarded points equal to the level of the food they helped collect divided by the sum of their levels.

LBF is a challenging environment, requiring the cooperation of multiple agents while being competitive at the same time, and the discount factor also necessitates speed for the maximization of rewards. Each agent is only awarded points if it participates in collecting food, so it has to balance collecting low-level food on its own or cooperating in acquiring higher rewards.

It is possible to customize some parameters of the environment to generate many different tasks in LBF. Figure 3 illustrates the two settings of the LBF environment. We define several distinct tasks with variable world size, number of agents, and number of food items to evaluate our approach and other baseline algorithms.

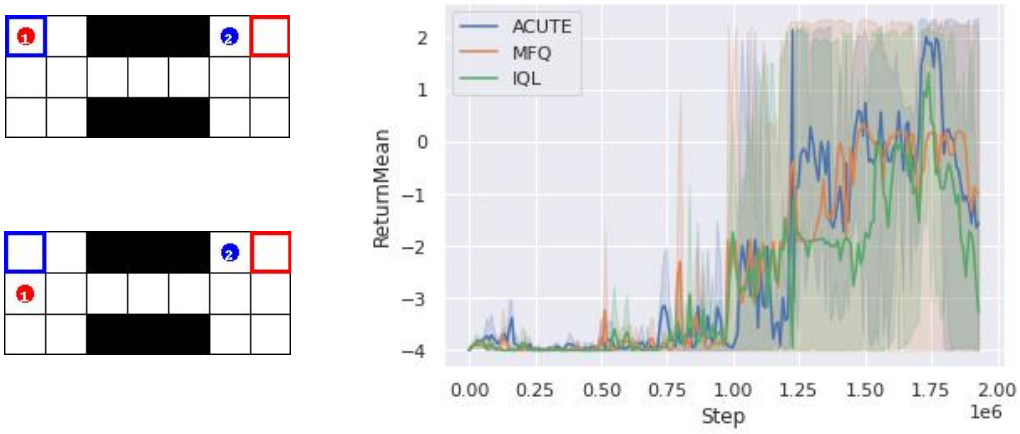


(a) 8x8-2p-2f (b) 19x19-8p-5f

Figure 3. Illustration of LBF environment: (**a**) 8 × 8 grid, 2 players, 2 food items (**b**) 19 × 19 grid, 8 players, 5 food items.

5.3.2. Description of Switch Environment

Switch environment is a grid world environment having two agents where each agent wants to move their corresponding home location (marked in boxes outlined in the same colors, as shown in Figure 4a). Agents must coordinate to pass through a narrow corridor that only one agent can traverse at a time. They need to cooperate and not block each others' passages. When the agents reach their home zone, they will be rewarded. The game ends when both agents have either reached their home state or have taken a maximum of 20 steps in the environment.



(a) Switch environment (b) Result of switch environment

Figure 4. Sample and experiment result in ma-gym: Switch2-v0.

5.3.3. Cooperation Task on Reliable Channel

In order to show the effectiveness of communication in our method, we consider the simplest case where there are only two agents in the 8x8 grid world, such as Figure 3a. Two agents can send and receive messages from each other over a reliable channel. We compare our model with the baseline IL algorithm DQN, MFQ with global information that averages all agents' actions, and the value-based baseline CTDE algorithm QMIX.

Figure 5 indicates that the attention module in ACUTE can efficiently extract more effective information from the local observation of the agent than the average action in MFQ, and the result is shown in Table 3. It performs even better than the value-based CTDE method QMIX. This result suggests that communication between agents during

execution is essential, and the more messages are exchanged, the more effective the resulting decisions are.

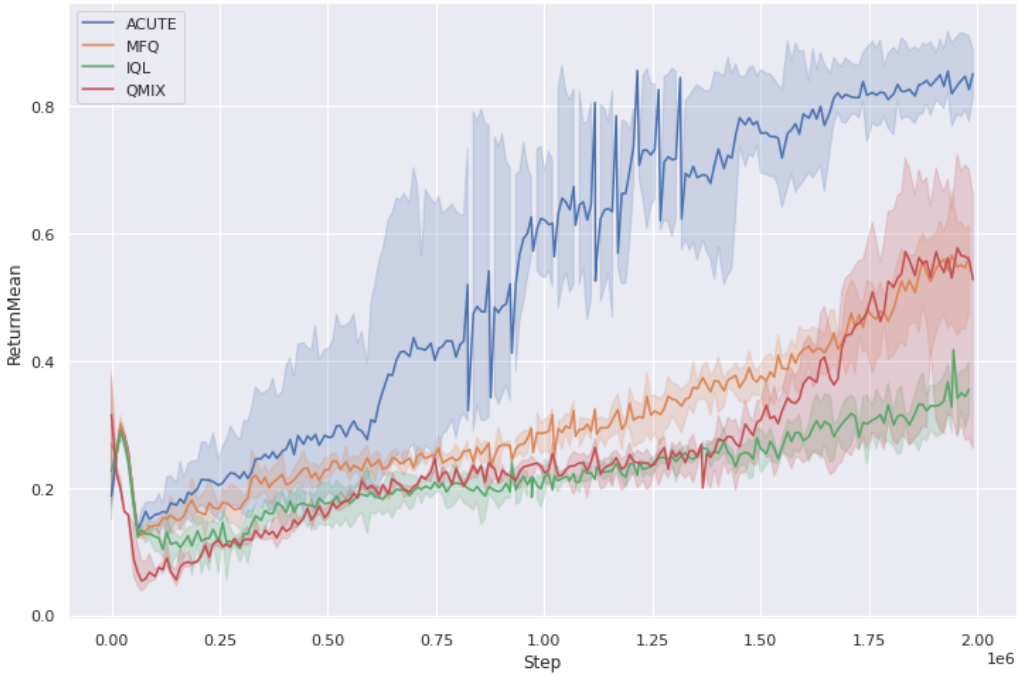


Figure 5. Experiment result in lbforaging: 8 × 8-2p-2f.

Table 3. Mean reward per episode on lbforaging: 8 × 8-2p-2f.

	ACUTE	MFQ	IQL	QMIX
mean reward	0.83	0.57	0.35	0.56

Experiments in the Switch environment also confirmed this speculation. From Figure 4b, it can be seen that ACUTE and MFQ can decide whether to give way by receiving a message from another agent. ACUTE analyzes the message directly, while MFQ decides its action based on the other side's action. IQL is limited by the non-smoothness of the algorithm, causing the model to converge slowly and the return value to fluctuate more. The experimental results can be found in Table 4.

Table 4. Mean reward per episode on ma-gym: Switch2-v0.

	ACUTE	MFQ	IQL
mean reward	2.31	2.23	2.05

5.3.4. Cooperation Task on Unreliable Channel

Section 5.3.3 describes the simplest case with two agents and no communication interference channel. We extend it to tasks with large-scale agents cooperating under different levels of interference to evaluate the impact of communication interference on ACUTE. We also tested the performance of other value-based methods for this task (without communication interference).

We select "5s-19 × 19-19p-9f" from LBF tasks. In this term, "5s" means that each agent can only see 5 × 5 squares centered on itself. In this 19 × 19 size environment, there are 19 agents and 9 food items. For ACUTE, we set different interference levels: "ACUTE-3" means that at each timestep, the agent will receive messages broadcast by any three of the other agents, "ACUTE-6" means that at each timestep the agent will receive messages broadcast by any six other agents, and so on. IQL does not require communication during

training or execution for other value-based methods, so whether or not it interferes does not affect its performance. MFQ collects the action of other agents to obtain the average action as part of the input and does not consider communication interference, so the experiment for MFQ is based on a reliable channel. Each algorithm was run for 2 million timesteps in the environment. Figure 6 shows the performance of ACUTE-3, ACUTE-6, ACUTE-9, MFQ, and IQL in this LBF task. Each set of experiments was run three times to better evaluate these methods' performance.

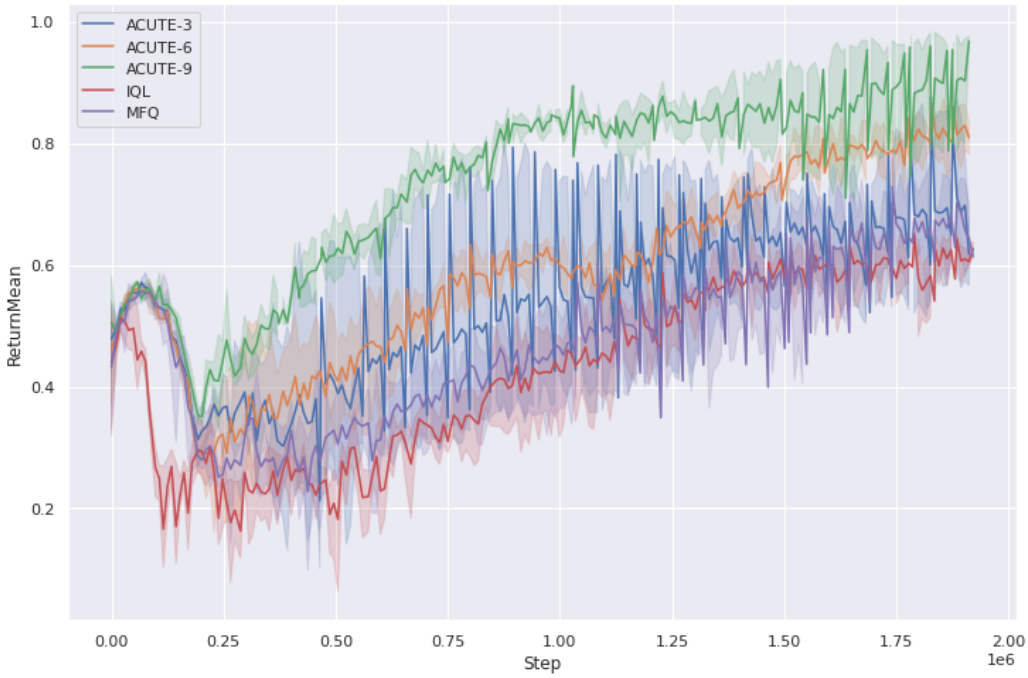


Figure 6. Experiment result in lbforaging: 5s-19 × 19-19p-9f.

Figure 6 shows that compared to the baseline IQL, which does not require communication during training and execution, communication between agents can improve decision-making. Intuitively, communication can extend an agent's perceptual range and assist its decisions based on those of other agents. MFQ approximates the effect of other agents on the current agent by computing the average of the previous actions of other agents. However, this estimate is rough, so the actual effect of MFQ is slightly better than the baseline method IQL. The attention module in ACUTE can extract information relevant to itself from the messages received by other agents. The experimental results on ACUTE-3, ACUTE-6, and ACUTE-9 suggest that, over a range, the more information received, the better the cooperation between agents. This is the reason why ACUTE-9 achieves the best performance in the experiment. Experimental results are given in Table 5.

Table 5. Mean reward per episode on lbforaging: 5s-19 × 19-19p-9f.

Method	Mean Reward
ACCUTE-3	0.67
ACCUTE-6	0.82
ACCUTE-9	0.96
IQL	0.61
MFQ	0.64

6. Conclusions and Future Work

In this paper, inspired by the real-world environment where the communication between agents often suffers from interference, we conceive a simplified unreliable channel

model and propose ACUTE, a framework for attentional communication between agents to address multi-agent cooperation tasks in partially communicable scenarios. It integrates reinforcement learning techniques and attention mechanisms that enable the agent to distinguish features that are relevant to itself from randomly received messages. We implement our framework based on Q-learning, but ACUTE can also be an extension of other MARL algorithms. We evaluated ACUTE using different Q-learning-based algorithms in several settings in ma-gym and LBF. The experimental results show that ACUTE can fully utilize the information delivered in unreliable channels to obtain higher returns. In addition, we tested the performance of ACUTE under different levels of interference and found that the performance of ACUTE is inversely correlated with the interference level. These results also confirm that the more information an agent receives, the better it can perceive the state in the global environment. Particularly, our work will help improve global environment awareness and decision-making effectiveness for individuals in large-scale UAV swarms under different levels of communication interference.

In our future work, we will research how ACUTE can be combined with other independent MARL algorithms such as TD3 [38] and SAC [39] and how to improve their performance in larger-scale multi-agent systems.

Author Contributions: Conceptualization, J.Z. and K.L.; methodology, C.Z.; software, C.Z.; validation, C.Z. and Z.D.; writing—Original draft preparation, C.Z.; writing—Review and editing, Z.D.; visualization, C.Z.; supervision, J.Z.; project administration, K.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China under Grant 62273071.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhang, K.; Yang, Z.; Başar, T. Multi-agent reinforcement learning: A selective overview of theories and algorithms. In *Handbook of Reinforcement Learning and Control*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 321–384.
2. Guicheng, S.; Yang, W. Review on Dec-POMDP Model for MARL Algorithms. In *Smart Communications, Intelligent Algorithms and Interactive Methods*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 29–35.
3. Tan, M. Multi-Agent Reinforcement Learning: Independent versus Cooperative Agents. In *Proceedings of the Tenth International Conference on International Conference on Machine Learning*; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 1993; pp. 330–337.
4. Foerster, J.; Farquhar, G.; Afouras, T.; Nardelli, N.; Whiteson, S. Counterfactual multi-agent policy gradients. In Proceedings of the AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2018; Volume 32.
5. He, H.; Boyd-Graber, J.; Kwok, K.; Daumé, H., III. Opponent modeling in deep reinforcement learning. In Proceedings of the International Conference on Machine Learning, PMLR, New York, NY, USA, 20–22 June 2016; pp. 1804–1813.
6. Gupta, J.K.; Egorov, M.; Kochenderfer, M. Cooperative multi-agent control using deep reinforcement learning. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 66–83.
7. De Witt, C.S.; Gupta, T.; Makoviichuk, D.; Makoviychuk, V.; Torr, P.H.; Sun, M.; Whiteson, S. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv* **2020**, arXiv:2011.09533.
8. Oliehoek, F.A.; Spaan, M.T.; Vlassis, N. Optimal and approximate Q-value functions for decentralized POMDPs. *J. Artif. Intell. Res.* **2008**, *32*, 289–353. [CrossRef]
9. Rashid, T.; Samvelyan, M.; Schroeder, C.; Farquhar, G.; Foerster, J.; Whiteson, S. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 4295–4304.
10. Lowe, R.; Wu, Y.I.; Tamar, A.; Harb, J.; Pieter Abbeel, O.; Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 6379–6390.
11. Yu, C.; Velu, A.; Vinitsky, E.; Wang, Y.; Bayen, A.; Wu, Y. The surprising effectiveness of ppo in cooperative, multi-agent games. *arXiv* **2021**, arXiv:2103.01955.
12. Hernandez-Leal, P.; Kartal, B.; Taylor, M.E. A survey and critique of multiagent deep reinforcement learning. *Auton. Agents-Multi-Agent Syst.* **2019**, *33*, 750–797 [CrossRef]
13. Sukhbaatar, S.; Fergus, R. Learning multiagent communication with backpropagation. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 2244–2252.

14. Jiang, J.; Lu, Z. Learning attentional communication for multi-agent cooperation. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 7254–7264.
15. Bridley, R.; Pastor, S. Military Drone Swarms and the Options to Combat Them. *Small Wars* **2022**. Available online: https://smallwarsjournal.com/jrnl/art/military-drone-swarms-and-options-combat-them (accessed on 7 December 2022).
16. Koul, A. Ma-Gym: Collection of Multi-Agent Environments Based on OpenAI Gym. 2019. Available online: https://github.com/koulanurag/ma-gym (accessed on 7 December 2022).
17. Christianos, F.; Schäfer, L.; Albrecht, S. Shared experience actor-critic for multi-agent reinforcement learning. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 10707–10717.
18. Papoudakis, G.; Christianos, F.; Schäfer, L.; Albrecht, S.V. Comparative evaluation of cooperative multi-agent deep reinforcement learning algorithms. *arXiv* **2020**, arXiv:2006.07869.
19. Foerster, J.; Assael, I.A.; De Freitas, N.; Whiteson, S. Learning to communicate with deep multi-agent reinforcement learning. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 2137–2145.
20. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
21. Mnih, V.; Heess, N.; Graves, A.; Kavukcuoglu, K. Recurrent models of visual attention. *Adv. Neural Inf. Process. Syst.* **2014**, *27*, 2204–2212.
22. Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv* **2014**, arXiv:1409.0473.
23. Fawzi, A.; Balog, M.; Huang, A.; Hubert, T.; Romera-Paredes, B.; Barekatain, M.; Novikov, A.; R Ruiz, F.J.; Schrittwieser, J.; Swirszcz, G.; et al. Discovering faster matrix multiplication algorithms with reinforcement learning. *Nature* **2022**, *610*, 47–53. [CrossRef] [PubMed]
24. Iqbal, S.; Sha, F. Actor-attention-critic for multi-agent reinforcement learning. In Proceedings of the International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 9–15 June 2019; pp. 2961–2970.
25. Sheng, J.; Wang, X.; Jin, B.; Yan, J.; Li, W.; Chang, T.H.; Wang, J.; Zha, H. Learning structured communication for multi-agent reinforcement learning. *arXiv* **2020**, arXiv:2002.04235.
26. Yang, Y.; Luo, R.; Li, M.; Zhou, M.; Zhang, W.; Wang, J. Mean field multi-agent reinforcement learning. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 5571–5580.
27. Cao, Z.; Shi, M.; Zhao, Z.; Ma, X. PooL: Pheromone-inspired Communication Framework forLarge Scale Multi-Agent Reinforcement Learning. *arXiv* **2022**, arXiv:2202.09722.
28. Watkins, C.J.; Dayan, P. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292. [CrossRef]
29. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef]
30. Silver, D.; Huang, A.; Maddison, C.J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. Mastering the game of Go with deep neural networks and tree search. *Nature* **2016**, *529*, 484–489. [CrossRef]
31. Van Hasselt, H.; Guez, A.; Silver, D. Deep reinforcement learning with double q-learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; Volume 30.
32. Oh, J.; Chockalingam, V.; Lee, H. Control of memory, active perception, and action in minecraft. In Proceedings of the International Conference on Machine Learning, PMLR, New York, NY, USA, 20–22 June 2016; pp. 2790–2799.
33. Ji, S.; Xie, Y.; Gao, H. *A Mathematical View of Attention Models in Deep Learning*; Texas A&M University: College Station, TX, USA, 2019. Available online: https://people.tamu.edu/sji/classes/attn.pdf (accessed on 16 August 2022).
34. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.
35. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
36. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078.
37. Papoudakis, G.; Christianos, F.; Schäfer, L.; Albrecht, S.V. Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS)*; PMLR: New York, NY, USA, 2021.
38. Fujimoto, S.; Hoof, H.; Meger, D. Addressing function approximation error in actor-critic methods. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 1587–1596.
39. Haarnoja, T.; Zhou, A.; Hartikainen, K.; Tucker, G.; Ha, S.; Tan, J.; Kumar, V.; Zhu, H.; Gupta, A.; Abbeel, P.; et al. Soft actor-critic algorithms and applications. *arXiv* **2018**, arXiv:1812.05905.