Pattern Recognition 145 (2024) 109917

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Robust multi-agent reinforcement learning via Bayesian distributional value estimation☆

Xinqi Du [a,b], Hechang Chen [a,b,*], Che Wang [a,b], Yongheng Xing [c], Jielong Yang [a,b], Philip S. Yu [d], Yi Chang [a,b], Lifang He [e]

[a] School of Artificial Intelligence, Jilin University, Changchun, China
[b] Engineering Research Center of Knowledge-Driven Human-Machine Intelligence, Ministry of Education, China
[c] College of Computer Science and Technology, Jilin University, Changchun, China
[d] Department of Computer Science, University of Illinois at Chicago, IL 60607, USA
[e] Department of Computer Science and Engineering, Lehigh University, PA 18015, USA

ARTICLE INFO

Keywords:
Multi-agent reinforcement learning
Bayesian inference
Distributional value function
Deep reinforcement learning

ABSTRACT

Reinforcement learning in multi-agent scenarios is essential for real-world applications as it can vividly depict agents' collaborative and competitive behaviors from a perspective closer to reality. However, most existing studies suffer from poor robustness, preventing multi-agent reinforcement learning from practical applications where robustness is the core indicator of system security and stability. In view of this, we propose a novel Bayesian Multi-Agent Reinforcement Learning method, named BMARL, which leverages the distributional value function calculated by Bayesian inference to improve the robustness of the model. Specifically, Bayesian linear regression is adopted to estimate a posterior distribution concerning value function parameters, rather than approximating an expectation value for Q-value by point estimation. In this way, the value function is more generalized than previously obtained by point estimation, which is beneficial to the robustness of our model. Meanwhile, we utilize the Gaussian prior knowledge to integrate more prior knowledge while estimating the value function, which improves learning efficiency. Extensive experimental results on three benchmark multi-agent environments comparing with seven state-of-the-art methods demonstrate the superiority of BMARL in terms of both robustness and efficiency.

1. Introduction

Reinforcement learning (RL) allows agents to maximize the cumulative reward by interacting with the environment. Recently, it has become the most promising method to master challenging domains, e.g., game playing [1], robotic control [2], finance [3], etc [4]. However, most existing studies are designed for single-agent scenarios, which are not suitable for real-world scenarios involving multiple cooperative or competitive agents. Taking traffic signal control as an example, it is an important but challenging real-world problem in our daily life [5], which objective is to safely direct drivers through the intersection. Assuming that each traffic signal light is modeled as an agent, their policies are cooperated with each other. The robustness of the model for traffic signal control is of great importance because it is directly related to the smoothness of traffic conditions and even the safety of human lives. However, most of the existing models are lacking in robustness and become incompetent when encountering unfamiliar environments. Therefore, enhancing the practicability of multi-agent reinforcement learning (MARL) by improving robustness have great potential in real applications.

In the literature, several studies have been proposed for multi-agent systems. For example, some of the earliest proposed methods tend to train agents individually using local observations [6,7]. It scarcely covers the complete view of a multi-agent environment and easily leads to the problem of instability. To address this problem, the subsequently proposed algorithms advocate integrating observations and actions of all agents as a joint input [8,9], but they incur the limitation of scalability [10]. Recently, a centralized training and decentralized execution (CTDE) framework is proposed [11], in which agents only

☆ This work is supported in part by the International Cooperation Project of Jilin Province (20220402009GH), the National Natural Science Foundation of China (61976102 and U19A2065), the National Key R&D Program of China (2021ZD0112501, 2021ZD0112502), Lehigh's Accelerator Foundation under grant No. S00010293, and the National Science Foundation (MRI 2215789, IIS 1909879).

* Corresponding author at: School of Artificial Intelligence, Jilin University, Changchun, China.
E-mail address: chenhc@jlu.edu.cn (H. Chen).

https://doi.org/10.1016/j.patcog.2023.109917
Received 17 May 2022; Received in revised form 3 November 2022; Accepted 23 August 2023
Available online 28 August 2023

incorporate joint information during training to broaden their perspectives. It demonstrates excellent performance in many domains by deploying value-based or policy-based reinforcement learning methods, e.g., QMIX [12] and MADDPG [11]. More recently, some emerging technologies, e.g., attention mechanism and graph neural network, are incorporated into MARL to enhance its performance [13,14].

Despite the success of existing multi-agent reinforcement learning methods achieved in many fields, they are still confronted with the challenge of poor robustness [15]. Robustness is the capability to be compatible with the data, which can be reflected laterally during training. Specifically, existing methods approximate the expectation of Q-value by point estimation, i.e., calculating a specific value as the parameter of the Q-value function according to samples. It is easily overfitting to the data generated during interacting with a specific environment. Especially, scenarios in the real world are always with high complexity, e.g., traffic signal control [5], which makes the problem more prominent. Models with poor robustness might lead to wrong decisions, which poses a severe threat to people's safety. Therefore, a more generalized value function is desirable to intensify the robustness of models. Meanwhile, more attention needs to be paid to method efficiency. Existing methods require considerable interactions when training the model to achieve a good performance [16], which is time-consuming, and even infeasible for tasks with high interaction overhead and against maximizing income. These deficiencies become the fatal bottleneck in applying MARL algorithms to complex real-world scenarios.

To this end, we propose a robust algorithm called Bayesian Multi-Agent Reinforcement Learning (BMARL) to approximate the distribution for parameters regarding the Q-value function through Bayesian inference. Specifically, we deploy the distribution value function in multi-agent systems and employ Bayesian linear regression to estimate a posterior distribution for the parameter, which provides comprehensive statistical information about the Q-value and captures the uncertainty in the environment. In this way, the distribution estimated by Bayesian linear regression is a continuous distribution, which is more comprehensive than a discrete distribution proposed in existing works and not heavily dependent on specific task information. With the support of the posterior distribution, we can obtain a more generalized value function and an accurate estimation for the Q-value, thus improving the robustness to stochastic environments. Furthermore, the Gaussian prior is incorporated into the value function parameters, enabling BMARL model to be partially independent of the interaction data so that the learning efficiency can be improved. Experimental results on three benchmark scenarios demonstrate the superiority of our proposed method comparing with the state-of-the-art methods. The main contributions of this paper are summarized as follows:

- A novel algorithm called BMARL is proposed to overcome the challenges of poor robustness and inefficiency. To the best of our knowledge, this is the first effort in the literature that uses Bayesian inference to maintain the posterior distribution regarding the value function of MARL.
- The proposed BMARL model employs the Bayesian linear regression to improve the robustness by refining the posterior distribution for value function parameters. Meanwhile, a Gaussian prior is leveraged to learn the value function, which is conducive to improving the model efficiency.
- The effectiveness of BMARL is validated by performing comprehensive experiments on three cooperative and competitive multi-agent environments comparing with seven state-of-the-art methods. Extensive experimental results demonstrate the superiority of BMARL in terms of both robustness and efficiency.

The remainder of this paper is organized as follows. Section 2 summarizes previous works related to ours. Section 3 presents the notations used in our method and simply reviews basic theorems. Section 4 elaborates the details of our model design. Section 5 conducts a series of experiments on benchmark scenarios to validate the effectiveness of BMARL and provides the result analysis in detail. Section 6 concludes this paper and discusses further work.

2. Related work

In this section, we briefly discuss the previous works related to our proposed method from two aspects, i.e., multi-agent reinforcement learning and distributional value function, which can distinguish and highlight the contributions of our methods.

2.1. Multi-agent reinforcement learning

In the literature, several studies have been proposed for multi-agent systems and apply it into games [11], resource management [17], and so forth [5]. The initial method to learn in multi-agent settings is to train agents in decentralized [6] modes. Unfortunately, it does not works well since the environment is partially observed and affected by other agents simultaneously, giving rise to instability in training. It is nature to combine the joint observations and actions of all agents and train in a full-centralized mode [8,9] to alleviate this problem. However, the dimensions of the joint value increase exponentially with the number of agents, and the scalability is severely limited. Hereafter, centralized training and decentralized execution (CTDE), a compromise, has emerged as the prevailing framework [18], and often combines with advanced deep reinforcement learning methods, e.g., value-based [19] or policy-based [11] methods, for multi-agent training.

Specifically, VDN [19] and QMIX [12] are typically value-based MARL, proficient in cooperative tasks. They decompose the team-reward into agents' rewards to promote cooperation and employ recurrent neural network to solve the instability caused by partial observations. In addition to the value-based methods, policy-based MARL approaches are appealing likewise. MADDPG [11] incorporates deep deterministic policy gradient [2] and trains agents with an ensemble of policies to relieve instability, good at cooperative and competitive tasks. Emerging technologies, e.g. attention mechanism and graph neural network are employed to enhance MARL further, for instance, MAAC [20] combines multi-head attention to filter irrelevant information that may interfere with policy training, but it poses a challenge of inefficiency due to its complex architecture. However, most prior works have studied the various axes of solutions for the challenges in isolation, and few copes with robustness and efficiency simultaneously.

2.2. Distributional value function

The challenges of poor robustness also exist in single-agent settings since the traditional reinforcement learning methods simply approximate the expectation of Q-value [1], suffering from the approximation errors [21,22], and thus leading to suboptimal policies and instability. To address this issue, methods of deploying more independent estimators are proposed to provide various perspective estimations. For example, Double DQN [23] uses two independent Q-network to alleviate the overestimation. Bootstrapped DQN [24] further deploys multiple Q-networks, similar to ensemble learning, to obtain more information about Q-value. Another way is to approximate the distribution over the value function to provide comprehensive information of Q-value, which is first proposed in [25]. Because of its comprehensive grasp of Q-value, it is conducive to improving the accuracy of estimation and policy generalization, which makes it an attractive approach. Specifically, the method proposed in [25] approximates the distribution in a discrete way. QR-DQN [26] is proposed to approximate the value distribution using quantile regression, which is more flexible than [25]. In contrast, some methods [27,28] directly learn the continuous distribution. Similarly, some studies combine distributional

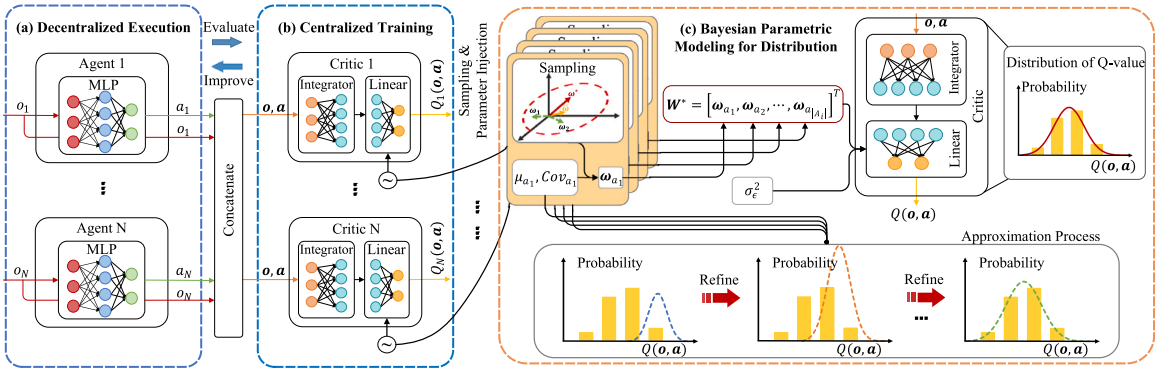


Fig. 1. The overall architecture of BMARL. (a) Decentralized execution: each agent merely uses private observation to make action decisions. (b) Centralized training: joint information is incorporated into critic networks, which can ease the problem of non-stationary in multi-agent settings. (c) Bayesian parametric modeling for distribution: it is employed to calculate the Q-function posterior distribution, and then BMARL samples its interest $\boldsymbol{\omega}_a$ according to the posterior, i.e., $\boldsymbol{\omega}_a \sim \mathcal{N}(\mu_a, Cov_a)$. The process described above is incorporated into the critic, represented by the symbol $\curvearrowright$.

perspectives [29] for multi-agent settings, e.g., MAD3PG [30] approximates the discrete distribution to stabilize training. MMD-MIX [31] implicitly learns the distribution to capture the randomness in the interaction environments. Distributional MARL has the ability to capture uncertainty and is successfully applied to games [31] and robotic control [32].

2.3. Bayesian reinforcement learning

Bayesian methods for reinforcement learning have been widely investigated in single-agent settings. The primary motivations for incorporating Bayesian methods into reinforcement learning are summarized as follows: (1) it provides a natural description of uncertainty in the system and facilitates exploration by the agents; and (2) it can provide a mechanism for incorporating prior knowledge into the model. Unlike existing work, we have devised Bayesian reinforcement learning methods applied to multi-agent systems, where more uncertainty exists that needs to be captured.

In general, inspired by the distributional value function for single-agent settings, the core idea of our work is to directly approximate the continuous distribution of Q-value for multi-agent settings by incorporating Bayesian inference technology. It improves the generalization and robustness of the multi-agent reinforcement learning methods by capturing more uncertainty in the environments. Meanwhile, to reduce the model's dependence on interaction data, we inject Gaussian prior to improve the efficiency of our proposed method. It is the first effort that enables the MARL method to improve robustness and efficiency simultaneously.

3. Preliminaries

In this section, we will present the notation used in our proposed model, and then review traditional reinforcement learning and distributional value function, which provide fundamental theoretical support for our method.

3.1. Notation

We consider a multi-agent scenario described as Decentralized Partially Observable Markov Decision Process (Dec-POMDP) [33]. A Dec-POMDP for N agents is defined by a tuple $G = \langle S, A, T, O, R, \gamma, N \rangle$, where S is a set of environment global state, $A_1, \dots, A_N$ and $O_1, \dots, O_N$ are the set of actions and observations for each agent (Table 1). Assume ρ_0 is the distribution of the initial state s_0. At each timestep, each agent uses a stochastic policy π_{θ_i} to choose an action a_i, i.e., $\pi_{\theta_i} : o_i \mapsto a_i, a_i \in A_i$. The joint actions induce a transition to the next state according to the state transition function: $T := S \times A_1 \times \cdots \times A_N \times S \mapsto [0, 1]$, and agents

Table 1
Notations.

Symbol	Description
N	The number of agents
S	The set of the global states
$A_1, \dots, A_N$	The set of the actions for each agent
$O_1, \dots, O_N$	The set of the partial observations for each agent
ρ_0	The distribution of the initial state
r_i	Reward function for agent i
o_i	Partial observation for agent i
π_{θ_i}	Policy parameterized by θ_i for agent i
γ	Discount factor

obtain their partial observation $o_i : S \mapsto O_i$. Reward r_i for each agent is defined as $r_i := S \times A_i \mapsto \mathbb{R}$. When the episode is terminated, the flag $done$ is set. History transitions H are restored into replay buffer for training, which item is formatted as a tuple of $\langle o_i, a_i, o_i', r_i, done \rangle$. Discount factor γ balances immediate reward and long-term gain. The objective for each agent is to maximize its expectation of discounted accumulated reward $\mathbb{E}[R_T] = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t r_i^t\right]$.

3.2. Policy gradient and actor-critic methods

Policy Gradient (PG) [34] and Actor-Critic (AC) [35] provide the basic theory for training our model. Specifically, PG directly updates the parameter θ of the policy π_θ to maximize its objective, formulated as follows:

$$J(\theta) = \mathbb{E}_{s\sim p^\pi, a\sim\pi_\theta}[R] = \mathbb{E}_{s\sim p^\pi, a\sim\pi_\theta}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r_t\right], \tag{1}$$

where θ is updated toward the direction of $\nabla_\theta J(\theta)$.

In practice, calculating the expected return suffers from large variance, so AC replaces it with approximated value function and modifies $\nabla_\theta J(\theta)$ as:

$$\nabla_\theta J(\theta) = \mathbb{E}_{s\sim p^\pi, a\sim\pi_\theta}\left[\nabla_\theta \log(\pi_\theta(a|s)) Q_\omega^\pi(s,a)\right], \tag{2}$$

where ω is the parameter of the value function. Recently, Haarnoja et al. [36] considers the entropy-augmented objective function to encourage exploration of more space to search for the optimal policy, which augments $\nabla_\theta J(\theta)$ with a policy entropy term as:

$$\begin{aligned}\nabla_\theta J(\theta) =& \mathbb{E}_{s\sim p^\pi, a\sim\pi_\theta}[\nabla_\theta \log(\pi_\theta(a|s)) \\ & (-\alpha \log(\pi_\theta(a|s)) + Q_\omega(s,a) - V(s))],\end{aligned} \tag{3}$$

where α balances the significance of entropy against reward, and $V(s)$ is used as baseline. Accordingly, the target Q-value function is modified as:

$$y = r(s,a) + \gamma Q_{\bar{\omega}}(s', a') - \alpha \log(\pi_{\bar{\theta}}(a'|s')). \tag{4}$$

Notably, the Q-target function is augmented accordingly.

3.3. Distributional value function

The traditional RL learns the expected value to model cumulative returns, expressed as the action value function $Q(s,a)$ [1], which is formulated as:

$$Q_\omega(s,a) = \mathbb{E}[r(s,a)] + \gamma \mathbb{E}[Q_\omega(s',a')], \tag{5}$$

where value function is parameterized by ω, which is a specific value obtained by point estimation. By contrast, the distributional value function focuses on the entire distribution of the Q-value [25], which is defined similarly, but one of a distribution nature:

$$Q_\omega(s,a) \overset{D}{=} r(s,a) + \gamma Q_\omega(s',a'), \tag{6}$$

where $\overset{D}{=}$ indicates that the equation is to describe the distribution, and ω obeys a distribution. It provides information contained in Eq. (5), also implicates other more information beneficial to model improvement. Thus, we maintain the posterior distribution over value function parameters for multi-agent systems, which captures environment uncertainty inherently and is more competitive.

4. Methodology

In this section, we will detail our proposed method, Bayesian Multi-Agent Reinforcement Learning (BMARL). BMARL is devised to ameliorate the robustness and efficiency for multi-agent reinforcement learning, and the overall architecture of BMARL is illustrated in Fig. 1. It consists of three main components: (1) decentralized execution, (2) centralized distributional value function, and (3) Bayesian parametric modeling for distribution, which are introduced in the following subsection.

4.1. A CTDE framework for BMARL

To overcome the inherent non-stationary of the multi-agent environment, we devise our method based on centralized training and decentralized execution (CTDE) framework (Fig. 1a and b). Specifically, the training process of the proposed method incorporates extra joint information to broaden the agent's perspective, whereas, during policy execution, it merely uses agent's partial observations. Considering an environment with N agents, BMARL unites information by designing *Integrator* $f_i^{\phi}(\boldsymbol{o}, \boldsymbol{a}^{(-i)})$, which output is used as the input of Q-value function. It is a global information representation function integrating all agents' observations o_i and actions a_i, where $\boldsymbol{o} = (o_1, \ldots, o_N)$ and $\boldsymbol{a} = (a_1, \ldots, a_N)$, and the subscripts correspond to each agent indexed by $i \in \{1, \ldots, N\}$. $\boldsymbol{a}^{(-i)}$ is defined as the joint action vector excluding ith element a_i. The global information representation is defined as:

$$f_{\phi_i}(\boldsymbol{o}, \boldsymbol{a}^{(-i)}) = \frac{1}{N}\left(rep_s(o_i) + \sum_{j=1,\ j\neq i}^{N} rep_{sa}(o_j, a_j)\right), \tag{7}$$

where + denotes the concatenation operation. rep_s and rep_{sa} are representation functions which transform state and state–action pair into the same dimensions, respectively. Given an observation vector $\boldsymbol{o} = (o_1, \ldots, o_N)$ and an action vector $\boldsymbol{a} = (a_1, \ldots, a_N)$, the global information representation function f_{ϕ_i} outputs a joint information in compact form.

When the agents interact with environment, they use the parameterized policy π_{θ_i}, a two-layer MLP with 128 units for the hidden layer, to make the action decisions a_i only based on the private observations o_i. Its objective function is:

$$J(\theta_i) = \mathbb{E}_{o_i \sim p^{\pi}}[\alpha \log(\pi_{\theta_i}(a_i|o_i)) - (Q(o,a) - b(o))], \tag{8}$$

where α balances the significance of entropy against reward, Q is the action-value function which is detailed later, and $b(o)$ is used as baseline. In general, the CTDE framework effectively alleviates the instability of the multi-agent system and avoids the problem of limited scalability in the fully-centralized framework.

4.2. Centralized distributional value function

As we discussed in the previous section, the existing MARL algorithms have poor robustness, which mainly depends on the quality of the Q-value estimation. The Q-value can reflect the quality of the action under a specific state, assisting in policy improvement. The existing MARL methods approximate expected values to model cumulative returns, expressed as the action value function $Q(s,a)$. In this modeling process, considerable distribution information is lost, leading to a vulnerable model that is easily affected by spiky values. Meanwhile, the value function employs the temporal difference method to update the Q-value estimation with its sub-sequential estimation, which exacerbates the problem of approximation error. Further, it will affect the updated direction of policies, leading to suboptimal policy.

In view of this, we approximate the distribution over the Q-value, which describes the real and complete situation of Q-value. Specifically, we formulate the Q-value function as a linear function of the global information representation $f_i^{\phi}(\boldsymbol{o}, \boldsymbol{a}^{(-i)})$ for each agent and define it as:

$$Q_{\boldsymbol{W}_i}(\boldsymbol{o}, \boldsymbol{a}^{(-i)}) = f_{\phi_i}^T(\boldsymbol{o}, \boldsymbol{a}^{(-i)}) \cdot \boldsymbol{W}_i, \tag{9}$$

where $\boldsymbol{W}_i \in \mathbb{R}^{|A_i| \times h_d}$ is the parameter metric of the linear function for action $a_i, \forall a_i \in A_i$, and h_d denotes the hidden layer size of the value function network. We assume that the parameter matrix $\boldsymbol{W}_i$ is subject to Gaussian distribution. In this way, BMARL realizes the modeling of the centralized distributional value function.

Meanwhile, the centralized distributional value function satisfies the variant of the Bellman equation in the maximum entropy framework:

$$Q_{\boldsymbol{W}_i}(\boldsymbol{o}, \boldsymbol{a}) \overset{D}{=} r_t + \gamma Q_{\overline{\boldsymbol{W}}_i}(\boldsymbol{o}', \boldsymbol{a}') - \alpha \log(\pi_{\overline{\theta}_i}(a_i'|o_i')), \tag{10}$$

where $\overset{D}{=}$ indicates that both sides describe the distribution, $\overline{\boldsymbol{W}}_i$ and $\overline{\theta}_i$ denote parameters of target critic network and target policy network, respectively.

Temporal difference is used to learn the Q-value function by minimizing the joint loss function:

$$\begin{aligned} &\mathcal{L}(Q, Q^{target}) = \sum_{i=1}^{N} (Q_{\boldsymbol{W}_i}(\boldsymbol{o}, \boldsymbol{a}) - y_i)^2, \\ \text{where} \quad &y_i = r_i + \gamma Q_{\overline{\boldsymbol{W}}_i}(\boldsymbol{o}', \boldsymbol{a}') - \alpha \log(\pi_{\overline{\theta}_i}(a_i'|o_i')), \\ &a_i' \sim \pi_{\overline{\theta}_i}(\cdot\,|o_i'), \end{aligned} \tag{11}$$

where $Q_{\overline{\boldsymbol{W}}_i}$ and $\pi_{\overline{\theta}_i}$ denotes the target critic and the target policy for agent i. The next action a_i' is determined by target policy $\pi_{\overline{\theta}_i}$. As the parameters of critic network are updated frequently, which impacts the gradients of both critic networks and policy networks, we introduce target networks $Q_{\overline{\boldsymbol{W}}_i}$ and $\pi_{\overline{\theta}_i}$ to stabilize the training process. They have the same structure as the Q and π_{θ_i}, and their parameters are updated softly as follows:

$$\begin{aligned} \overline{\theta}_i &\leftarrow \tau \overline{\theta}_i + (1-\tau)\theta_i, \\ \overline{\phi}_i &\leftarrow \tau \overline{\phi}_i + (1-\tau)\phi_i. \end{aligned} \tag{12}$$

By substituting Eq. (9) into Eq. (10), the expanded form of the target Q-value y_i of agent i can be formulated as:

$$y_i = r_i + \gamma f_{\overline{\phi}_i}(\boldsymbol{o}', \boldsymbol{a}'^{(-i)})^T \cdot \overline{\boldsymbol{W}}_i - \alpha \log(\pi_{\overline{\theta}_i}(a_i'|o_i')), \tag{13}$$

where $\overline{\phi}_i$ is the parameter of *Integrator* used in the target critic. $\overline{\boldsymbol{W}}_i \in \mathbb{R}^{|A_i| \times h_d}$ denotes the parameter of the target critic linear-layer for action $a_i', \forall a_i' \in A_i$.

4.3. Bayesian parametric modeling for distributional value estimation

Instead of learning the expectation of cumulative returns by point estimation, BMARL models the entire distribution over cumulative returns, expressed as action value function Q. Compared with estimating

the statistical value of data (e.g., mean or median), the distribution can better depict the overall nature of data and assist in mining contained information in-depth. Meanwhile, unlike existing works obtaining discrete distributions, it maintains a continuous distribution and does not require additional task information for setting the distribution range.

More specifically, unlike traditional multi-agent reinforcement learning methods, BMARL approximates the distribution over value function by combining with Bayesian inference. Bayesian inference is one of the most important techniques in statistics and it calculates the posterior distribution based on samples. Inspired by the Bayesian inference, BMARL employs the Bayesian linear regression for Q-value function approximation, as shown in Fig. 1c. In this way, BMARL obtains an approximated posterior distribution over parameter metric $\boldsymbol{W}$ and Q-value function, called *Bayesian Distributional Value Function*, which offers a natural way to unfold experimental distributions to get the best estimates of the true ones. Each agent injects the prior knowledge into the parameter metric $\boldsymbol{W}_i$, and when initializing, it is assumed that $\boldsymbol{W}_i$ is subject to Gaussian distribution with mean zero and variance σ^2, i.e., $\boldsymbol{W}_i \sim \mathcal{N}(0, \sigma^2)$, $\forall a \in A_i$. Meanwhile, the distribution over the Q-value function is determined by $\boldsymbol{W}_i$. Furthermore, noise perturbation is taken into consideration, and we assume that the noise is i.i.d and satisfies Gaussian distribution: $p(\epsilon) = \mathcal{N}(0, \sigma_\epsilon)$. Thus, the actual observed value of the Q-value is as follows:

$$y_a = Q_{\boldsymbol{W}_i}\left(\boldsymbol{o}, (a, \boldsymbol{a}^{(-i)})\right) + \epsilon = f_{\phi_i}^T(\boldsymbol{o}, \boldsymbol{a}^{(-i)}) \cdot \boldsymbol{W}_i + \epsilon, \quad \text{where} \quad \epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2). \tag{14}$$

Algorithm 1: Bayesian Multi-Agent Reinforcement Learning (BMARL)

Input: The number of the agent N
Output: Parameters for policy π_i and critic Q_i
1 Initialize network parameters $\theta_i, \overline{\theta}_i, \phi_i, \overline{\phi}_i, \boldsymbol{W}_i, \overline{\boldsymbol{W}}_i$ for each agent i
2 $\forall a \in A_i$, initialize μ_a, Cov_a for each agent i
3 Create replay buffer $RB \leftarrow \{\}$
4 **for** $episode = 1, 2, \cdots, max\_episodes$ **do**
5 $s_0 \sim \rho_0$, get the initial o_i for each agent i
6 **if** $RB! = Empty$ **then**
7 Sample a minibatch of B samples from RB
8 For each agent i, update the posterior distribution μ_a, Cov_a, $\forall a$, by using Equation (16)
9 Update target parameter $\overline{\boldsymbol{W}}_i$: $\forall \overline{\boldsymbol{\omega}}_a \in \overline{\boldsymbol{W}}_i$, $\overline{\boldsymbol{\omega}}_a \leftarrow \mu_a$
10 Sample $\boldsymbol{W}_i$ for each agent i, where $\forall \boldsymbol{\omega}_a \in \boldsymbol{W}_i$, $\boldsymbol{\omega}_a \sim \mathcal{N}(\mu_a, Cov_a), a \in A_i$
11 **for** $t = 1, 2, \cdots, max\_length$ **do**
12 Select actions $a_i \sim \pi_i(\cdot|o_i)$
13 Execute actions $\boldsymbol{a}$, receive $\boldsymbol{o'}$ and $\boldsymbol{r}$
14 Store transitions in RB, and set $\boldsymbol{o} \leftarrow \boldsymbol{o'}$
15 **if** $episode > threshold$ **then**
16 Sample a K-minibatch from RB
17 Compute the target value y by Equation (13)
18 Update critic parameter: $\phi_i \leftarrow \phi_i - \alpha \nabla_{\phi_i} (y - \boldsymbol{W}_i^T \cdot f_{\phi_i}(\boldsymbol{o}, \boldsymbol{a}^{(-i)}))^2$
19 Update policy parameter: $\theta_i \leftarrow \theta_i - \alpha \nabla_{\theta_i} J(\theta_i)$
20 Update target network parameters:
21 $\overline{\theta}_i \leftarrow \tau \overline{\theta}_i + (1-\tau)\theta_i$
22 $\overline{\phi}_i \leftarrow \tau \overline{\phi}_i + (1-\tau)\phi_i$

According to Bayes' theorem, we know that posterior $P(\boldsymbol{W}|H)$ is proportional to $P(H|\boldsymbol{W})$ times $P(\boldsymbol{W})$, where $P(\boldsymbol{W})$, $P(H|\boldsymbol{W})$ denote the prior and the likelihood, respectively. H is the interaction data. We assume the $P(\boldsymbol{W})$ satisfies Gaussian distribution. Since the Q-value function is modeled as a linear function of the global information representation $f_i^\phi(\boldsymbol{o}, \boldsymbol{a}^{(-i)})$, the likelihood $P(H|\boldsymbol{W})$ also obeys a Gaussian distribution. Through iterative using Bayes' theorem, the understanding of $\boldsymbol{W}$ is refined, and finally the posterior distribution over the parameter metric $\boldsymbol{W}$ is trustworthy. Specifically, BMARL deploys Bayesian linear regression to update the $\boldsymbol{W}_i$ posterior distribution for each agent i. The parameter metric $\boldsymbol{W}_i$ is formulated as:

$$\boldsymbol{W}_i = \left[\boldsymbol{\omega}_{a_1}, \boldsymbol{\omega}_{a_2}, \ldots, \boldsymbol{\omega}_{a_{|A_i|}}\right]^T, \tag{15}$$

where $\boldsymbol{\omega}_{a_k} \in \mathbb{R}^{1 \times h_d}$ is a parameter vector for action $\forall a \in A_i$, and action index $k \in \{1, \ldots, |A_i|\}$. For brevity, it is abbreviated as $\boldsymbol{\omega}_a$ sometimes. The Gaussian distribution is self-conjugate with the Gaussian likelihood, which can directly give the closed form of the posterior distribution. Thus, given a batch of interaction data $H = \langle o, a, r, o' \rangle$, the distribution of $\boldsymbol{\omega}_a \in \boldsymbol{W}_i$, $\forall a \in A_i$, is empirically approximated as follows:

$$P(\boldsymbol{\omega}_a|H) \sim \mathcal{N}(\mu_a, Cov_a), \quad \text{where} \quad \mu_a = \frac{1}{\sigma_\epsilon^2} Cov_a \mathcal{F}_a \mathcal{Y}_a, \quad Cov_a = \left(\frac{1}{\sigma_\epsilon^2} \mathcal{F}_a \mathcal{F}_a^T + \frac{1}{\sigma^2} I\right)^{-1}, \tag{16}$$

where $\mathcal{F}_a$ is the integration of f_a^ϕ in the batch, and $f_a^\phi = f^\phi(\boldsymbol{o}, (a, \boldsymbol{a}^{(-i)}))$ is the representation of global information and the agent i selects action $a_i = a$ to execute. $\mathcal{F}_a^T$ is the transpose of $\mathcal{F}_a$. $\mathcal{Y}_a$ is the vector of target value in batch. Accordingly, the computational complexity is evaluated as $O(nh_d^2 + h_d^3)$, where n is the number of samples, and h_d is the dimension of input embedding features.

As shown in Fig. 1c, BMARL gets the statistical information of $\boldsymbol{\omega}_a, \forall a \in A_i$, from the empirical data and then searches for interest point $\boldsymbol{\omega}^*$ in the potential area to form $\boldsymbol{W}_i^*$ as the value function parameter: $Q_{\boldsymbol{W}_i^*}(\boldsymbol{o}, \boldsymbol{a})$. When given the same input, the value function $Q_{\boldsymbol{W}_i^*}$ may output a slightly different estimation according to the current distribution, which will cover as many values as possible.

Algorithm 1 summarizes the whole process of BMARL in pseudo-code. At the beginning of episodes, BMARL samples $\boldsymbol{W}_i$, for each agent i, according to its distribution (Line 10) and implements Q-value function correspondingly, which is beneficial to provide comprehensive information for policies and further improves the generalization of BMARL. Periodically, Lines 6–9 play back the replay buffer and update the posterior distribution $\boldsymbol{W}_i$, gradually approaching the true distribution.

In general, through Bayesian inference to approximate a distributional value function, BMARL naturally captures the uncertainty, which is conducive to generalization enhancement. The estimation error of Q-value can also be greatly reduced, which enables assist policy improvement efficiently. Furthermore, BMARL incorporates the prior into the value function, bringing more information than the data itself, and improving efficiency. In a nutshell, BMARL is more competitive in complex and stochastic MARL tasks.

5. Experiments

In this section, we investigate our methods on several multi-agent scenarios. We first describes the experimental setup, and then we conduct extensive experiments to evaluate the performance and superiority of the proposed BMARL. Finally, we perform the ablation study and sensitivity analysis.

5.1. Experimental setup

Environments Three benchmark multi-agent environments, including cooperative and competitive agents [11,20], are selected to evaluate our method. As shown in Fig. 2, environments are composed of agents and landmarks, where rewards are related to the distance between the agent and the target entity.

Cooperative Communication (Co-Comm) [11]: It consists of landmarks and two cooperative agents, i.e., listener and speaker in Fig. 2(a). The speaker delivers information about the target landmark to assist listener in navigating to destination.

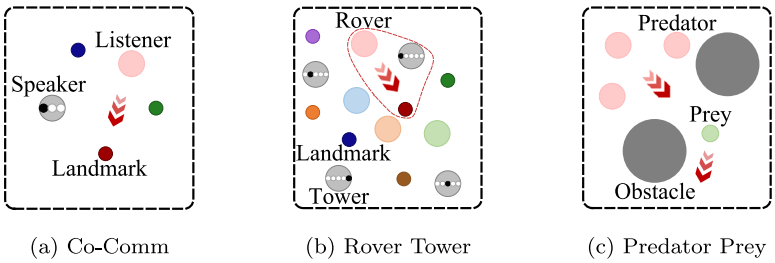


Fig. 2. Three multi-agent benchmark scenarios. (a) Co-Comm: the speaker (gray) assists the listener in navigating to the correct landmark. (b) Rover Tower: agents match randomly in pairs and collaborate to finish the task like Co-Comm. (c) Predator Prey: the prey (green) tries to escape from the predators (red). Obstacles (black) help the prey to block the way of predators. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

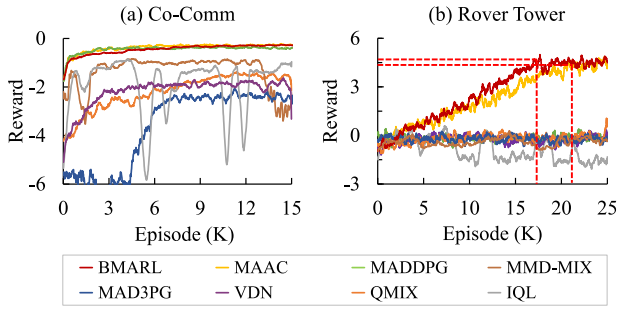


Fig. 3. Overall performance comparison on Cooperative Communication (a) and Rover Tower (b) in terms of reward. Red dotted line marks the model convergence information.

Table 2
The hyper-parameter settings of the BMARL.

	Hyper-parameter	Value
Actor	Hidden layers of actor networks	[128, 128]
	Learning rate of actor	0.0005
	The optimizer of actor networks	Adam
Critic	Hidden layers of Integrator networks	[128]
	Learning rate of critic	0.0005
	The optimizer of critic networks	Adam
	The size of the linear layer in critic networks	128
	The Gaussian variance σ of the prior	0.001
Others	Buffer length	10^6
	Discount factor γ	0.99
	Soft update factor τ	0.005

Rover Tower [20]: It is a variant of Cooperative Communication with more entities, in which the agents are randomly paired, e.g., in the red circle in Fig. 2(b), and cooperate to complete the task. A '+10' reward will be given when an agent reaches the goal.

Predator Prey [11]: It is a competitive scenario, where the predators (red) aim to catch the prey (green), while the obstacles (black) help the prey to block the ways of predators (Fig. 2(c)). It will be rewarded if the prey escapes; otherwise, it will be punished.

Baselines Seven state-of-the-art MARL comparisons are selected to demonstrate the superiority of our proposed method, which are detailed as follows:

MAAC [20]: It is based on CTDE. A multi-head attention mechanism is incorporated to get more relevant information and stabilize the training.

MADDPG [11]: It combines DDPG with CTDE. Besides, agents learn approximate models of other agents to improve their stability during training.

MMD-MIX [31]: It implicitly learns the value distribution for MARL by modifying the mixing net in QMIX with a set of particles and employs maximum mean discrepancy for value function update.

MAD3PG [30]: It extends MADDPG by maintaining a discrete distributional value function with fixed atoms to mitigate that the policies are fragile.

VDN [19]: It decomposes the team-reward into the sum of individual rewards. To solve the challenge caused by partial observation, it uses recurrent neural network to refer to historical information.

QMIX [12]: It serves the same purpose as VDN to decompose the team reward. Note that QMIX estimates joint action values as a nonlinear combination of per-agent values and promises the monotonic.

IQL [6]: It takes advantage of the successful RL to enable multi-agent tasks. Then, it merely applies RL methods to train the agents individually.

Hyper-parameter Settings We conduct a variety of experiments on these scenarios to verify the validity of our method. And the detailed parameter settings of BMARL are shown in Table 2, which are consistent in different experiments.

5.2. Results and analysis

To validate the BMARL in terms of robustness and effectiveness, we conduct a series of experiments on both cooperative and competitive scenarios, comparing it with state-of-the-art MARL approaches. Especially, model performance is measured by the online per-step average reward achieved by the agents; and robustness is measured by the degree of fluctuation in rewards during training. The detailed analysis is as follows.

5.2.1. Results in cooperative scenarios

To verify the effectiveness of BMARL in collaborative tasks, we randomly select five seeds to train each model on Cooperative Communication and Rover Tower. Fig. 3 presents the overall performance for all methods, and it is obvious that BMARL achieves the same level of performance or even better than state-of-the-art methods in most cases.

Specifically, in Cooperative Communication, BMARL shows the same strength as MAAC and MADDPG in convergence rate and accumulative reward (Fig. 3a). In Rover Tower, BMARL can converge to the maximum reward faster than all these comparisons (Fig. 3b). Notably, MADDPG, MMD-MIX, MAD3PG, VDN, QMIX, and IQL are obviously weak compared with BMARL and MAAC, especially in Rover Tower. The possible reasons are: (1) Rover Tower is more complicated than Co-Comm; (2) MADDPG suffers from value function approximation errors; (3) IQL fails to capture the inter-relation among agents; (4) MMD-MIX, VDN, and QMIX only apply to tasks with global rewards, but MMD-MIX with the distributional value function performs better than VDN and QMIX; (5) MAD3PG is hard to train and converge while approximating the distribution of value function; and (6) MAAC with a complex architecture entirely relies on data information, which slows down its training. In summary, BMARL considers the distribution of value function and prior information when designing the model, which is more competitive in effectiveness and efficiency, especially in dealing with complex tasks.

Based on the above comparable results, we present more in-depth analyses to verify the model's superiority in terms of robustness and efficiency. As shown in Figs. 4 and 5, the shaded region indicates the real

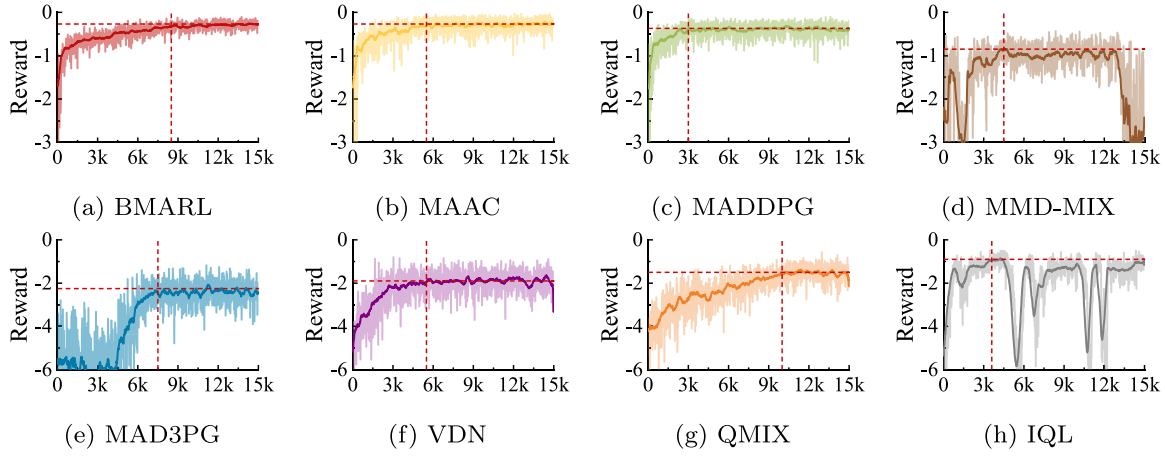


Fig. 4. Learning curves comparison on Cooperative Communication for all methods. The X-axis denotes the episode number and Y-axis denotes the per-step average reward (online). The solid line and shaded region represent the mean and standard deviation, respectively, across five runs. Red dotted line marks convergence information. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

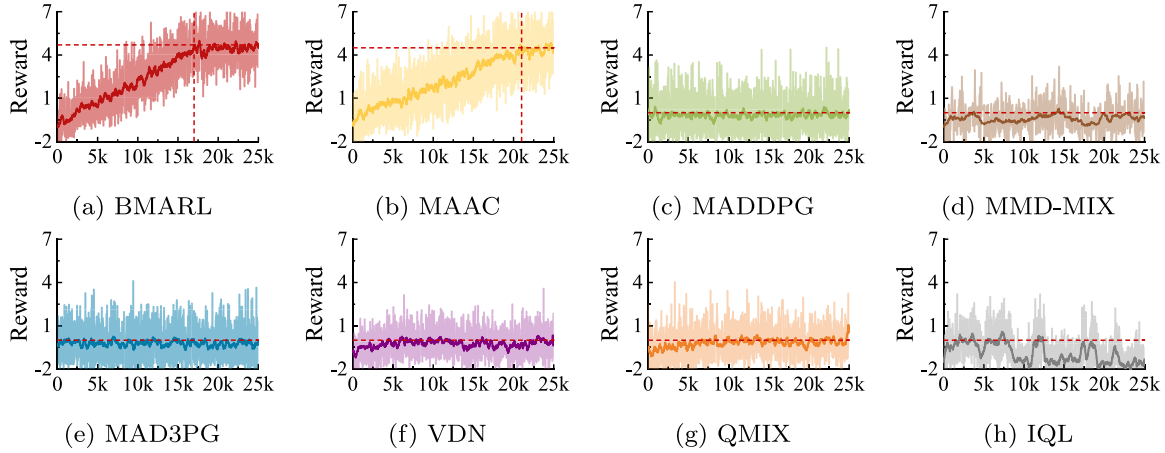


Fig. 5. Learning curves comparison on Rover Tower for all methods. The X-axis denotes the episode number and Y-axis denotes the per-step average reward (online). The solid line and shaded region represent the mean and standard deviation, respectively, across five runs. Red dotted line marks convergence information. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

reward achieved during training, where a small area indicates strong robustness. The solid line denotes the average reward indicating the overall convergence trend, and the red dotted line marks the point that a model converged. Specifically, BMARL converges smoothly during training, and the light-colored area is smaller than all the comparisons (Figs. 4(a) and 5(a)), which intuitively reflects the advantage of BMARL in robustness. Meanwhile, MAAC and MADDPG with high rewards converge slightly faster than BMARL in a simple scenario, e.g., Co-Comm. However, the fluctuation of convergence trends increases, and the convergence speed decreases when encountering a complicated scenario, e.g., Rover Tower. To make the matter worse, MADDPG, MMD-MIX, MAD3PG, VDN, QMIX, and IQL are almost invalid. Although MMD-MIX and MAD3PG are also distribution MARL, they perform not as well as BMARL because their discrete distributions are not comprehensive enough as the continuous distribution approximated by BMARL. Overall, our method maintains high robustness and efficiency in most cases, especially in confronting complicated challenges. In summary, the advantages of BMARL are: (1) the distributional value function estimated by Bayesian inference is beneficial to the robustness; and (2) prior information introduced into the model improves the training efficiency.

More specifically, Tables 3 and 4 show the average reward μ and standard deviation σ to quantitatively demonstrate the advantage of BMARL in terms of robustness. Particularly, we use the coefficient of variation c_v to denote the fluctuation degree of the algorithm, which is defined as $c_v = \frac{\sigma}{|\mu|} \times 100\%$, i.e., method with strong robustness will obtain a small c_v [37]. For example, in Rover-Tower, the μ and σ for BMARL are 4.61 and 0.87 and for MAAC are 4.29 and 1.04, respectively. The corresponding coefficient of variation c_v for BMARL and MAAC are 18.87% and 24.24%, respectively. Similarly, in Co-Comm, the c_v for BMARL and MAAC are 33.33% and 46.66%, respectively. It indicates that BMARL is more stable than MAAC.

Besides, as shown in Fig. 6, we plot the max–min normalization of the reward deviation in different cooperative scenarios. The lower value of max–min normalization indicates that the corresponding method is more stable. Obviously, BMARL has relatively small max–min normalization compared with MAAC and MADDPG, i.e., BMARL is relatively robust, which is according to our prior findings. Although IQL has the lowest value on Rover Tower, it cannot achieve excellent performance.

To sum up, whether in simple or complicated scenarios, the overall performance of BMARL demonstrates its superiority in robustness and efficiency in handling cooperative tasks.

5.2.2. Results in competitive scenarios

To validate the competitiveness of the proposed model, we conduct another experiment on competitive scenario, Predator Prey, compared with four baselines, i.e., MAAC, MADDPG, MAD3PG, and IQL. QMIX,

Table 3
The statistics of reward on benchmark scenarios, Co-Comm. Each performance value is the average of 5 runs with different random seeds, respectively. It is expressed in the form of mean ± standard deviation. The best results are highlighted in boldface.

Environment	Method	3K-Episode	6K-Episode	9K-Episode	12K-Episode	15K-Episode
Co-Comm	BMARL	−0.6 ± 0.11	−0.43 ± 0.09	−0.33 ± 0.09	**−0.29 ± 0.08**	**−0.24 ± 0.08**
	MAAC	−0.42 ± 0.22	**−0.33 ± 0.18**	**−0.29 ± 0.17**	−0.29 ± 0.16	−0.30 ± 0.14
	MADDPG	**−0.37 ± 0.14**	−0.35 ± 0.12	−0.40 ± 0.14	−0.33 ± 0.13	−0.39 ± 0.15
	MMD-MIX	−1.23 ± 0.49	−0.97 ± 0.19	−0.96 ± 0.23	−0.93 ± 0.21	−2.43 ± 1.21
	MAD3PG	−6.21 ± 1.50	−2.92 ± 0.48	−2.55 ± 0.47	−2.36 ± 0.52	−2.48 ± 0.40
	VDN	−2.19 ± 0.51	−1.96 ± 0.41	−1.99 ± 0.53	−1.75 ± 0.38	−1.87 ± 0.47
	QMIX	−2.78 ± 0.58	−2.05 ± 0.38	−1.78 ± 0.37	−1.46 ± 0.25	−1.50 ± 0.34
	IQL	−1.27 ± 0.23	−1.57 ± 0.56	−1.29 ± 0.29	−3.85 ± 1.97	−1.50 ± 0.20

Table 4
The statistics of reward on benchmark scenarios, Rover Tower. Each performance value is the average of 5 runs with different random seeds, respectively. It is expressed in the form of mean ± standard deviation. The best results are highlighted in boldface.

Environment	Method	5K-Episode	10K-Episode	15K-Episode	20K-Episode	25K-Episode
Rover Tower	BMARL	0.48 ± 0.78	**2.31 ± 0.97**	**3.97 ± 1.12**	**4.49 ± 0.88**	**4.61 ± 0.87**
	MAAC	**0.80 ± 1.03**	1.59 ± 1.27	2.73 ± 1.22	3.65 ± 1.26	4.29 ± 1.04
	MADDPG	−0.24 ± 0.88	−0.26 ± 0.98	−0.07 ± 0.98	0.03 ± 1.11	0.19 ± 0.91
	MMD-MIX	−0.79 ± 0.89	−0.63 ± 0.76	−0.09 ± 0.84	−0.10 ± 1.17	−0.33 ± 1.02
	MAD3PG	−0.10 ± 0.99	−0.25 ± 0.99	0.04 ± 1.19	−0.44 ± 1.08	−0.06 ± 1.21
	VDN	−0.25 ± 0.92	−0.01 ± 0.77	−0.15 ± 1.04	−0.16 ± 0.72	−0.03 ± 0.88
	QMIX	−0.41 ± 0.96	0.17 ± 0.93	0.01 ± 0.92	−0.10 ± 0.71	0.22 ± 1.07
	IQL	−0.73 ± 0.91	−1.46 ± 0.56	−1.34 ± 0.70	−1.63 ± 0.89	−1.68 ± 0.65

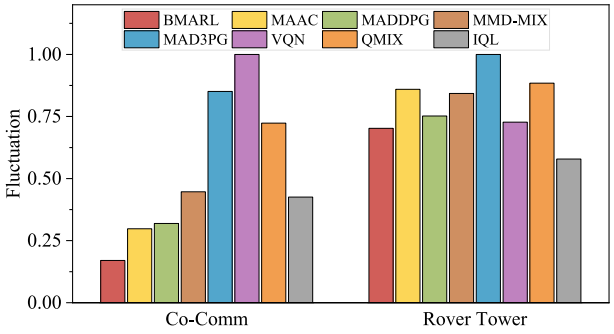


Fig. 6. Overall robustness comparison among methods on cooperative scenarios. The Y-axis represents the max–min normalization of reward deviation.

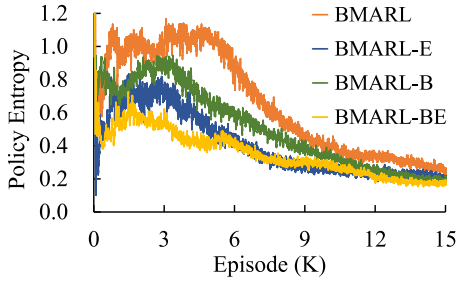


Fig. 7. The impact of different variants of BMARL, namely Bayesian parametric modeling for distribution and entropy-augmented objective function, in terms of policy entropy on Rover Tower. Higher entropy means that the method captures more uncertainty.

VDN, and MMD-MIX are not compared in this experiment since they are designed for collaborative scenarios.

Table 5 shows the average reward R_μ and standard deviation R_σ indicating the competitiveness and robustness, respectively. The competitive ability of IQL, MADDPG, and MAD3PG is inadequate because: (1) IQL cannot capture relational information among agents, which will disrupt their policies execution; (2) MADDPG cannot provide accurate action-value estimation and reliable guidance for policy training; and

Table 5
Average reward for prey on Predator Prey. Each performance value is the average of 5 runs with different random seeds. R_μ and R_σ denote the mean and the standard deviation, respectively. The best results are highlighted in boldface.

	BMARL	MAAC	MADDPG	MAD3PG	IQL
R_μ	−0.69	**−0.65**	−3.41	−0.95	−4.45
R_σ	**±0.20**	±0.38	±1.91	±0.41	±0.41

(3) MAD3PG cannot provide comprehensive information due to it uses fixed atoms to approximate the distribution discretely and MAAC is competitive to achieve a relatively high accumulative reward, which is close to ours, but its standard deviation is almost twice that of ours. Generally speaking, BMARL achieves relatively high average reward and the lowest standard deviation, demonstrating the effectiveness and robustness in handling competitive tasks.

5.3. Ablation study

In this subsection, we first conduct an ablation study on Rover Tower to verify the necessity of each component designed in the model, namely Bayesian parametric modeling for distributional value function and entropy-augmented objective function. To better demonstrate its superiority, we devise the three variants of BMARL as follows: (1) **BMARL-E**: It removes the entropy term in the objective function. (2) **BMARL-B**: It approximates the value function, using determined parameter values learned by point estimation, not distributions. (3) **BMARL-BE**: It is the simplest method and does not add either of the above mechanisms.

Fig. 7 shows the policy entropy changes during 15 thousand episodes, reflecting the degree of uncertainty about the current policy. BMARL-BE obtains the lowest entropy since only part of the uncertainty is considered, which fails to explore the entire space and likely converges to suboptimal policies. BMARL-B and BMARL-E, considering either distributional value function or entropy, are capable of capturing uncertainty and perform better than BMARL-BE. BMARL achieves the best policy entropy against all these variants, indicating that the Bayesian distributional value function and entropy considered are crucial and beneficial to the model.

Besides, we investigate whether the way of maintaining the distributions over the Q-value function impacts the model performance.

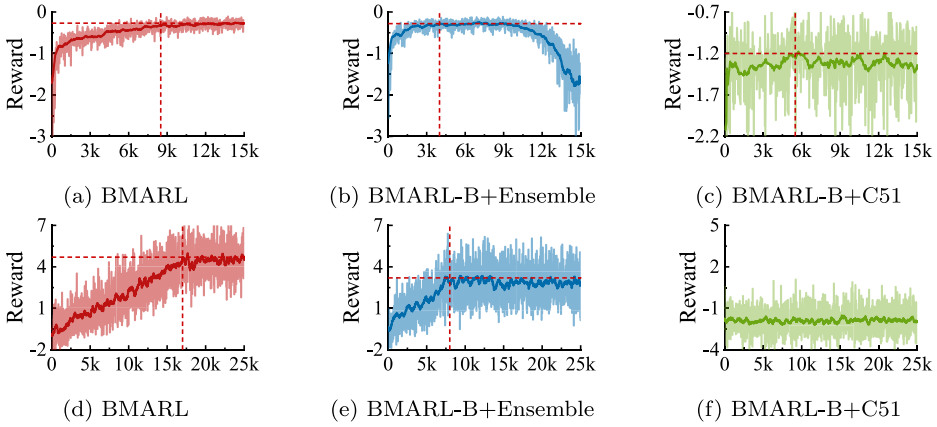


Fig. 8. The impact of different variants of BMARL on Cooperative Communication (a)–(c) and Rover Tower (d)–(f). The X-axis denotes the episode number and Y-axis denotes the per-step average reward (online).

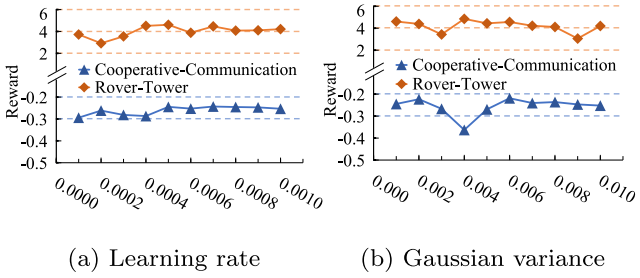


Fig. 9. Parameter sensitivity study on the learning rate and Gaussian variance. The X-axis denotes the setting of the learning rate (or Gaussian variance). The Y-axis denotes the per-step average reward (online).

We detail the variants of BMARL in terms of distributional value function module as follows: (1) **BMARL**: It is our proposed model which employs Bayesian Linear Regression to calculate the distribution over the Q-value function. It approximates the continuous distribution without additional knowledge. (2) **BMARL-B+Ensemble**: It makes combinations of multiple Q-value estimators to approximate the discrete distribution and has no demand for prior knowledge about tasks. (3) **BMARL-B+C51**: It adopts the idea of C51 method [25] to maintain the distributional value function. Specifically, it approximates the discrete distribution by deploying 51 atoms within the range of distribution and updating the probability for each atom.

Fig. 8 presents the learning curve on cooperative scenarios. We find that (1) BMARL-B+C51 does not perform well, mainly because it needs additional information to tune the distribution range manually, which is closely related to its performance; (2) BMARL-B+Ensemble performs better than BMARL-B+C51 but is less robust than BMARL, which is reflected by the area of the shaded region; and (3) BMARL performs best against all these variants with robustness, owing to the continuous distribution maintained by Bayesian linear regression. The primary reason is that the distribution obtained by Bayesian linear regression captures more uncertainty, which is beneficial to model performance and robustness.

5.4. Sensitivity analysis

In this subsection, we mainly investigate the impact of two primary parameters, i.e., learning rate lr and Gaussian variance σ, on the proposed model. To ensure the uniqueness of the effect from the parameters, we employ single-parameter sensitivity analysis by varying one parameter while fixing the others each time. The results of the experiment are shown in Fig. 9, and it is not difficult to see that BMARL can always maintain stable and high-level performance under different parameter settings on Cooperative Communication and Rover Tower.

First, we analyze the sensitivity of the learning rate lr, which determines the step size at each iteration. As illustrated in Fig. 9(a), BMARL achieves the highest reward on both scenarios when $lr = 5 \times 10^{-4}$, and the performance varies little with the change of the learning rate settings, which demonstrates that BMARL is not sensitive to the learning rate. In terms of the Gaussian variance σ, which controls the distribution information, BMARL also maintains a relatively stable working state in most of the cases with different σ (Fig. 9(b)). It verifies the effectiveness and robustness of the proposed BMARL model.

6. Conclusion

This paper presents a novel algorithm called BMARL to settle the challenge of poor robustness in MARL. We adopt Bayesian linear regression to refine the posterior distribution over the value function parameters. In this way, it obtains comprehensive Q-value information and innately captures the uncertainty of the environments, enabling BMARL to be robust to various scenarios. Meanwhile, we incorporate the Gaussian prior into BMARL, reducing the dependency of interaction data and thus, speeding up the learning efficiency of BMARL. Empirically, validations across cooperative and adversarial scenarios demonstrate the superiority of BMARL compared with state-of-the-art methods. Furthermore, we conduct ablation study and sensitivity analysis for BMARL to make it more convincing.

In terms of limitations, BMARL makes an assumption of using Gaussian prior as a prior for the value function. Besides, we validate BMARL on multi-agent benchmark environments but have not applied it to a real-world scenario. In the future, the proposed BMARL will be readily investigated by replacing the Gaussian prior with a general form. Meanwhile, exploring more details of the optimizing process of BMARL and applying it to more complex scenarios will also be new directions in the future.

Declaration of competing interest

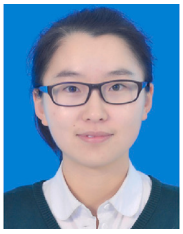
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

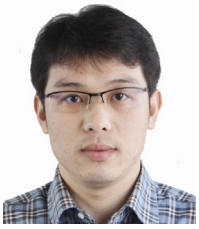
Data will be made available on request

References

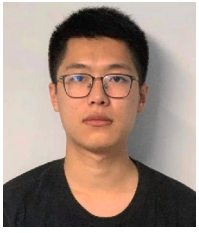
[1] V. Mnih, K. Kavukcuoglu, D. Silver, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.
[2] T. Lillicrap, J. Hunt, et al., Continuous control with deep reinforcement learning, in: Proceedings of the ICML, 2016, pp. 1501–1506.
[3] W. Zhang, N. Zhang, J. Yan, G. Li, X. Yang, Auto tuning of price prediction models for high-frequency trading via reinforcement learning, Pattern Recognit. (2022) 108543.
[4] M. Sun, J. Xiao, E.G. Lim, Y. Xie, J. Feng, Adaptive ROI generation for video object segmentation using reinforcement learning, Pattern Recognit. 106 (2020) 107465.
[5] X. Wang, L. Ke, Z. Qiao, X. Chai, Large-scale traffic signal control using a novel multiagent reinforcement learning, IEEE Trans. Cybern. 51 (1) (2021) 174–187.
[6] M. Tan, Multi-agent reinforcement learning: Independent versus cooperative agents, in: Proceedings of the ICML, 1993, pp. 330–337.
[7] S. Omidshafiei, J. Pazis, C. Amato, et al., Deep decentralized multi-task multi-agent reinforcement learning under partial observability, in: Proceedings of the ICML, 2017, pp. 2681–2690.
[8] C. Claus, C. Boutilier, The dynamics of reinforcement learning in cooperative multiagent systems, in: Proceedings of the IAAI, 1998, pp. 746–752.
[9] L. Busoniu, R. Babuska, B. De Schutter, A comprehensive survey of multiagent reinforcement learning, IEEE Trans. Syst. Man Cybern. C 38 (2) (2008) 156–172.
[10] Y. Zhan, H.B. Ammar, M.E. Taylor, Scalable lifelong reinforcement learning, Pattern Recognit. 72 (2017) 407–418.
[11] R. Lowe, Y.I. Wu, et al., Multi-agent actor-critic for mixed cooperative-competitive environments, in: Advances in NeurIPS, Vol. 30, 2017, pp. 6379–6390.
[12] T. Rashid, M. Samvelyan, et al., QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning, in: Proceedings of the ICML, 2018, pp. 4295–4304.
[13] Y. Liu, W. Wang, Y. Hu, et al., Multi-agent game abstraction via graph attention neural network, in: Proceedings of the AAAI, Vol. 33, 2020, pp. 7211–7218.
[14] Y. Niu, R. Paleja, et al., Multi-agent graph-attention communication and teaming, in: Proceedings of the AAMAS, 2021, pp. 964–973.
[15] S. Li, Y. Wu, et al., Robust multi-agent reinforcement learning via minimax deep deterministic policy gradient, in: Proceedings of the AAAI, 2019, pp. 4213–4220.
[16] R. Saphal, B. Ravindran, et al., SEERL: Sample efficient ensemble reinforcement learning, in: Proceedings of the AAMAS, 2021, pp. 1100–1108.
[17] J. Cui, Y. Liu, A. Nallanathan, Multi-agent reinforcement learning-based resource allocation for UAV networks, IEEE Trans. Wireless Commun. 19 (2) (2019) 729–743.
[18] J. Foerster, I.A. Assael, N. de Freitas, et al., Learning to communicate with deep multi-agent reinforcement learning, in: Advances in NeurIPS, Vol. 29, 2016, pp. 2137–2145.
[19] P. Sunehag, G. Lever, A. Gruslys, et al., Value-decomposition networks for cooperative multi-agent learning, in: Proceedings of the AAMAS, 2018, pp. 2085–2087.
[20] S. Iqbal, F. Sha, Actor-attention-critic for multi-agent reinforcement learning, in: Proceedings of the ICML, 2019, pp. 2961–2970.
[21] S. Fujimoto, H. Hoof, et al., Addressing function approximation error in actor-critic methods, in: Proceedings of the ICML, 2018, pp. 1587–1596.
[22] L. Pan, Q. Cai, Q. Meng, W. Chen, L. Huang, Reinforcement learning with dynamic Boltzmann softmax updates, in: Proceedings of the IJCAI, 2020, pp. 1992–1998.
[23] H. Van Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double q-learning, in: Proceedings of the AAAI, 2016, pp. 2094–2100.
[24] I. Osband, C. Blundell, A. Pritzel, B. Van Roy, Deep exploration via bootstrapped DQN, in: Advances in NeurIPS, Vol. 29, 2016, pp. 4033–4041.
[25] M.G. Bellemare, W. Dabney, et al., A distributional perspective on reinforcement learning, in: Proceedings of the ICML, 2017, pp. 449–458.
[26] W. Dabney, M. Rowland, et al., Distributional reinforcement learning with quantile regression, in: Proceedings of the AAAI, Vol. 32, 2018.
[27] K. Azizzadenesheli, E. Brunskill, A. Anandkumar, Efficient exploration through bayesian deep q-networks, in: 2018 Information Theory and Applications Workshop, ITA, 2018, pp. 1–9.
[28] W. Dabney, G. Ostrovski, D. Silver, et al., Implicit quantile networks for distributional reinforcement learning, in: Proceedings of the ICML, 2018, pp. 1096–1105.
[29] W.-F. Sun, C.-K. Lee, C.-Y. Lee, A distributional perspective on value function factorization methods for multi-agent reinforcement learning, in: Proceedings of the AAMAS, 2021, pp. 1671–1673.
[30] R. Li, R. Wang, T. Tian, F. Jia, Z. Zheng, Multi-agent reinforcement learning based on value distribution, in: Journal of Physics: Conference Series, 2020, 012017.
[31] Z. Xu, D. Li, Y. Bai, G. Fan, MMD-MIX: Value function factorisation with maximum mean discrepancy for cooperative multi-agent reinforcement learning, in: IJCNN, 2021, pp. 1–7.
[32] W. Sheng, H. Guo, et al., PD-FAC: Probability density factorized multi-agent distributional reinforcement learning for multi-robot reliable search, IEEE Robot. Autom. Lett. 7 (4) (2022) 8869–8876.
[33] F.A. Oliehoek, C. Amato, The decentralized POMDP framework, in: A Concise Introduction to Decentralized POMDPs, 2016, pp. 11–32.
[34] R.S. Sutton, D.A. McAllester, S.P. Singh, Y. Mansour, Policy gradient methods for reinforcement learning with function approximation, in: Advances in NeurIPS, 2000, pp. 1057–1063.
[35] V.R. Konda, J.N. Tsitsiklis, Actor-critic algorithms, in: Advances in NeurIPS, 2000, pp. 1008–1014.
[36] T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, in: Proceedings of the ICML, 2018, pp. 1861–1870.
[37] A. Fachantidis, M.E. Taylor, I. Vlahavas, Learning to teach reinforcement learning agents, Mach. Learn. Knowl. Extract. 1 (1) (2019) 21–42.



Xinqi Du is currently a Ph.D. student at the School of Artificial Intelligence, Jilin University, China. She received the B.E degree from the Computer Science and Technology, Jilin University, in 2018. Her research interests include multi-agent reinforcement learning, deep reinforcement learning and machine learning. She has served as a peer reviewer for the international journals, including IEEE Internet of Things Journal and IEEE Vehicular Technology Magazine.



Hechang Chen is currently an associate professor of the School of Artificial Intelligence, Jilin University (JLU), China. He received the M.S. and Ph.D. degrees from the College of Computer Science and Technology, Jilin University, in 2014 and 2018, respectively. He was enrolled in the University of Illinois at Chicago (UIC) as a joint training Ph.D. student from November 2015 to December 2016 and enrolled in HongKong Baptist University (HKBU) as a visiting student from July 2017 to January 2018. He has published more than 40 articles in international journals and conferences, including IEEE TPAMI, KBS, TKDD, IJCAI, SIGIR, ICDE, WSDM, etc. His current research interests lie in the areas of machine learning, data mining, complex/social network analysis, deep reinforcement learning, and knowledge graph.



Che Wang received the B.S. degree from the School of Engineering, Northeast Agricultural University, China, in 2020, and he is currently a master degree student in the School of Artificial Intelligence, Jilin University, China. His research interests include deep reinforcement learning, agent exploration and the control system design for robot. He has served as a peer reviewer for the Conference on Information and Knowledge Management.

Yongheng Xing is currently pursuing the Ph.D. degree in the College of Computer Science and Technology, Jilin University, China. He received the B.E degree from the Computer Science and Technology, Jilin University, in 2018. His research interests include graph neural network, Internet of Things (IoTs) Data Analytics and machine learning. He has served as a peer reviewer for the international journals, including IEEE TII, IEEE IoTJ and Neurocomputing.

Jielong Yang received the Ph.D. degree in Electrical and Electronic Engineering from Nanyang Technological University, Singapore, in 2020. He is currently an Assistant Professor in the School of Artificial Intelligence at Jilin University, P. R. China. He received the B.Eng. and M.Sc. degree in Mechanical Engineering from Xi'an Jiaotong University in 2012 and 2014, respectively. His research interests include semi-supervised and unsupervised learning with Bayesian networks and Graph neural networks.

Philip S. Yu is a Distinguished Professor in Computer Science at the University of Illinois at Chicago and also holds the Wexler Chair in Information Technology. His research interest is on big data, including data mining, data stream, database and privacy. He has published more than 1680 papers in refereed journals and conferences. He holds or has applied for more than 300 US patents. Dr. Yu is a Fellow of the ACM and the IEEE. He is the Editor-in-Chief of ACM Transactions on Knowledge Discovery from Data. He is on the steering committee of the IEEE Conference on Data Mining and ACM Conference on Information and Knowledge Management and was a member of the IEEE Data Engineering steering committee.

Yi Chang is dean of the School of Artificial Intelligence, Jilin University (JLU). This author became a Senior Member (SM) of IEEE in 2010, a Chinese National Distinguished Professor in 2017, an ACM Distinguished Scientist in 2018. Before joining academia, he was a Technical Vice President at Huawei Research America, in charge of knowledge graph, question answering and vertical search projects. Before that, he was at Yahoo Labs/Research from 2006 to 2016 as a research director, and in charge of search relevance of Yahoo's web search engine and vertical search engines. His research interests include information retrieval, data mining, machine learning, natural language processing, and artificial intelligence. He is the author of two books and more than 100 papers in top conferences or journals, and the associate editor of IEEE TKDE. He won the Best Paper Award on ACM KDD'2016 and ACM WSDM'2016. He served as one of the conference General Chairs for ACM WSDM'2018 and ACM SIGIR'2020.

Lifang He is currently an Assistant Professor in the Department of Computer Science and Engineering at Lehigh University. Prior to joining Lehigh, she did her first postdoc in the Big Data and Social Computing Lab at University of Illinois at Chicago, and her second postdoc in medical school jointly at Cornell University and University of Pennsylvania. Her research interest primarily lies in the areas of machine learning and data mining, particularly with tensor analysis and deep learning for medical data, such as brain network and neuroimaging analysis. She has published more than 100 papers in refereed journals and conferences, such as NeurIPS, ICML, KDD, CVPR, WWW, IJCAI, AAAI, TKDE, etc. She is an IEEE member since 2012 and serves on the Program Committees of a number of international conferences in the areas of machine learning, data mining, bioinformatics, artificial intelligence and signal processing.