This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

BAVDM: Behavioral-Pattern-Aware Value Decomposition for Multi-Agent Reinforcement Learning

Baofu Fang, Mengyuan Fang, Hao Wang, and Xiaohui Yuan, *Senior Member, IEEE*

***Abstract*—Value decomposition is a powerful method for addressing credit assignment problems within the centralized training decentralized execution (CTDE) paradigm. However, existing value decomposition algorithms lack effective mechanisms to attribute individual agents' contributions by modeling their dynamically emergent roles through specific behavioral patterns, thereby limiting the further enhancement of credit assignment and policy optimization. To address this gap, we propose Behavioral-pattern-Aware Value Decomposition for Multi-agent reinforcement learning (BAVDM). BAVDM integrates the roles that agents play under specific behavioral patterns as key information flows within an attention-based mixing network to directly derive their contribution, thereby achieving more efficient credit assignment. Specifically, a role-aware module is designed, which initially comprehends the agent's behavioral patterns from its trajectory embeddings and generates role representations along with corresponding policy parameters. These role-based policy parameters connect the agent's decisions to its contribution to the team, and contrastive loss ensures the effectiveness and stability of the learned role representations. During the credit assignment phase, by combining the agents' individual role representations, local Q-values, and global state information, a value decomposition mixing network based on a multi-head attention mechanism is employed to effectively decompose the global Q-value. Experimental results in StarCraft II and MPE multi-agent cooperative scenarios demonstrate that BAVDM outperforms baseline methods. Specifically, in the challenging 5m_vs_6 m scenario from SMAC, BAVDM achieves nearly an 8% higher win rate than state-of-the-art approaches, showcasing its significant advantage.**

***Index Terms*—Multi-agent, reinforcement learning, mixing network, credit assignment.**

Received 7 August 2025; revised 12 November 2025 and 16 December 2025; accepted 9 April 2026. This work was supported in part by the Natural Science Foundation of Anhui Province under Grant 2308085MF203, in part by the Advanced Materials National Science and Technology Major Project under Grant 2025ZD0610101, and in part by the China University Industry—Academia—Research Innovation Fund under Grant 2024ZY005. *(Corresponding author: Baofu Fang.)*

Baofu Fang, Mengyuan Fang, and Hao Wang are with the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230009, China, and also with the Key Laboratory of Knowledge Engineering with Big Data (Hefei University of Technology), Ministry of Education, Hefei 230009, China (e-mail: fangbf@hfut.edu.cn; fmyabcd1018@163.com; jsjxwangh@hfut.edu.cn).

Xiaohui Yuan is with the Computer Science Department, University of North Texas, Denton, TX 76201 USA (e-mail: Xiaohui.Yuan@unt.edu).

Recommended for acceptance by J. Liu.

Digital Object Identifier 10.1109/TETCI.2026.3697333

I. Introduction

As THE extension of single-agent reinforcement learning, Multi-Agent Reinforcement Learning (MARL) holds significant importance in real-world applications, especially in enabling multiple agents to make effective decisions in cooperative tasks. It is widely applied in areas such as UAV formation flight [1], network optimization [2], [3], [4], intelligent transportation systems [5], [6], [7], robot swarm control [8], and autonomous driving [9]. In many MARL environments, due to scalability and communication constraints, agents often need to make decisions locally based on their own partial observations. In this context, training agents using decentralized policies means that, from the perspective of a specific agent, the other agents are part of the environment, leading to a highly non-stationary environment during exploration. To address this issue, the Centralized Training with Decentralized Execution (CTDE) paradigm [10] has been proposed. Within the CTDE framework, during the centralized training phase, agents can access global information that is unavailable during the policy execution phase to optimize their local policies. During the decentralized execution phase, agents make decisions solely based on their local observations. Credit assignment plays a crucial role in achieving effective cooperation within multi-agent teams under the CTDE framework. It refers to attributing the global reward to individual agents in order to infer each agent's contribution to the joint task. Effective credit assignment methods can greatly facilitate policy optimization among agents in the team, thereby significantly enhancing the efficiency of agent cooperation. In recent years, significant progress has been made in credit assignment within cooperative MARL. Current credit assignment paradigms can be broadly categorized into explicit credit assignment and implicit credit assignment [11]. Explicit credit assignment evaluates the actions of individual agents by calculating the difference between the reward and a certain baseline, thereby guiding the optimization of each agent's policy gradient and ensuring local optimality. COMA [12] provides a centralized critic network to train decentralized actors by evaluating each agent's actions using a counterfactual advantage function. SQDDPG [13] utilizes the Shapley Value [14] from game theory to estimate the approximate marginal contribution of agents sequentially added to the team. Value function decomposition is one of the core methods for implicit credit assignment. Its fundamental idea is illustrated in Fig. 1(a): at each timestep, agents interact with the

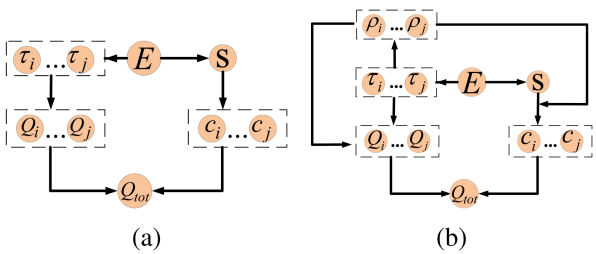
2471-285X © 2026 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE I
SYMBOL DEFINITIONS

Notation	Description
S	State space
$\mathcal{U}$	Joint action set
Ω	Observation space
R	Shared reward function in MDP
r	Team reward
π	A joint policy
s_t	Global state at time t
$P(s'\|s,a)$	State transition function
$r(s,a)$	Shared reward function
$O(s,a)$	Observation function
τ	The joint trajectory of the agent team
Q_{tot}	Joint action value function
Q_i	Individual action value function of agent i
γ	Discount factor
N	The number of agents in the system
$e_{\tau_{i,t}}$	The trajectory embedding of $agent_i$ at timestep t
$o_{i,t}$	Current local observation of agent i at time t
$u_{i,t-1}$	The action of agent$_i$ at time $t-1$
MLP	Multilayer Perceptron
C	The number of clusters
n	The number of agents in BAVDM
E	The external environment
m	The momentum coefficient
c_i	The agent's credit
ρ_i	The role of $agent_i$



(a) (b)

Fig. 1. (a) Basic workflow of value function decomposition; (b) Workflow of BAVDM.

external environment E to generate trajectories τ and the global state s. They then select actions based on their local state-action value functions. Subsequently, the contribution metrics c_i, which quantify each agent's contribution in relation to the global state, are integrated with individual Q-values to formulate the global state-action value function Q_{tot}. Such methods generally require that individual optimal policies are consistent with the global optimal policy (Individual-Global-Max(IGM) principle [15]). VDN [16] and QMIX [17] aggregate local Q-values into a joint Q-value using different approaches: VDN employs simple linear addition, while QMIX integrates them through a monotonicity constraint based on the global state. To maintain IGM consistency and the expressive power of the value function class, QPLEX [18] introduces a duplex dueling architecture based on advantage value decomposition to alleviate suboptimality from monotonicity constraints, while employing a multi-head attention mechanism to generate global Q-values. ResQ [19] addresses limitations in sample efficiency and approximation errors by masking out some ineffective state-action pairs from the joint state-action value function, transforming it into the sum of a main function and a residual function.

Upon reviewing the aforementioned value function decomposition algorithms, it becomes evident that they generally lack effective mechanisms to attribute individual agents' contributions by modeling their dynamically emergent roles through specific behavioral patterns, which in turn constrains the effectiveness of credit assignment and limits further improvements in policy optimization. In the real-world scenarios, team managers, technical experts, resource coordinators, and support personnel respond to and contribute to the same task in distinct ways. Analogously, in multi-agent systems, agents' action-observation histories can reveal their capabilities. These capabilities should not only guide agents to take actions that align with their own behavioral patterns in cooperative tasks but also allow for more precise evaluation of their contributions to the team based on role performance, thereby optimizing cooperative effectiveness.

Building on this, this paper proposes Behavioral-pattern-Aware Value Decomposition for Multi-Agent Reinforcement Learning (BAVDM). As shown in Fig. 1(b), compared to the traditional value decomposition approaches (Fig. 1(a)), BAVDM directly derives each agent's contribution based on the agent's role ρ in conjunction with the global state s. BAVDM understands each agent's behavioral patterns through trajectory embeddings and derives each agent's role representation through an encoder. By integrating the role representations into the value decomposition mixing network and combining them with local Q-values, BAVDM utilizes a multi-head attention mechanism for value decomposition. To enhance the effectiveness and stability of the learned role representations, this paper introduces a contrastive learning framework based on MoCo (Momentum Contrast) to optimize the quality of the role representations. Additionally, in order to ensure that each agent's local Q-value is influenced by its role under its behavioral pattern, BAVDM decodes each agent's role representation into policy parameters that are integrated into the corresponding local utility network's parameters, thereby guiding each agent's decision-making process. Under the CTDE paradigm, BAVDM updates the parameters of the overall framework, meaning that during decentralized execution, the mixing network is removed, and the role representations as well as the local Q-values are generated by the local agent networks.

The main contributions of this paper are summarized as follows:

- From the perspective of agent behavioral patterns, We propose BAVDM to directly derive each agent's global contribution for credit assignment within the value decomposition mixing network based on the agent's role representation. BAVDM encourages agents to select actions that

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

align with their role characteristics to enhance cooperative efficiency.

- We design a Role-Aware Module (RA-Module) and a role representation-based value decomposition mixing network. The RA-Module generates role representations by encoding agents' trajectory embeddings and enhances the compactness and diversity of role representation vectors using a self-supervised MoCo contrastive learning framework. Within the mixing network, a multi-head attention mechanism integrates the role representations into the value decomposition process, achieving more precise credit assignment.
- The proposed method is evaluated on both the StarCraft II micromanagement benchmark and the Multi-Agent Particle Environment (MPE), and the results demonstrate its superior performance.

II. Realted Work

A. Deep Q-Learning for Dec-POMDP

In complex fully cooperative multi-agent tasks, challenges such as partial observability and decentralization are commonly encountered. The Decentralized Partially Observable Markov Decision Process (Dec-POMDP) [20] is a framework designed to effectively address such tasks. In Dec-POMDP, a tuple is typically defined as $\langle N, S, \mathcal{U}, P, \Omega, R, \gamma \rangle$, where N denotes the number of agents in the system, S is the finite set of global states of the environment, $\mathcal{U}$ denotes the joint action set, Ω denotes the set of agents' observations, and $\gamma \in [0, 1]$ is the discount factor. At each timestep, each agent $a_i (i \in \{1, 2, \ldots, N\})$ selects an action $u_i \in \mathcal{U}$ based on its own observation $o_i \in \Omega$. These actions collectively form the joint action $u = u_i|_{i=1}^{N}$, and all agents share the team reward $R(S, u) : S \times u \to R$ generated at that timestep. Meanwhile, the environment transitions to the next state $S' = P(. \mid S, u) : S \times u \times S \to [0, 1]$. Under the condition of partial observability, each agent has a trajectory $\tau_i \in (\Omega \times \mathcal{U})^*$, and learns an individual policy $\pi_i(u_i \mid \tau_i)$ to maximize the team reward. Within this framework, the objective of Dec-POMDP is to find a joint policy $\pi = \{\pi_i, \pi_2 \ldots \pi_N\}$ that maximizes the joint value function $V^{\pi}(S) = E[\sum_t \gamma^t r_t \mid S, \pi]$ to evaluate long-term returns. To achieve this objective, Q-learning is a popular algorithm for learning the optimal policy π^*, which is defined by the following joint action-value function:

$$Q^{\pi^*}(S, u) = r(S, u) + \gamma \mathbb{E}_{S'} \left[\max Q^{\pi^*}(S', u') \right]. \quad (1)$$

With the rapid development of deep learning, the Deep Q-Network (DQN) [21], which parameterizes the Q-value function using neural networks (denoted as θ), has become a foundational framework in recent research on value decomposition algorithms, leading to further improvements and extensions. Algorithms of this type commonly utilize an experience replay buffer D to store state transition samples in the format (τ, u, R, τ'), where τ' denotes the trajectory after the state transition. Due to the limitation of partial observability for agents, τ is generally used as a substitute for the state S. The network parameter θ is updated using the TD loss as follows:

$$L_{TD} = \mathbb{E}_D[r + \gamma V(\tau'; \theta') - Q(\tau, u; \theta)]^2, \quad (2)$$

where $V(\tau'; \theta') = \max_{u'} Q(\tau', u'; \theta')$.

B. Coordination and Roles

Recently, some algorithms have focused on team coordination or roles within the value function decomposition framework to enhance cooperative efficiency. MACC [22] addresses the issue that traditional MARL methods overlook improving team coordination via subtask decomposition, by proposing a task-decomposition-based centralized coordination framework for multi-agent systems. In this framework, subtask representations are first learned from local information, and an attention mechanism is employed to focus on relevant subtasks. This allows agents with the appropriate capabilities to handle specific subtasks more effectively, thereby enhancing their coordination and cooperation abilities in complex tasks. GoMARL [23] introduces a group-oriented automatic grouping mechanism that decomposes the joint action-value function into a combination of group-level action-value functions. By using groups as bridges to connect individual minimal units, GoMARL promotes active cooperation among agents within groups. Assigning roles to agents can, to some extent, improve cooperation among agents within a team. ROMA [24] proposes a method to obtain role embedding vectors from agents' local observations and employs two regularizers to ensure that agents with similar roles can share policies and focus on certain subtasks in cooperative tasks. ACORM [25] presents a framework for role representation learning through maximizing mutual information and derives contrastive learning-based objectives, implicitly guiding agents to learn within a proficient role space for efficient role adaptation. Reviewing these achievements reveals that, although these methods are effective in guiding agents to cooperate efficiently, they lack an effective mechanism to directly derive agents' global contributions based on role-specific behavioral characteristics. This deficiency may lead to ambiguity when assessing the contribution of agents in completing specific subtasks for the team. To address this issue, BAVDM models the behavioral patterns exhibited by agents across different subtasks to learn corresponding role representations, and integrates them with global state information to attribute each agent's contribution to the overall team performance in cooperative tasks.

C. Attention Mechanism

The attention mechanism [26] generates outputs by leveraging queries, keys, and values. It is capable of assigning weights to different parts of the input data, thereby highlighting the information that requires focused attention. Consequently, it has been widely applied in many fields. Specifically, the input to an attention mechanism consists of queries Q (target vectors) and keys K (context vectors $K = \{K_1, K_2 \ldots K_n\}$). A relevance function f is used to calculate the correlation between the target vectors and context vectors, based on which attention weight W_i

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

is assigned:

$$W_i = \frac{\exp(f(Q, K_i))}{\sum_{j=1}^{n} \exp(f(Q, K_j))}. \tag{3}$$

In this paper, the attention mechanism is utilized in the mixing network for value decomposition, with the specific details presented in Section III.

D. Contrastive Learning

Contrastive Learning, as a self-supervised learning paradigm, has achieved remarkable advancements in the domain of computer vision in recent years. It primarily focuses on learning effective feature representations of data. SimCLR [27] enhances model performance under a simple architecture by employing data augmentation and the use of large batches of negative samples. However, because SimCLR heavily relies on large batch sizes, it requires substantial computational resources. MoCo [28], another contrastive learning method, effectively mitigates the challenge posed by small batch sizes by constructing a dynamic dictionary and incorporating momentum updates. Specifically, MoCo transforms data into queries and keys, and uses the keys generated by a momentum encoder to align positive and negative samples, thereby learning meaningful feature representations. Currently, contrastive learning has also shown tremendous potential in the domain of reinforcement learning. CURL [29] is a representative method that leverages contrastive learning to extract high-level features from raw observations, making the observations used in the reinforcement learning component more refined and effective. CURL not only improves the quality of feature representations but also significantly enhances the efficiency and performance of reinforcement learning in complex environments. This method provides a powerful tool for policy generation and policy optimization in reinforcement learning through contrastive learning. By taking advantage of contrastive learning in feature extraction, reinforcement learning models can obtain representations with greater semantic meaning, which is especially important for tasks with complex state spaces. In practical applications, reinforcement learning models combined with contrastive learning exhibit stronger generalization capabilities and adaptability, opening up new possibilities for reinforcement learning in environments rich with unlabeled data.

III. Methods

Table I lists the principal symbols used in this paper. BAVDM is constructed based on a value decomposition framework and primarily consists of two main modules: the Agent Network and the Mixing Network. Fig. 2(a) illustrates the overall structure of BAVDM, with the Mixing Network detailed on the right side. Its main function is to derive the global Q-value by combining the individual Q-values and role representations obtained from the Agent Network with the global state. Fig. 2(b) provides a detailed elaboration of the Agent Network, mainly comprising the agent's individual utility network (indicated by the dashed line on the left side) and the RA-Module (indicated by the dashed line on the right side). The individual utility network is utilized for the agent's decision-making processes and generating local Q-values. The RA-Module consists of an encoder and a decoder: the encoder is used to obtain the agents' role representations and is updated under the control of Contrastive Loss during the centralized training phase, while the decoder is used to decode policy parameters based on the agent's role representation. In this section, the RA-Module, the contrastive loss based on trajectory embeddings that controls role representations, and mixing network with role-oriented multi-head attention mechanism will be introduced separately.

A. Role-Aware Module

As depicted in Fig. 2(b) in the Agent Network, to capture the behavioral features of the $agent_i$, BAVDM employs a GRU (Gated Recurrent Unit) in the utility network to encode and store the agent's trajectory embeddings of $agent_i$. Its local observation at timestep t (denoted as $o_{i,t}$), the action taken by $agent_i$ at the previous timestep $t-1$ (denoted as $u_{i,t-1}$) and its trajectory embedding $e_{\tau_{i,t-1}}$ from the previous timestep are used as input, this process outputs the trajectory embedding $e_{\tau_{i,t}}$ at timestep t as:

$$e_{\tau_{i,t}} = GRU(MLP(o_{i,t}, u_{i,t-1}), e_{\tau_{i,t-1}}), \tag{4}$$

where MLP denotes Multilayer Perceptron, $e_{\tau_{i,t}}$ integrates information from the agent's trajectory by GRU. To learn the role representation corresponding to the agent's behavior patterns, we integrate a role encoder f_{θ_e} in the agent network to map the agent's trajectory embedding into a latent role representation:

$$z_{\rho_{i,t}} = f_{\theta_e}(z_{\rho_{i,t}} \mid e_{\tau_{i,t}}). \tag{5}$$

Where $z_{\rho_{i,t}}$ denotes the role representation of $agent_i$ at timestep t, since $z_{\rho_{i,t}}$ is related to the trajectory information of $agent_i$, it can effectively capture the behavioral features of the agent. Complete parameter sharing, though significantly enhancing learning efficiency, often leads to homogeneous behaviors among agents [30]. To better align the agents' strategies with their behavioral patterns and thereby enhance cooperative efficiency in complex scenarios, we retain partial sharing of the agents' network parameters while employing a decoder g_{θ_d} to assign unique policy parameters to agents with distinct role representations. This approach allows the agents' behaviors to better reflect their role-specific characteristics, resulting in differentiated strategies when selecting actions in complex scenarios. Specifically, the role representation is decoded into latent strategy parameters through the decoder, which influences the weight settings of the second MLP in the individual utility network. In this way, the agent's local Q-value is affected by the role-related strategy parameters and dynamically adjusts its strategy based on the feedback from the state-action value function.

B. Contrastive Loss Based on Trajectory Embedding

Suppose the set of roles in a fully cooperative agent team follows a certain distribution $P(Z)$, and is denoted as Z, where $Z_a \in Z$ describes the role corresponding to a specific behavioral pattern of the agent. Under Z_a, the agent interacts with the external environment to generate the corresponding trajectory embedding e_τ. let $|Z| = C$, and the role encoder f_{θ_e} is employed to transform e_τ into its corresponding role

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

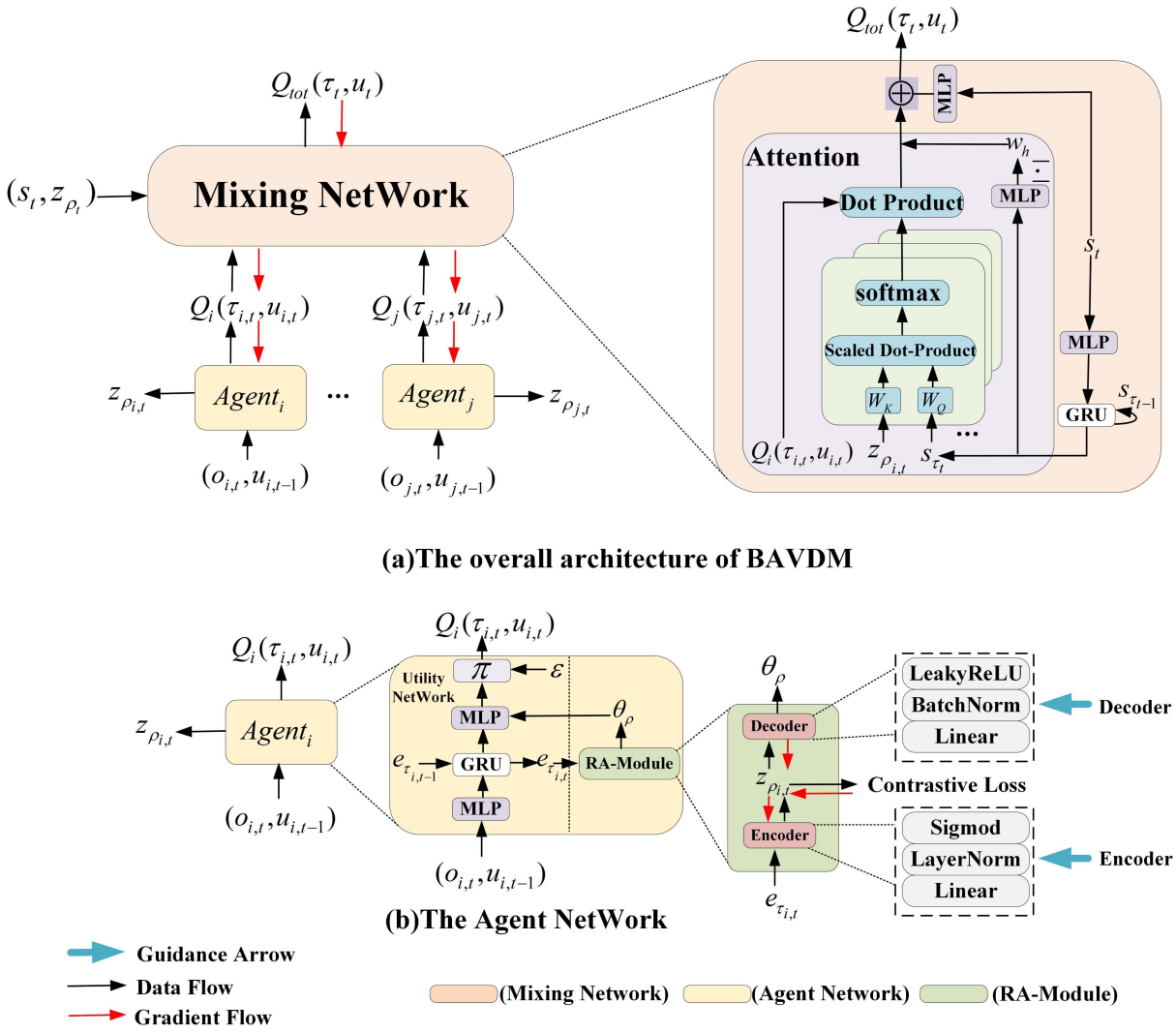


Fig. 2. (a) illustrates the overall architecture of BAVDM. The local Q-values and role representations generated by the Agent Network are fed into the Mixing Network, which incorporates a multi-head attention mechanism, to produce Q_{tot}. Here, z_{ρ_t} denotes $[z_{\rho_{1,t}}, z_{\rho_{2,t}}, \ldots, z_{\rho_{n,t}}]$. (b) depicts the agent network. The left side of the dashed line constitutes the individual utility network, which comprises two MLPs and a GRU. This network outputs the individual action value $Q_{i,t}$ and determines the current action $u_{i,t}$ by $\epsilon - greedy$ strategy. On the right side, the network integrates a Role-Aware (RA) Module. Within this module, the trajectory embedding $e_{\tau_{i,t}}$ is processed by an encoder to generate the role representation $z_{\rho_{i,t}}$. The decoder g_{θ_d} then generates the policy parameters θ_ρ, which influences the weight generation of the second MLP in the utility network.

representation z_ρ. We aim for the mutual information $I(z_\rho; Z_a)$ between the learned role representation and the corresponding role to achieve the maximum value, thereby ensuring that the learned role representation fully captures the behavioral pattern of the agent. Obviously, the mutual information between e_τ and Z_a is difficult to estimate directly because a set of prior role distributions has not been predefined. Fortunately, considering that the agent's role representation is generated based on the agent's trajectory embedding, and the trajectory embedding e_τ is produced by a specific role Z_a within the role set Z that exhibits particular behavioral pattern, we have the following derivation:

$$I(z_\rho; Z_a) = \sum_{z_\rho, Z_a} p(z_\rho, Z_a) \log \frac{p(Z_a|z_\rho)}{p(Z_a)}$$

$$= \mathbb{E}_{z_\rho, Z_a}\left[\log \frac{p(z_\rho|Z_a)}{p(z_\rho)}\right]$$

$$= \mathbb{E}_{z_\rho, Z_a} \log\left[\int_{e_\tau} \frac{p(e_\tau|Z_a)p(z_\rho|e_\tau)}{p(z_\rho)}\, de_\tau\right]$$

$$= \mathbb{E}_{e_\tau, z_\rho, Z_a}\left[\log \frac{p(z_\rho|e_\tau)}{p(z_\rho)}\right]. \quad (6)$$

We derive the relationship between trajectory embedding and role representation to calculate $I(z_\rho; Z_a)$ in (6). Inspired by [25] and [31], we can directly use InfoNCE [31] from contrastive learning to obtain a tight lower bound for the maximum mutual information (Proof in Appendix Theorem 2):

$$I(z_\rho; Z_a) = \mathbb{E}_{e_\tau, z_\rho, Z_a}\left[\log \frac{p(z_\rho|e_\tau)}{p(z_\rho)}\right] \geq \log(C) - L_{\text{CL}}, \quad (7)$$

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

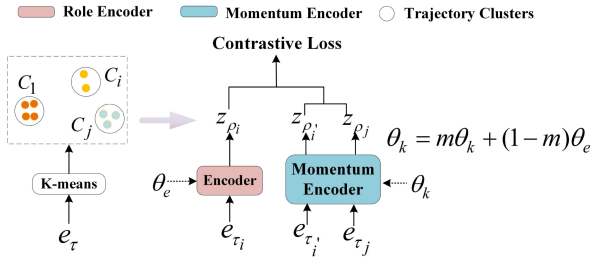


Fig. 3. The contrastive loss is defined over trajectory embeddings, with the role encoder θ_e serving as the main encoder and the momentum encoder θ_k providing auxiliary updates. The left side illustrates the clustering partition for e_τ, while the right side illustrates the computation process of the contrastive loss.

where L_{CL} is InfoNCE, and its general expression is:

$$L_{\mathrm{CL}} = -\log \frac{\exp\left(q^T w k_+\right)}{\exp\left(q^T w k_+\right) + \exp\left(q^T w k_-\right)}. \tag{8}$$

To calculate the similarity between the anchor (q) and keys (where k_+ and k_- together form the keys k), a bilinear product [29] $q^T w k$ is used here for modeling. The positive samples k_+ and negative samples k_- are distinguished based on the agents' trajectory embeddings e_τ. As shown in Fig. 3, to effectively differentiate k_+ and k_-, during the centralized training phase, we apply K-means clustering on e_τ every c timesteps, dividing them into C clusters C_k. This preliminary processing ensures that trajectory embeddings with similar behavioral patterns are positioned close to each other, while those with significantly different behavioral patterns are placed farther apart. For cluster C_i, the role representation z_{ρ_i} obtained from the trajectory embedding e_{τ_i} is used as the anchor q. The role representation $z_{\rho_{i'}}$ obtained from other trajectory embeddings in C_i serves as k_+, and the role representation z_{ρ_j} obtained from trajectory embeddings in cluster $C_j (j \neq i)$ is used as k_-. Based on the similarity between these keys and the anchor, we calculate InfoNCE. To ensure that the learned role representations are more stable, this paper adopts the contrastive learning loss framework from MoCo [28] to update the encoder's parameters. The momentum encoder and the main (role) encoder are updated as:

$$\theta_k = m\theta_k + (1-m)\theta_e. \tag{9}$$

During the training process, we use a relatively large momentum coefficient $m = 0.99$ to ensure that the updates of the momentum encoder are smoother and more stable. In the iterative process, only the main encoder θ_e undergoes gradient descent, while the updates of the momentum encoder θ_k depend on θ_e. The corresponding optimization objective is:

$$\begin{aligned} L_{CL} &= -\mathbb{E}\left[\log \frac{\exp(q^T w k_+)}{\exp(q^T w k_+) + \sum_{i=0}^{K-1} \exp(q^T w k_i)}\right] \\ &= -\mathbb{E}\left[\log \frac{\exp(q^T w k_+)}{\exp(q^T w k_+) + \exp(q^T w k_-)}\right] \\ &= -\log\left(\frac{\sum_{i' \in C_i} \exp\left(z_{\rho_i}^\top w z_{\rho_{i'}}\right)}{\sum_{i' \in C_i} \exp\left(z_{\rho_i}^\top w z_{\rho_{i'}}\right) + \sum_{j \notin C_i} \exp\left(z_{\rho_i}^\top w z_{\rho_j}\right)}\right). \end{aligned} \tag{10}$$

C. Mixing Network With Role-Oriented Multi-Head Attention Mechanism

As shown in Fig. 2(a), within the Mixing Network, BAVDM combines the perceived role representations $z_{\rho_{i,t}}$, individual action values $Q_{i,t}$, and the global state s_t to perform value function decomposition using a multi-head attention mechanism. Based on the implicit function theorem, we can naturally derive the general relationship between the global value function Q_{tot} and the local value functions Q_i under the value function decomposition paradigm as follows:

$$Q_{\mathrm{tot}} = Q_{\mathrm{tot}}\left(s, Q_1(\tau_1, u_1), Q_2(\tau_2, u_2), \ldots, Q_n(\tau_n, u_n)\right), \tag{11}$$

where s denotes the global state, n denotes the number of agents. This paper assumes that in a fully cooperative multi-agent reinforcement learning environment, there are no system-independent agents, and each agent affects the system. Specifically, the partial derivative of the global state-action value Q_{tot} with respect to the individual state-action value Q_i is non-zero. Given that $agent_i$ selects action $u = u_{i,0}$, the partial derivative $\frac{\partial Q_{tot}}{\partial u_i}$ is equal to zero. Consequently, $\frac{\partial Q_i}{\partial u_i}$ evaluated at $u_i = u_{i,0}$ is also zero. By performing a Taylor expansion of Q_i around $u_{i,0}$, we can see that:

$$Q_i(u_i) = c_{i,1} + c_{i,2}(u_i - u_{i,0})^2 + o((u_i - u_{i,0})^2)), \tag{12}$$

where $c_{i,1}$, $c_{i,2}$ are constants. When performing a Taylor series expansion and fitting the global state-action value Q_{tot} with a polynomial of the individual state-action values Q_i, we can neglect the higher-order terms. Specifically, we can analyze one of the nonlinear terms $Q_i Q_j$ to illustrate the rationality of its expansion. We directly construct the product according to the first-order expansions of Q_i and Q_j (Q_j's expansion is similar to that of Q_i). It can be observed that in the product $Q_i Q_j$, the term $c_{i,1}c_{j,1}$ is a constant term, while

$$c_{i,1}c_{j,2}(u_j - u_{j,0})^2 = c_{i,1}(Q_j - c_{j,1}) \tag{13}$$

contains both a linear term $c_{i,1}Q_j$ and a constant term $-c_{i,1}c_{j,1}$. Similarly, $c_{i,2}c_{j,1}(u_i - u_{i,0})^2$ can also be analyzed in the same way. For the higher-order fourth-order term $c_{i,2}(u_i - u_{i,0})^2 c_{j,2}(u_j - u_{j,0})^2$, it can be safely ignored in this approximation. Through such simplified calculations, we observe that the main part of the second-order term $Q_i Q_j$ coincides with the main part of the linear terms Q_i and Q_j. Therefore, we can effectively approximate Q_{tot} using only the individual Q values. In the value decomposition framework based on the multi-head attention mechanism [32], the expression for Q_{tot} in terms of

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Q_i is:

$$Q_{tot} \approx MLP(s_t) + \sum_{h=1}^{H} \sum_{i=1}^{n} \alpha_{i,h,t} Q_i(\tau_i, u_i). \tag{14}$$

Where H denotes the number of attention heads, $\alpha_{i,h,t}$ is constructed from the partial derivatives of Q_{tot} with respect to Q_i, it is a linear functional of the partial derivative $\frac{\partial^h Q_{tot}}{\partial Q_{i,1} \ldots \partial Q_{i,h}}$. Since $\alpha_{i,h,t}$ is a linear combination of higher-order partial derivatives, it decays at a super-exponential rate. This allows the higher-order derivative terms to be ignored when performing a series expansion of Q_{tot}, thereby ensuring the algorithm converges effectively [32]. Detailed proof can be found in the Appendix derivation of Theorem 1. We aim, during the value decomposition process, to directly derive the contribution value of an agent to the team based on the perceived agent's role representation combined with the global state. To achieve this goal, the attention module learns the key value by mapping the agent's role representation $z_{\rho_{i,t}}$ at timestep t through a learnable matrix W_K. The historical trajectory s_{τ_t}, obtained from the global state s_t, serves as a critical information flow connecting each individual agent. s_{τ_t} is processed through a learnable matrix W_Q to derive the query value. By applying the softmax function, the attention score $\alpha_{i,h,t}$ for each attention head at timestep t is calculated as:

$$\alpha_{i,h,t} = \frac{\exp\left(\frac{\lambda (W_Q s_{\tau_t})^T W_K z_{\rho_{i,t}}}{\sqrt{d_k}}\right)}{\sum_{j=1,h}^{n} \exp\left(\frac{\lambda (W_Q s_{\tau_t})^T W_K z_{\rho_{j,t}}}{\sqrt{d_k}}\right)}. \tag{15}$$

Where λ is the temperature coefficient, which is set to 1 by default. To alleviate the constraints on weight boundaries caused by the self-attention mechanism and to enhance non-linearity for improving the adaptability of value decomposition, we input s_{τ_t} into a two-layer MLP to assign corresponding head weights w_h to different heads in the mixing network. Finally, the relationship between Q_{tot} and Q_i can be expressed as:

$$Q_{tot,t} \approx MLP(s_t) + \sum_{h=1}^{H} w_h(s_{\tau_t}) \sum_{i=1}^{n} \alpha_{i,h,t} Q_{i,t}. \tag{16}$$

Obviously, it can be easily deduced that:

$$\frac{\partial Q_{tot,t}}{\partial Q_{i,t}} = \sum_{h=1}^{H} w_h(s_{\tau_t}) \alpha_{i,h,t} > 0, \tag{17}$$

where $w_h(s_{\tau_t}) > 0$, as we take its absolute value. From (17), it can be concluded based on the monotonicity principle of value function decomposition that BAVDM complies with the IGM principle.

D. Algorithm Training

The overall optimization objective of BAVDM consists of two parts, namely the contrastive loss L_{CL} and the TD loss L_{TD}. Specifically, L_{CL} is optimized once every fixed time interval, and at each timestep, all parameters in the BAVDM framework are updated by minimizing the TD loss according to the Bellman equation:

$$L_{TD}(\theta) = \mathbb{E}_D \left[\left(y' - Q_{\text{tot}}(s, u; \theta) \right)^2 \right],$$
$$y' = r + \gamma \max_{u'} Q'_{\text{tot}}(s', u'; \theta'). \tag{18}$$

In the equation above, $\theta = \{\theta_m, \theta_e, \theta_d\}$, where θ_m represents all parameters of the mixing network, and θ_d represents the parameters of the decoder g_{θ_d}. D is the replay buffer (which is a collection storing experience tuples), and θ' represents the parameters of the target network. The implementation process of BAVDM can be found in Algorithm 1.

Algorithm 1: Behavioral-pattern-Aware Value Decomposition.

Initialize:
Mixing Networkθ_m, target Mixing Networkθ'_m, agent's Q-value Networks, Role Encoderθ_e, Role Decoderθ_d, and Replay Buffer $\mathcal{D}$.
for episode $= 1$ to M **do**
  $t = 0$, $s_0 =$ initial state, initialize observation $o_{i,0}$, action sets $u_{i,0}$ and $e_{\tau_{i,0}}$ for $i \in n$
  while s is not terminal and $t < T$ **do**
    $t = t + 1$
    for each agent $agent_i$ **do**
      obtain the trajectory embedding by (4)
      obtain the Role embedding by (5)
      obtain the Role parameter θ_ρ by Role Decoder
      obtain $Q_{i,t}$ by Q-value network
      sample $u_{i,t}$ from $\pi_i(\mathbf{Q}_{i,t}, \epsilon)$
    end for
    Execute actions $\mathbf{u}_t = (\mathbf{u}_{1,t}, \ldots, \mathbf{u}_{n,t})$
    Receive reward r_{t+1} and next state $\mathbf{s}_{t+1}$
  end while
  Store episodes in Replay Buffer $\mathcal{D}$
  Sample a random mini-batch of episodes from $\mathcal{D}$
  if episodes$\%c == 0$ **then**
    Partition the agent's trajectory embeddings into C clusters
    update parameters by (9), (10)
  end if
  obtain the $Q_{tot,t}$ by (16)
  update the parameter θ by (18)
end for

IV. Experiments

In the experimental section, this paper covers the following aspects: Section IV-A introduces the experimental setup, Section IV-B presents and analyzes the comparative experimental results of BAVDM against some of the most advanced MARL methods within SMAC [33] and MPE benchmark. To gain a deeper understanding of how agents' behavioral pattern learning in BAVDM enhances value decomposition in cooperative tasks, Section IV-C utilizes the specific 1c3s5z combat scenario on the SMAC environment to provide a detailed analysis of the

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

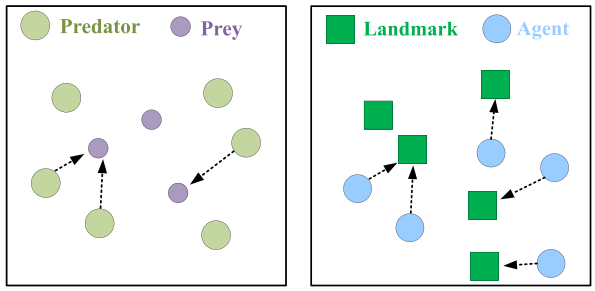


Fig. 4. State visualization of benchmark environments in MPE, (a) is Predator-Prey, (b) is Cooperative Navigation.

distribution of agents' attention scores and their roles' evolution at three different time steps within an episode. Finally, Section IV-D verifies the impact of each component of BAVDM, as well as different numbers of clusters, on overall performance through ablation experiments.

A. Experimental Setup

In this paper, we evaluate BAVDM on both the SMAC micromanagement tasks and the multi-agent particle environment (MPE). Fig. 4 presents the two particle environments chosen for the experiments, namely Predator-Prey and Cooperative Navigation. In the predator-prey task, L predators (large circles) must cooperate to capture T faster-moving prey (small circles) without collisions, as illustrated in Fig. 4(a). The prey follow a predefined strategy that selects actions maximizing their distance from the nearest predator. Both predators and prey have five available actions: moving up, down, left, right, or staying still. In our experiments, we select two scenarios: 7 predators hunting 4 prey and 6 predators hunting 3 prey, denoted as 7v4 and 6v3, respectively. In the cooperative navigation task, L agents need to cooperate to occupy L landmarks while avoiding collisions, as shown in Fig. 4(b). Each agent has 5 possible actions: moving up, down, left, right, or choosing to stay still. At the start of each episode, the environment is initialized by randomly placing L agents (circles) and L landmarks (squares). Agents must coordinate in the task scene to occupy as many landmark positions as possible. Whenever any agent occupies a landmark, the entire team receives a $6/L$ reward at that timestep. In this task scenario, we select an environment with 5 agents for our experiments, abbreviated as CN_5.

SMAC [33] is a StarCraft II–based reinforcement learning benchmark focused on multi-agent micromanagement tasks with diverse unit types. We follow SMAC's default settings, including very hard difficulty, an observation radius of 9, and an attack range of 6. Each agent observes features of both allies and enemies, such as health, position, shield, map coordinates, and unit types, and selects actions from a discrete set of movement, attack, and stop commands. During combat, an agent receives 10 reward points each time it kills an enemy unit, and once all enemy units are eliminated, the allied team receives an additional joint reward of 200 points. In this paper, we evaluate the performance of BAVDM and the baseline methods on six maps of three different difficulty levels, including two easy scenarios (1c3s5z and 3s5z), two hard scenarios (3s_vs_5z and 5m_vs_6 m), and two super-hard scenarios (MMM2 and 6h_vs_8z).

TABLE II
HYPERPARAMETER SETTINGS

Types	Hyperparameters	Values
Common	buffer D size	5000
	batch size	32
	discount factor γ	0.99
	epsilon start	1.0
	epsilon end	0.05
	learner log interval	5000
	runner log interval	5000
	test interval	10000
	test episode	32
	mixing embedding dimension (QMIX)	32
	hypernet layers (QMIX, MACC, ACORM)	2
	hypernet embedding (QMIX)	64
Unique	learning rate (BAVDM)	0.0005
	momentum coefficient m (BAVDM)	0.99
	contrastive learning rate (BAVDM)	0.0006
	recon_loss_weight (MACC)	0.0001
	similarity_loss_weight (MACC)	0
	adv hypernet embedding (QPLEX)	64
	attention_output dimension (BAVDM)	64
	agent embedding dim e_τ (BAVDM)	128
	role representation z_ρ dim (BAVDM)	64
	headers number H (BAVDM)	4
	query Q embedding layer number (BAVDM)	2
	key K embedding layer number (BAVDM)	1
	Clustering frequency c on SMAC (BAVDM)	100
	Clustering frequency c on MPE (BAVDM)	50

Under the CTDE paradigm, we select several advanced reinforcement learning methods, including QMIX [17], QPLEX [18], QATTEN [32], MACC [22], and ACORM [25], as our baselines and compare them with BAVDM. The first three are typical value decomposition methods, while the latter two focus on addressing multi-agent coordination issues. To ensure fairness, all algorithms are run with 5 independent random seeds. In SMAC, every 10,000 timesteps, we evaluate each algorithm over 32 episodes using a greedy policy, recording the percentage of winning episodes as the test win rate. In MPE, the total team rewards obtained in each episode are used as the primary evaluation metric. The curve represents either the average win rate or the episode rewards, with the 95% confidence interval shown as a shaded region. In the experiments, the three value decomposition algorithms (QMIX, QPLEX, QATTEN) are implemented using the Pymarl [33] framework, and both MACC and ACORM are also open-source. To ensure the fairness of the experiments, we adjust the parameters of each algorithm to ensure that all implementations adhere to the parameter settings specified in their original papers. In BAVDM, the dimension of the role representation vectors generated by the encoder is set to 64. The number of heads in the mixing network is set to 4, with the query matrix W_Q consisting of a 2-layer embedding network and the key matrix W_K being a single-layer network. The number of clusters for trajectory embeddings is set to 3 by default, and set to 4 in scenarios with a larger number of agents, such as 1c3s5z, 3s5z, and MMM2. For clarity and conciseness, Table II summarizes the recommended hyperparameter settings for all

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

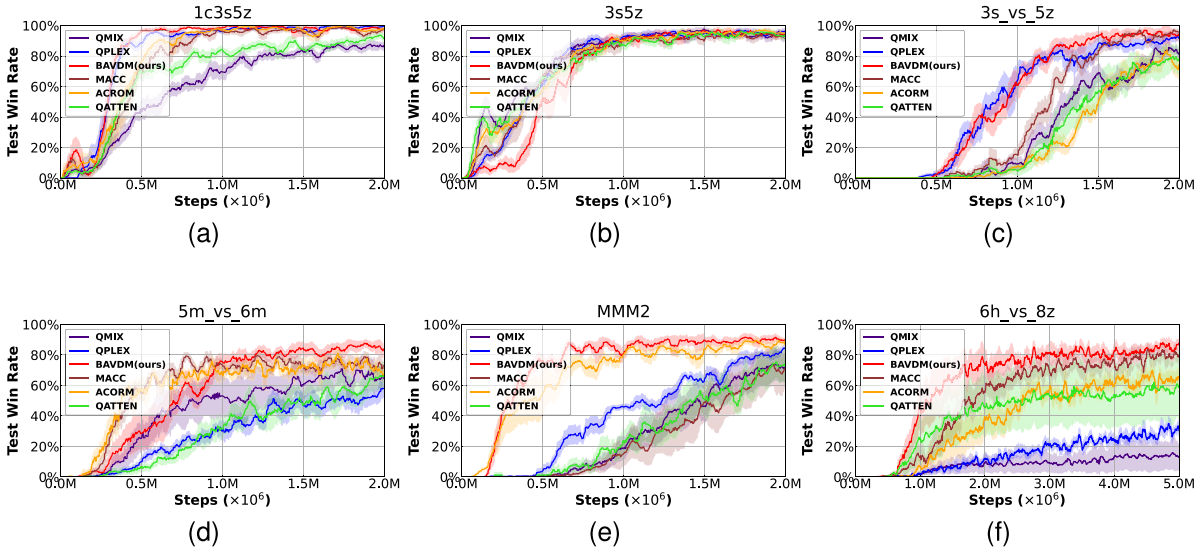


Fig. 5. Comparison of the average win rates of each algorithm across six different map scenarios: (a) 1c3s5z, (b) 3s5z, (c) 3s_vs_5z, (d) 5m_vs_6 m, (e) MMM2, and (f) 6h_vs_8z.

TABLE III
PERFORMANCE COMPARISON OF DIFFERENT METHODS (TEST WIN RATE VS TIME EXPENSE)

Methods	1c3s5z		5m_vs_6m		MMM2		6h_vs_8z	
	Test Win Rate (%)	Time (h)	Test Win Rate (%)	Time (h)	Test Win Rate (%)	Time (h)	Test Win Rate (%)	Time (h)
QMIX	84	5.3	71	5.5	78	6.4	17	10.2
QPLEX	99	8.3	59	7.6	80	8.6	35	14.4
QATTEN	91	7.1	66	7.2	76	8.1	60	11.7
MACC	97	5.7	77	5.4	71	6.2	80	10.8
ACORM	99	6.3	77	8.3	86	6.8	65	11.8
BAVDM (ours)	100	7.5	83	8.5	88	8.7	84	13.9

algorithms. All algorithms in this paper are implemented using PyTorch and trained on an NVIDIA GeForce RTX 3070Ti GPU. The source code for our implementation is publicly available at: https://github.com/FMaureen/BAVDM.git.

B. Experimental Results and Analysis

Fig. 5 presents the average win rates of each algorithm across six different SMAC task scenarios. Additionally, Table III summarizes the performance and runtime results of the algorithms on several representative map scenarios. In the two simple task scenarios, 3s5z and 1c3s5z, all methods are able to handle the tasks effectively, achieving satisfactory performance. Among them, BAVDM demonstrates the best performance. In the 1c3s5z scenario, BAVDM begins to converge at 0.5 M timesteps, while in the 3s5z scenario, it gradually converges around 1 M steps. Additionally, other baseline algorithms are able to achieve convergence relatively quickly. In the more challenging 3s_vs_5z scenario, although BAVDM initially lags behind QPLEX during training, it gradually surpasses QPLEX after approximately 1.2 M timesteps and ultimately achieves performance comparable to MACC. It is noteworthy that in this scenario, the number of agents is relatively small and the role distribution is not prominent. Consequently, the RA-Module constructed in BAVDM fails to fully leverage its capability to be aware of the behavioral patterns of agents. In the 5m_vs_6 m scenario, BAVDM initially underperforms compared to ACORM and MACC during the first 1 M timesteps. However, as training progresses, BAVDM gradually surpasses them, ultimately achieving optimal performance. This improvement may be attributed to the algorithm's continuous learning to effectively integrate roles into the value decomposition network in the early stages. As training advances, agents are able to make increasingly better decisions based on their individual role characteristics. In contrast, within the MMM2 scenario, which involves a larger number of agents and more complex team compositions, BAVDM demonstrates the best performance.

In the 6h_vs_8z map scenario, both the number and types of agents on opposing sides display significant heterogeneity. In such contexts, it is crucial for agents to comprehend their own behavioral patterns and base their decisions on this understanding to achieve victory. Furthermore, due to the considerable complexity of this task, all algorithms in this study are trained for 5 M timesteps. Experimental results indicate that although QMIX, QPLEX, and QATTEN adhere to the IGM principle, which ensures consistency between individual agent strategies and the globally optimal strategy, they fail to incorporate the impact of behavioral patterns on individual decision-making and team coordination within the value decomposition process. This omission may hinder their ability to effectively adjust strategies in complex scenarios, resulting in less than optimal performance. In contrast to ACORM, which integrates roles

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

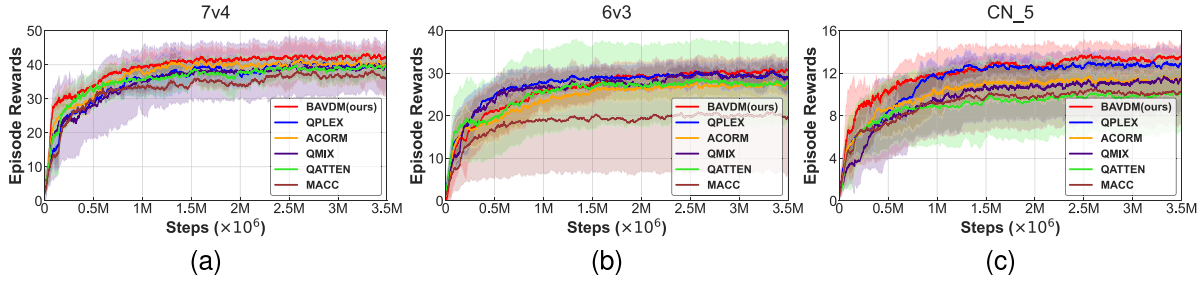


Fig. 6. Performance comparison between the proposed BAVDM and baseline methods on the MPE benchmark: (a) 7 predators vs. 4 prey, (b) 6 predators vs. 3 prey, (c) 5 agents and 5 landmarks.

into the global state, and MACC, which enhances coordination by considering task decomposition from the agents' perspective, BAVDM leverages role representations derived from agents' specific behavioral patterns within a multi-head attention mechanism-based mixing network. This approach directly infers each agent's contribution to the team, thereby facilitating more effective credit assignment. Consequently, BAVDM is capable of managing more intricate tasks and achieves superior performance.

Fig. 6

shows the performance curves of BAVDM compared to other baseline algorithms, with all methods trained for 3.5 M timesteps in the MPE environments. It can be seen that BAVDM achieves strong performance across all three multi-agent particle environments, demonstrating good generalization ability across diverse experimental settings.

C. Attention Analysis

To investigate the behavioral features that BAVDM focuses on at different timesteps during the SMAC cooperation process, we conduct a visualization analysis of the attention weights of different heads for each agent in the 1c3s5z scenario, and present the clustering results of the learned role representations of different agents at the corresponding timesteps. In this scenario, both the enemy and our side possess three types of combat units: Colossus, Stalker, and Zealot. Specifically, Colossus is a large ground unit capable of inflicting substantial damage to ground units through thermal energy rays; Stalker is a ranged ground attack unit equipped with a flexible blink ability, allowing for swift movement and attacks; Zealot is a melee unit that can rapidly charge and assault enemies. To facilitate description, agents of the corresponding categories are abbreviated as c_i, s_i, and z_i. To provide a more intuitive depiction of the cooperation process, as shown in Fig. 7, selected snapshots of battle scenarios are presented, illustrating the changes in attention weight distributions of different agents at these specific time steps.

At time step $t = 2$, Figs. 7(a), (d), (g) show that agents with ranged attacks (c_0, $s_{1:3}$) exhibit higher attention weights, indicating the algorithm's focus on ranged combat. This occurs when team formations are developing and inter-team distances remain large, making ranged attacks optimal. Consequently, melee units like Zealots receive lower attention. Role clustering further groups c_0 and $s_{1:3}$ are grouped into two closely related clusters, while the other agents are assigned to two other, more distant clusters. From Figs. 7(b), (e), and (h), at $t = 11$, the attention weights of c_0 are relatively high and evenly distributed across all heads, while s_1 and s_2 exhibit the highest attention weights in specific heads. At this point, our Colossus is critically wounded (HP ≈ 0) from an enemy Zealot's charge attack. BAVDM prioritizes maximizing damage output through ranged backline attacks, enabling near-death Colossus to secure victory despite being unable to heal. Due to terrain and distance constraints, enemy units cannot evade our counterattacks. At this phase, s_1, s_2, and z_6 prioritize offensive actions over defensive coordination, effectively splitting enemy focus. Specifically, s_1 and s_2 divert enemy focus to the flanks. The attention weight distribution reveals BAVDM's tactical intent: they attract maximal enemy firepower. Meanwhile, z_6 exploits this distraction to execute a melee charge against the enemy Colossus. Meanwhile, the remaining z_4, z_5, z_7 and z_8 use their numerical superiority to quickly encircle and eliminate the enemy units that charge toward them. The dimensionality-reduced visualization of role representations in Fig. 7(h) further supports the validity of this analysis.

Figs. 7(c), (f), (i) show that at $t = 23$, c_0 is eliminated (zero contribution). Its attention weights approach minima (evident from color mapping), with role embeddings clustering separately. Meanwhile, s_1 and s_2, which had absorbed a significant portion of the enemy's attacks in the previous battles, are in a critical state with very low health. At this time step, they focus on evading the enemy Zealot's assault while still engaging in counterattacks, with similar attention weight distributions. Notably, s_3 shows the highest attention weight on $head_1$, indicating that BAVDM is focusing on enabling effective blink attacks while keeping the agent's health close to full. At the same time, z_5, z_6, and z_7 charge the enemy Colossus, while z_4 and z_8 engage enemy Stalkers in melee combat, providing attack coverage for allies. The role representation clustering of these agents is also visualized in Fig. 7(i). This analysis demonstrates that BAVDM's focus on the agents' behavioral patterns evolves dynamically at different time points during combat. This targeted attention helps guide agents to adopt strategies that align with their roles, encouraging active participation in cooperation.

D. Ablation Study

1) Component Ablation: To investigate the impact of each component within BAVDM on its performance, this study conducts ablation experiments on two map scenarios: 3s5z

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

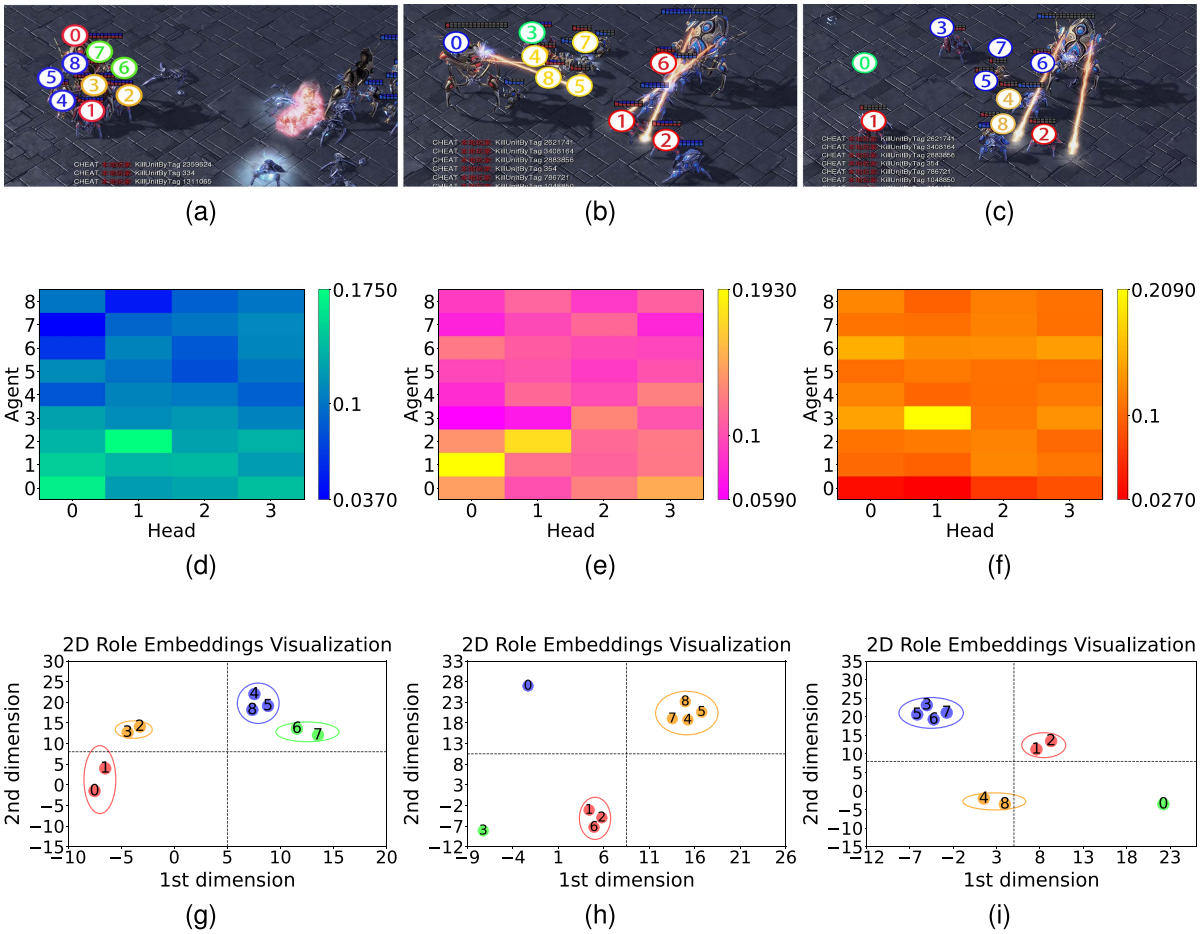


Fig. 7. (a), (b), and (c) are snapshots of the BAVDM-controlled agent team on the 1c3s5z map at time steps t=2, t=11, and t=23, respectively. Correspondingly, (d), (e), and (f) illustrate the corresponding attention weight heatmap visualizations of the agents at these respective time steps. Each heatmap has the horizontal axis representing four distinct heads, the vertical axis corresponding to agent identifiers, and each cell displaying the attention weight value of the respective agent for a specific head. (g), (h), and (i) respectively present the t-SNE visualization of agent role representations at different training timesteps. The high-dimensional role representations are projected into a 2D space using this nonlinear dimensionality reduction technique, where points sharing the same color indicate agent representations clustered into the same role category.

and 5m_vs_6 m. In each experiment, we remove one component while strictly maintaining the integrity of the remaining structure. Specifically: (i) BAVDM_NoAttention: We remove the attention module in the mixing network. The obtained role representations are concatenated with the global state and fed into a QMIX-like mixing network to aggregate local Q-values. (ii) BAVDM_NoDecoder: We remove the decoder used during training to obtain policy parameters that guide the agents, thereby influencing the agents' decision-making process. (iii) BAVDM_NoCL: We remove the contrastive loss L_{CL}, which affects the effectiveness of the agent's learned role representations.

As shown in Figs. 8(a) and 8(b), in the 3s5z simple task scenario, the decoder has a minor impact on performance. Although BAVDM_NoDecoder performs slightly worse than the fully structured BAVDM, it is still sufficient to handle simple tasks where the number of enemy and ally agents is equal. However, BAVDM_NoCL and BAVDM_NoAttention exhibit significant performance gaps compared to the full model. This is because the learned ineffective role representations and the reliance on implicit credit assignment via the Mixing Network weaken the value decomposition effect. Consequently, this may prevent agents from forming effective cooperative formations to attack enemy units during certain time periods. In the challenging 5m_vs_6 m scenario, each component impacts performance to varying degrees, with the attention mechanism module and contrastive loss L_{CL} continuing to have more pronounced effects. This is because, in complex tasks, the effectiveness of agents' role representation learning determines their understanding of their own behavioral patterns, thereby influencing the allocation of attention weights during value decomposition. Some agents that are capable of matching sub-tasks remain in a passive cooperative state due to low weights, which is detrimental to overall efficiency of coordination. The attention mechanism module performs value decomposition based on role representations of behavioral patterns, and removing this module severely degrades performance. The decoder guides agents to make decisions in line with their learned behavioral patterns, and its influence on BAVDM's performance in complex tasks is also significant. Through the above analysis, it is evident that in more complex scenarios, these components are crucial factors in shaping the performance of BAVDM.

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

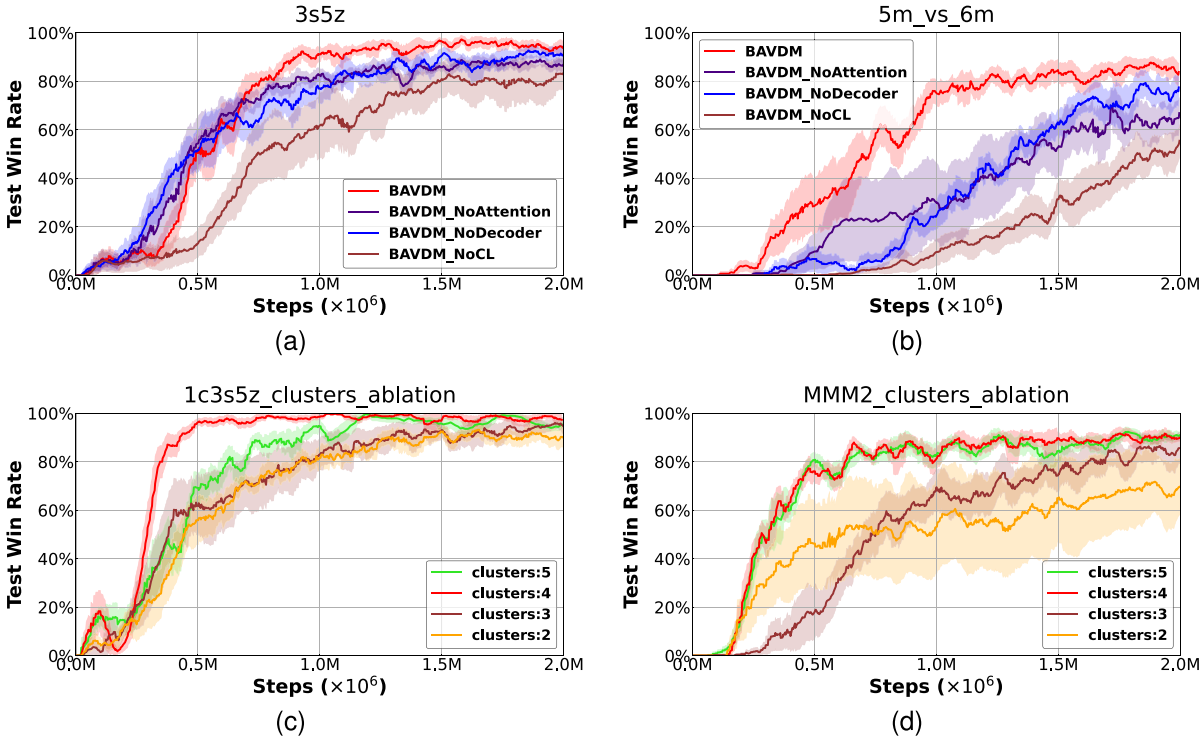


Fig. 8. (a) and (b) respectively illustrate the average win rates of ablation experiments on the key components of BAVDM in the 3s5z and 5m_vs_6 m scenarios. (c) and (d) respectively present the ablation results of BAVDM under different numbers of clusters when using k-means clustering.

2) Ablation on the Number of Clusters: To verify the robustness of the chosen number of clusters in our experiments, we conduct a cluster number ablation study on the 1c3s5z and MMM2 scenarios in SMAC. In these experiments, we compare the performance when the number of clusters for trajectory embeddings was set to 2, 3, 4, and 5. The results shown in Fig. 8(c) and 8(d) indicate that in the simpler 1c3s5z scenario, performance is worst with 2 clusters and achieves the best result with 4 clusters. This suggests that too few clusters lead to insufficient differentiation of behavioral patterns, while increasing beyond 4 clusters does not yield significant improvement, possibly because overly fine-grained clustering reduces the efficiency of policy sharing among agents. In the more complex MMM2 scenario, the negative impact of too few clusters is even more pronounced, but the performance difference between 4 and 5 clusters is small, indicating that the BAVDM model is robust to the choice of cluster number.

V. Conclusion and Future Work

We introduce BAVDM to address the credit assignment problem in multi-agent cooperation. The method learns role representations that demonstrate the behavioral capabilities of agents based on trajectory embeddings and integrates the decoded policy parameters into the individual utility networks of agents, thereby achieving role-based decision-making. Leveraging role representations, local Q-values, and global state, BAVDM employs a multi-head attention mechanism within a mixing network to directly derive the contribution values of agents, achieving the decomposition of the value function. Additionally, BAVDM ensures the adaptability of value decomposition by utilizing the weights generated for each head of multi-head attention from the trajectory of global state. Visualization analysis of the attention weights reveals that incorporating role representations into the value function decomposition network to directly derive agents' contribution values effectively encourages agents to make role-appropriate decisions in cooperative tasks based on their individual capabilities. Experimental results on the SMAC and MPE demonstrate that BAVDM outperforms a variety of baseline algorithms. Particularly in the challenging 5m_vs_6 m scenario of SMAC, BAVDM achieves a nearly 8% higher win rate than the state-of-the-art method, which highlights its superior performance. Nonetheless, it is imperative to acknowledge the inherent limitations of BAVDM. The test environments selected in this study are fully cooperative scenarios, assuming that there are no agents completely independent of the system. However, this assumption may not hold in partially cooperative settings (such as mixed-motive scenarios), where agents need to balance team rewards and their own interests, potentially limiting the applicability of BAVDM. Moreover, in large-scale and more complex multi-agent environments, BAVDM may also face scalability challenges. Future work could explore grouping agents and performing value decomposition within sub-teams to reduce the complexity of inter-agent interactions. In addition, unlike traditional value function decomposition methods that employ complete parameter sharing, BAVDM partially avoids the homogenization of agents' behaviors. Further research should consider how to further explore the individuality of agents within this framework to facilitate more effective cooperation.

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

APPENDIX
GENERATING FUNCTION PROOF

Theorem 1: Let Q_{tot} be an analytic function of individual Q_i with a convergent Taylor series expansion. Then there exist coefficients $\alpha_{i,k}$ such that:

$$Q_{tot} = \text{const} + \sum_{k=1}^{\infty} \sum_{i} \alpha_{i,k} Q_i, \tag{19}$$

where the coefficients decompose as:

$$\alpha_{i,k} = k \sum_{i_1,\ldots,i_{k-1}} \mu_{i_1 \ldots i_{k-1} i} \alpha_{i_1} \cdots \alpha_{i_{k-1}}, \tag{20}$$

with $\mu_{i_1 \ldots i_k} = \frac{1}{k!} \frac{\partial^k Q_{tot}}{\partial Q_{i_1} \cdots \partial Q_{i_k}}$ and the series converges super-exponentially.

Proof of Theorem 1: Step 1: Generating Function Construction

Define the generating function:

$$G(\mathbf{Q}) = Q_{tot} - \text{const} = \sum_{k=1}^{\infty} \frac{1}{k!} \mathbf{Q}^T \nabla^k Q_{tot} \mathbf{Q}, \tag{21}$$

where $\nabla^k Q_{tot}$ is the k-th order derivative tensor.

Step 2: Coefficient Extraction

For each Q_i, extract the linear coefficient via:

$$\alpha_i = \left. \frac{\partial G}{\partial Q_i} \right|_{\mathbf{Q}=\boldsymbol{\alpha}} = \sum_{k=1}^{\infty} \underbrace{\frac{1}{(k-1)!} \sum_{i_1,\ldots,i_{k-1}} \frac{\partial^k Q_{tot}}{\partial Q_{i_1} \cdots \partial Q_{i_{k-1}} \partial Q_i} \alpha_{i_1} \cdots \alpha_{i_{k-1}}}_{\alpha_{i,k}} \tag{22}$$

Step 3: Convergence Analysis

By the Cauchy-Hadamard theorem, if $\|\nabla^k Q_{tot}\| \leq \sigma^k / k!$, then:

$$\limsup_{k \to \infty} |\alpha_{i,k}|^{1/k} \leq c \limsup_{k \to \infty} (k^{1/k}) = \sigma. \tag{23}$$

Step 4: Attention Mechanism Approximation

Truncate at H heads:

$$\alpha_i \approx \sum_{h=1}^{H} \alpha_{i,h}, \quad \alpha_{i,h} = \frac{\exp\left(\frac{\lambda (W_Q s_\tau)^T W_K z_{\rho_i}}{\sqrt{d_k}} \right)}{\sum_{j=1,h}^{n} \exp\left(\frac{\lambda (W_Q s_\tau)^T W_K z_{\rho_j}}{\sqrt{d_k}} \right)}. \tag{24}$$

with error bound $\|\alpha_i - \sum_{h=1}^{H} \alpha_{i,h}\| \leq e^{-\sigma H}$. ■

Theorem 2: Let z_ρ be the role representation encoded from trajectory embeddings e_τ via $f_\theta(e_\tau)$, and Z_a be the role set with cardinality C. The mutual information satisfies:

$$I(z_\rho; Z_a) = \mathbb{E}_{e_\tau, z_\rho, Z_a} \left[\log \frac{p(z_\rho | e_\tau)}{p(z_\rho)} \right] \geq \log(C) - L_{\text{CL}}, \tag{25}$$

where L_{CL} is the contrastive loss with $\mathcal{O}(e^{-\eta C})$ convergence rate.

Proof of Theorem 2: Given the cluster-based contrastive loss defined as:

$$L_{\text{CL}} = -\mathbb{E} \left[\log \frac{\sum_{i' \in C_i} \exp(z_{\rho_i}^\top w z_{\rho_{i'}})}{\sum_{i' \in C_i} \exp(z_{\rho_i}^\top w z_{\rho_{i'}}) + \sum_{j \notin C_i} \exp(z_{\rho_i}^\top w z_{\rho_j})} \right], \tag{26}$$

where C_i is the cluster of z_{ρ_i}, w is the projection matrix, and expectations are over e_τ and cluster assignments.

The optimal critic function is:

$$s^*(z_{\rho_i}, z_{\rho_j}) = \log \frac{p(z_{\rho_j} | Z_a)}{p(z_{\rho_j})}, \tag{27}$$

which achieves the variational bound:

$$I(z_\rho; Z_a) \geq \mathbb{E}\left[s^*(z_{\rho_i}, z_{\rho_i}) \right] - \mathbb{E} \left[\log \sum_{k=1}^{C} \exp(s^*(z_{\rho_i}, z_{\rho_k})) \right]. \tag{28}$$

Substituting (27) into (26) yields:

$$L_{\text{CL}} = -\mathbb{E} \left[\log \frac{\sum_{k \in C_i} \exp(s^*(z_{\rho_i}, z_{\rho_k}))}{\sum_{k=1}^{C} \exp(s^*(z_{\rho_i}, z_{\rho_k}))} \right]. \tag{29}$$

Under uniform clustering ($p(z_{\rho_k}) = 1/C$ and $|C_i| = C_i$):

$$L_{\text{CL}} = -\mathbb{E} \left[\log \frac{\sum_{k \in C_i} \frac{p(z_{\rho_k} | Z_a)}{p(z_{\rho_k})}}{\sum_{k=1}^{C} \frac{p(z_{\rho_k} | Z_a)}{p(z_{\rho_k})}} \right] = -\mathbb{E} \left[\log \frac{\sum_{k \in C_i} p(z_{\rho_k} | Z_a)}{\sum_{k=1}^{C} p(z_{\rho_k} | Z_a)} \right]. \tag{30}$$

As $C \to \infty$ and assuming cluster concentration:

$$\sum_{k \in C_i} p(z_{\rho_k} | Z_a) \approx p(C_i | Z_a) \cdot |C_i|, \qquad \sum_{k=1}^{C} p(z_{\rho_k} | Z_a) \approx 1,$$

yielding:

$$L_{\text{CL}} \geq -\mathbb{E}\left[\log p(C_i | Z_a) \right] - \log |C_i| + \mathcal{O}(e^{-\eta C}). \tag{31}$$

Recognizing that:

$$\mathbb{E}\left[-\log p(C_i | Z_a) \right] = H(C_i | Z_a) \geq 0,$$

and under uniform cluster size ($|C_i| = N/C$ where N is total samples):

$$L_{\text{CL}} \geq -\log(N/C) + \mathcal{O}(e^{-\eta C}) = \log C - \log N + \mathcal{O}(e^{-\eta C}). \tag{32}$$

Combining with the variational bound (28):

$$I(z_\rho; Z_a) \geq \log C - L_{\text{CL}} - \log N + \mathcal{O}(e^{-\eta C}). \tag{33}$$

For large C, $\log N$ is negligible compared to $\log C$, thus:

$$I(z_\rho; Z_a) \geq \log C - L_{\text{CL}} + \mathcal{O}(e^{-\eta C}),$$

which completes the proof of (25). ■

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

REFERENCES

[1] H. Chang, Y. Chen, B. Zhang, and D. Doermann, "Multi-UAV mobile edge computing and path planning platform based on reinforcement learning," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 6, no. 3, pp. 489–498, Jun. 2022.

[2] Y. Gao, W. Wang, and N. Yu, "Consensus multi-agent reinforcement learning for volt-VAR control in power distribution networks," *IEEE Trans. Smart Grid*, vol. 12, no. 4, pp. 3594–3604, Jul. 2021.

[3] L. Liang, H. Ye, and G. Y. Li, "Spectrum sharing in vehicular networks based on multi-agent reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2282–2292, Oct. 2019.

[4] X. Zhou, Z. Ke, and T. Qiu, "Recommendation-driven multi-cell cooperative caching: A multi-agent reinforcement learning approach," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 4764–4776, May 2024.

[5] J. Dong, A. Yassine, A. Armitage, and M. S. Hossain, "Multi-agent reinforcement learning for intelligent V2G integration in future transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 12, pp. 15974–15983, Dec. 2023.

[6] W. Li, L. Zheng, L. Liao, X. Yang, D. Sun, and W. Liu, "A multiline customized bus planning method based on reinforcement learning and spatiotemporal clustering algorithm," *IEEE Trans. Comput. Social Syst.*, vol. 11, no. 3, pp. 3691–3705, Jun. 2024.

[7] T. Wang, A. Hussain, L. Zhang, and C. Zhao, "Collaborative edge computing for social internet of vehicles to alleviate traffic congestion," *IEEE Trans. Comput. Social Syst.*, vol. 9, no. 1, pp. 184–196, Feb. 2022.

[8] M. Hüttenrauch, A. Šošić, and G. Neumann, "Guided deep reinforcement learning for swarm systems," 2017, *arXiv:1709.06011*.

[9] H. Li, Q. Zhang, and D. Zhao, "Deep reinforcement learning-based automatic exploration for navigation in unknown environment," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 6, pp. 2064–2076, Jun. 2020.

[10] J. Foerster, I. A. Assael, N. De Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, vol. 29, pp. 2145–2153.

[11] M. Zhou, Z. Liu, P. Sui, Y. Li, and Y. Y. Chung, "Learning implicit credit assignment for cooperative multi-agent reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, vol. 33, pp. 11853–11864.

[12] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proc. AAAI Conf. Artif. Intell.*, 2018, vol. 32, no. 1, pp. 2974–2982.

[13] J. Wang, Y. Zhang, T.-K. Kim, and Y. Gu, "Shapley Q-value: A local reward approach to solve global reward games," in *Proc. AAAI Conf. Artif. Intell.*, May 2020, vol. 34, pp. 7285–7292.

[14] L. S. Shapley, "A value for n-person games," in *Contributions to the Theory of Games*, vol. 2, pp. 307–317, 1953.

[15] K. Son, D. Kim, W. J. Kang, D. E. Hostallero, and Y. Yi, "QTRAN: Learning to factorize with transformation for cooperative multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 5887–5896.

[16] P. Sunehag et al., "Value-decomposition networks for cooperative multi-agent learning," 2017, *arXiv:1706.05296*.

[17] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, "QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4295–4304.

[18] J. Wang, Z. Ren, T. Liu, Y. Yu, and C. Zhang, "QPLEX: Duplex dueling multi-agent Q-learning," in *Proc. 9th Int. Conf. Learn. Represent.*, May 2021, pp. 11609–11635.

[19] S. Shen et al., "ResQ: A residual q function-based approach for multi-agent reinforcement learning value factorization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, vol. 35, pp. 5471–5483.

[20] F. A. Oliehoek et al., *A Concise Introduction to Decentralized POMDPs*, vol. 1. Cham, Switzerland: Springer, 2016.

[21] V. Mnih, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[22] L. Yuan et al., "Multi-agent concentrative coordination with decentralized task representation," in *Proc. Int. Joint Conf. Artif. Intell.*, 2022, pp. 599–605.

[23] Y. Zang et al., "Automatic grouping for efficient cooperative multi-agent reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2024, vol. 36, pp. 46105–46121.

[24] T. Wang, H. Dong, V. Lesser, and C. Zhang, "ROMA: Multi-agent reinforcement learning with emergent roles," in *Proc. 37th Int. Conf. Mach. Learn.*, Jul. 2020, pp. 9876–9886.

[25] Z. Hu, Z. Zhang, H. Li, C. Chen, H. Ding, and Z. Wang, "Attention-guided contrastive role representations for multi-agent reinforcement learning," in *Proc. 12th Int. Conf. Learn. Representations*, 2024, pp. 23627–23649.

[26] V. Ashish, "Attention is all you need," in *Proc. Adv. Neural Inf. process. Syst.*, 2017, vol. 30, pp. 5998–6008.

[27] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.

[28] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9729–9738.

[29] M. Laskin, A. Srinivas, and P. Abbeel, "CURL: Contrastive unsupervised representations for reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 5639–5650.

[30] C. Li, T. Wang, C. Wu, Q. Zhao, J. Yang, and C. Zhang, "Celebrating diversity in shared multi-agent reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 3991–4002.

[31] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018, *arXiv:1807.03748*.

[32] Y. Yang et al., "Qatten: A general framework for cooperative multiagent reinforcement learning," 2020, *arXiv:2002.03939*.

[33] M. Samvelyan et al., "The starcraft multi-agent challenge," 2019, *arXiv:1902.04043*.

Baofu Fang received the Ph.D. degree in computer application technology from the Harbin Institute of Technology, Harbin, China, in 2013. He joined the Department of Computer Science and Technology, School of Computer Science and Information Engineering, Hefei University of Technology, Hefei, China, in 2000, Associate Professor in 2010, and Master's Supervisor in 2011. His research interests include multirobot/agent system, emotion/self-interest robot, and machine learning. He is the Technology Chair of Anhui Robot Competition, Member of Standing Committee of China Association of Artificial Intelligence (CAAI) YoungCommittee and Member of Standing Committee of the China Association of Artificial Intelligence (CAAI) Robot and Culture Committee.

Mengyuan Fang received the B.Eng. degree in computer science and technology from the Anhui University of Technology, Ma'anshan, China, in 2023. He is currently working toward the M.Eng. degree in artificial intelligence with the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei, China. His research interests include multi-agent systems, reinforcement learning, and multi-agent reinforcement learning.

Hao Wang received the B.Eng. degree from Shanghai Jiao Tong University, Shanghai, China, in 1984, and the M.S. and Ph.D. degrees from the Hefei University of Technology, Hefei, China, in 1989 and 1997, respectively. He is currently a Professor and Doctoral Supervisor with the School of Computer Science and Information Engineering, Hefei University of Technology. His research interests include intelligent computing theory and software, distributed intelligentsystems, complex system theory, and modeling.

Xiaohui Yuan (Senior Member, IEEE) received the B.S. degree in electrical engineering from the Hefei University of Technology, Hefei, China, in 1996, and the Ph.D. degree in computer science from Tulane University, New Orleans, LA, USA, in 2004. He is currently a tenured Associate Professor with the Department of Computer Science and Engineering, University of North Texas, Denton, TX, USA. His research interests include computer vision, machine learning, and artificial intelligence.

Authorized licensed use limited to: University of North Texas. Downloaded on June 17,2026 at 16:55:03 UTC from IEEE Xplore. Restrictions apply.