2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Policy Diversification through Representation Distinguishability Regularization for Multi-Actor Deep Reinforcement Learning

Meng Xu, Xinhong Chen, Shuguang Wang, Guanyi Zhao, and Jianping Wang, *Fellow IEEE*

***Abstract*— Deep reinforcement learning (DRL) has been widely applied to various applications, but improving exploration remains a key challenge. Recently, multi-actor DRL has emerged as a promising approach that enhances exploration by simultaneously deploying multiple actors for learning. Among these methods, actor diversity helps actors discover better policies. However, existing multi-actor DRL methods still lack effective techniques to promote actor diversity, leading to homogeneous, redundant actors and suboptimal policies. To address this, this work proposes a generic solution that can be seamlessly integrated into existing multi-actor DRL methods to promote actor diversity, thereby enabling better policy learning. Specifically, we decompose each actor into a representation module and a decision-making module, where the representation module receives the environment state and outputs a representation vector for the decision module to generate actions. We then compute the difference between each actor's representation vector and those of all other actors as an additional loss, referred to as representation distinguishability regularization, and train the actor alongside its original loss to promote actor diversity. We demonstrate that our method effectively improves the performance of nine state-of-the-art (SOTA) multi-actor DRL methods across eight benchmark tasks, in terms of return.**

I. INTRODUCTION

DRL, by autonomously exploring and learning from the environment to derive effective task policies, has recently found applications in fields such as autonomous driving [1] and robot control [2]. The long-term goal of DRL is to learn a policy close to the theoretical optimal one. Improving exploration ability is crucial, as it helps DRL agents discover more optimal policies, making it a key challenge.

Recently, multi-actor DRL methods, which deploy multiple actors to explore the environment concurrently and dynamically select the best-performing actor to take actions, have significantly enhanced exploration capabilities compared to single-actor DRL methods, making them a promising approach. Current multi-actor DRL methods include double-actor DRL, Evolutionary DRL (EDRL), and policy ensemble methods. Double-actor DRL methods assign one actor to each critic to improve Q-value estimation and exploration [3][4][5][6]. EDRL enhances exploration by evolving a group of actors, as demonstrated in [7][8][9], while policy ensemble methods leverage multiple actors working together to explore the environment, as shown in [10] and [11]. In multi-actor DRL, diverse actors lead to a wider variety of policies, making it more likely to discover better strategies for task completion. In contrast, homogeneous actors lead to redundancy, reducing the likelihood of generating superior policies. However, existing multi-actor DRL methods still lack effective mechanisms to ensure actor diversity, leading to suboptimal policies. For instance, EDRL attempts to distinguish actors by adding random noise, which does not guarantee diversity. Policy ensemble methods rely on manually assigning different styles for each actor or training actors with diverse samples to indirectly promote diversity, which also exists in some EDRL methods, such as [12], [13], [14], but this is inefficient and cannot scale to more actors. As for double-actor DRL methods, they directly lack a mechanism to ensure actor diversity.

Motivated by the above observations, this work aims to propose a generic solution that can be seamlessly integrated into existing multi-actor DRL methods to promote actor diversity, helping them learn better policies. However, directly maximizing the divergence between actors is generally infeasible. For instance, in the context of EDRL, such an approach may compel certain actors to adopt divergent yet suboptimal actions. Since EDRL relies on actors sharing their learning experiences, these suboptimal actors could propagate detrimental knowledge to others [7][8][9], thereby degrading the overall performance of the actor population.

Given that a deep neural network (DNN) can be decomposed into a representation module and a decision-making module, where the representation module extracts key input information and converts it into input representations for the decision module, which then relies on accurate input representations to produce suitable outputs, the representation module plays a dominant role in the DNN [15][16]. Therefore, we decompose the actor into the representation module and the decision module, where the decision module is the final layer of the actor model, thus transforming the difference between actors into the difference between their representation vectors. This approach not only encourages actors to develop diverse understandings of the environment, thus promoting actor diversity, but also retains the flexibility of their decision-making modules. Since representation vectors are typically complex, high-dimensional, and the data distribution is unknown, direct measurement using methods like Kullback-Leibler divergence is not feasible. Thus, we convert the difference between actors' representation vectors into their inner products, which turns the maximization of

This work is supported in part by the Hong Kong Research Grant Council under General Research Fund (GRF) 11219624 and Collaborative Research Fund (CRF) project C1045-23G. Meng Xu, Xinhong Chen, Shuguang Wang, Guanyi Zhao, and Jianping Wang are affiliated with the Department of Computer Science at City University of Hong Kong and the City University of Hong Kong Matter Science Research Institute (Futian), China. (Email: mxu247-c, guanyzhao3-c, sgwang6-c@my.cityu.edu.hk, xinhong.chen@cityu.edu.hk, jianwang@cityu.edu.hk). Corresponding author (Jianping Wang).

the difference between these vectors into the minimization of their inner product. Based on these principles, for each actor, we add the inner product of its representation vector with those of all other actors as an additional loss, which is combined with its original loss to form the final actor loss, thereby promoting actor diversity.

Our contributions are as follows:

- We propose a representation distinguishability regularization method that can be seamlessly integrated into existing multi-actor DRL approaches to promote actor diversity, thereby improving their policy performance. We provide a theoretical analysis to explain the reasons behind the improvement.
- Our method calculates the inner product of each actor's representation vector with those of other actors as an additional loss, which is combined with the actor's original loss. This new loss is minimized during training to encourage actors to develop diverse understandings of the environment, thereby promoting diversity.
- We demonstrate that our method improves the performance of nine SOTA multi-actor DRL methods across eight benchmark tasks, in terms of return. Additionally, hyperparameter sensitivity tests explain the optimality of our current configuration.

II. Preliminaries

This section presents the problem formulation and summarizes the limitations of existing multi-actor DRL methods.

A. Problem formulation

The DRL learning process is framed as a Markov Decision Process (MDP), defined by $\mathscr{M} = \langle \mathscr{S}, \mathscr{A}, r, \mathbb{P}, \gamma \rangle$, where $\mathscr{S}$ is the state space, $\mathscr{A}$ is the action space, r is the reward function, $\mathbb{P}$ describes the state transitions, and γ is the discount factor. At each time step t, the agent observes the state $s_t \in \mathscr{S}$, takes an action $a_t \in \mathscr{A}$ based on its policy $\pi : \mathscr{S} \to \mathscr{A}$, and receives a reward r_t. The goal of DRL is to identify the optimal policy $\pi(s)$ that maximizes the expected cumulative reward $J(\pi(s))$ over time: $J(\pi(s)) = \max_\pi \mathbb{E}_{\pi(s)} \left[\sum_{i=t}^{T} \gamma^{i-t} r_i \mid s_0, a_0 \right]$. The learning duration is represented by T, with the initial time step $t = 0$. One prominent DRL method is the actor-critic approach, which consists of two main components: the actor $\pi(s)$, responsible for deciding actions a based on the state s, and the critic θ, which parameterizes the Q-function $Q_\pi(s,a;\theta)$. The $Q_\pi(s,a)$ function represents the expected cumulative rewards when following policy π from state s after performing action a, as expressed by $Q_\pi(s,a) = \mathbb{E}_\pi \left[\sum_{i=t}^{T} \gamma^{i-t} r_i \mid s_t = s, a_t = a \right]$.

B. Multi-actor DRL

In multi-actor DRL methods, the agent uses N distinct actors $\pi = (\pi_1, \pi_2, \ldots, \pi_N)$ to generate different actions $a_t^1, a_t^2, \ldots, a_t^N$ from the same state s_t. Afterward, the most effective actors are chosen to produce actions for the environment. Various multi-actor DRL methods adopt different strategies to select the best actor. For example, prior work [3][5] often involves critics to assess the Q-value of each actor, such as $Q_1(s_t, a_t^1)$, $Q_2(s_t, a_t^2)$, ..., $Q_N(s_t, a_t^N)$, and selects the actor with the highest Q-value to generate the action a_t. The objective of multi-actor DRL methods, similar to single-actor methods, is to identify an optimal policy $\pi(s)$ that maximizes the long-term cumulative reward $J(\pi(s))$.

Representative types of multi-actor DRL include double-actor DRL, EDRL, and policy ensemble methods. Double-actor DRL methods have shown considerable potential in improving Q-value estimation accuracy and exploration by pairing each actor with a distinct critic. This method improves Q-value estimation, as demonstrated by Pan et al.'s softmax-regularized approach [3], with further advancements by Guo et al. [4], Lyu et al. [5], and Li et al. [6]. Additionally, it enhances exploration, as shown by Li et al. [17] and others [18], [19], [20], [21], with practical applications in federated learning [22], energy management [23], and resource allocation [24]. However, these methods lack research focused on promoting actor diversity. EDRL improves exploration by evolving a population of actors. Typical methods include evolutionary algorithm-guided DRL (e.g., CEM-RL [7], PGPS [8], and ERL [9]), and evolutionary algorithm module-embedded DRL (e.g., PPO-CMA [25] and GRAC [26]). These approaches primarily promote diversity through crossover and mutation, where crossover allows actors to exchange partial weights with a certain probability, and mutation introduces random noise into actors' model weights. However, these methods are inherently stochastic and cannot reliably ensure distinctiveness among actors. Policy ensemble methods coordinate multiple actors to simultaneously explore the environment. However, they either manually assign distinct styles to actors [10], a process that is inefficient and lacks scalability, or promote actor diversity indirectly by sampling different samples for each actor [11], a technique also observed in EDRL [12], [13], [14]. This method, however, is not scalable to larger actor populations due to the limited number of diverse samples that can be chosen. While single-actor DRL encourages exploration by distinguishing the current policy from past policies [27], [28], [29] and selectively sampling diverse samples [30], these techniques are not applicable to multi-actor DRL. Additionally, diversity exists in multi-agent DRL, where agents assume different roles for heterogeneous coordination [31][32][33][34], but our research focuses on single-agent tasks, making these methods inapplicable.

Unlike existing approaches, we propose a novel method that calculates and maximizes the representational differences between each actor during training, promoting actor diversity. This approach complements current methods for enhancing diversity in multi-actor DRL.

III. Methodology

This section presents the framework, technical details, and theoretical analysis of our method.

A. The framework of our approach

Fig. 1 illustrates the framework of our approach. Regular DRL methods use a single actor and multiple critics (Fig. 1

(a)), while multi-actor DRL methods employ multiple actors and critics, where each actor is allowed to explore independently. A strategy is then employed to dynamically select the best actor for generating actions to interact with the environment (Fig. 1 (b)). Our method (Fig. 1 (c)) divides the actor $\pi(s)$ into a state representation module $\phi(s)$ and a decision module w, with the final actor represented as $\pi(s) = \phi^\top(s)w$. For each actor, we compute the difference between its representation $\phi(s)$ and those of the other actors, referred to as the representation distinguishability regularization. This, combined with the actor's original loss, forms a new loss function used to train the actor. This approach encourages each actor to distinguish itself from others, thereby creating diverse actors and ultimately enabling the multi-actor DRL to explore more optimal policies.

B. Model Training

This section explains the training procedure for our method. First, we sample a mini-batch, $\mathscr{D}_0$, from the replay buffer $\mathscr{D}$, following the baseline method, as our approach can be integrated with existing multi-actor DRL techniques. This mini-batch is then used to update model components, such as the actor π_i and critic θ^i, sequentially. Let N represent the number of actors in the baseline method. For instance, in the case of double-actor DRL [3], N is set to 2. The update process for each model component is outlined below:

(1) Calculating next Q-value: Let π' denote the target actor, and $\theta^{i'}$ the target critic. To update each critic, we begin by computing the next Q-value from the sampled mini-batch $\mathscr{D}_0$. The next Q-value, $\mathscr{T}Q_{\theta^{i'}}(s_{t+1}, \tilde{a})$, can be calculated differently depending on the specific multi-actor DRL method used. For instance, Pan et al. [3] improve Q-value accuracy by applying a softmax function to the next Q-value. When integrating our method with various multi-actor DRL techniques, it is essential to adhere to their respective procedures for computing the next Q-value. Here, we illustrate the calculation of the next Q-value using the widely adopted double critic approach, as follows: $\hat{Q}(s_t, a_t) \leftarrow r_t + \gamma \min_{i=1,2} \mathscr{T}Q_{\theta^{i'}}(s_{t+1}, \tilde{a})$. Here, $\tilde{a} \leftarrow \pi_{\pi'}(s_{t+1}) + \varepsilon$, where $\varepsilon \sim \mathscr{N}(0, 0.2)$. After calculating the next Q-value, we sequentially update each critic.

(2) Updating critics: Let α denote the learning rate. Initially, we update each critic, such as the i^{th} critic θ^i, using the next Q-value $\hat{Q}(s_t, a_t)$, as follows:

$$\theta^i \leftarrow \theta^i + \alpha \nabla_{\theta^i} \frac{1}{|\mathscr{D}_0|} \sum_{(s_t, a_t) \in \mathscr{D}_0} \left(\hat{Q}(s_t, a_t) - Q_{\theta^i}(s_t, a_t)\right)^2 \quad (1)$$

(3) Updating actors: Once the critics are updated, the next step is to update each actor, such as the i-th actor π_i.

(3.1) Computing the representation distinguishability regularization $\mathscr{L}_i$: For the update of the i-th actor, we first calculate $\mathscr{L}_i$ and incorporate it with the actor's original loss to form the final loss used to train the actor. The loss $\mathscr{L}_i$ helps distinguish the actor from those of other actors, thereby improving the diversity of each actor.

Specifically, let $\pi_i(s_t)$ represent the i-th actor and $\pi_j(s_t)$ represent the j-th actor. Maximizing the difference between these two actors means we aim to:

$$\max \left\{ \frac{1}{|\mathscr{D}_0|} \sum_{s \in \mathscr{D}_0} \left\| \pi_i(s_t) - \pi_j(s_t) \right\|^2 \right\} \quad (2)$$

The DRL policy $\pi(s)$ can be viewed as a combination of two key components: a representation module $\phi(s) \in \mathbb{R}^{N_\phi \times 1}$, which generates a feature vector of dimension N_ϕ and plays a central role in the actor, and a decision module $w \in \mathbb{R}^{N_\phi \times 1}$, which is implemented as the final layer of the actor, typically a fully connected layer (FCL). Consequently, the difference between $\pi_i(s_t)$ and $\pi_j(s_t)$ can be transformed into the difference between their respective representations, as follows:

$$\frac{1}{|\mathscr{D}_0|} \sum_{s \in \mathscr{D}_0} \left\| \pi_i(s_t) - \pi_j(s_t) \right\|^2 = \frac{1}{|\mathscr{D}_0|} \sum_{s \in \mathscr{D}_0} \left\| \phi_i^\top(s_t) - \phi_j^\top(s_t) \right\|^2 \|w\|^2$$

Next, we expand the Euclidean distance $\left\| \phi_i^\top(s_t) - \phi_j^\top(s_t) \right\|^2$ between the two actors:

$$\begin{aligned} &\frac{1}{|\mathscr{D}_0|} \sum_{s \in \mathscr{D}_0} \|\phi_i^\top(s_t) - \phi_j^\top(s_t)\|^2 \\ &= \frac{1}{|\mathscr{D}_0|} \sum_{s \in \mathscr{D}_0} \left(\|\phi_i^\top(s_t)\|^2 + \|\phi_j^\top(s_t)\|^2 - 2\langle \phi_i^\top(s_t), \phi_j^\top(s_t) \rangle \right) \end{aligned} \quad (3)$$

From this equation, we see that the inner product of the representation vectors $\phi_i^\top(s_t)$ and $\phi_j^\top(s_t)$ is directly related to the Euclidean distance $\left\| \pi_i(s_t) - \pi_j(s_t) \right\|^2$. Therefore, minimizing the inner product $\langle \phi_i^\top(s_t), \phi_j^\top(s_t) \rangle$ corresponds to maximizing the distance between these vectors, indicating greater diversity between them, expressed as:

$$\begin{aligned} &\max \left\{ \frac{1}{|\mathscr{D}_0|} \sum_{s \in \mathscr{D}_0} \left\| \pi_i(s_t) - \pi_j(s_t) \right\|^2 \right\} \\ &\iff \min \left\{ \frac{1}{|\mathscr{D}_0|} \sum_{s \in \mathscr{D}_0} \left\langle \phi_i^\top(s_t), \phi_j^\top(s_t) \right\rangle \right\} \end{aligned} \quad (4)$$

Based on this principle, for the i-th actor, we compute the difference between its representation and the representations of all other actors as $\mathscr{L}_i$ to promote distinction from the other actors. This yields the following expression:

$$\mathscr{L}_i = \sum_{k=1}^{N} \sum_{k \neq i} \left(\frac{1}{|\mathscr{D}_0|} \sum_{s \in \mathscr{D}_0} \langle \phi_i(s_t), \phi_k(s_t) \rangle \right) \quad (5)$$

By applying this loss $\mathscr{L}_i$ as a diversity regularization to each actor's training, we encourage each actor to be distinct from the others.

(3.2) Computing the gradient of the actor: After computing the representation distinguishability regularization $\mathscr{L}_i$, we update the i^{th} actor π_i by combining its original loss $\mathscr{L}_o(\pi_i)$ with $\mathscr{L}_i$ as follows:

$$\nabla_{\pi_i(s_t)} \mathscr{L}(\pi_i) = \nabla_{\pi_i(s_t)} \left(\beta \mathscr{L}_i + \mathscr{L}_o(\pi_i) \right) \quad (6)$$

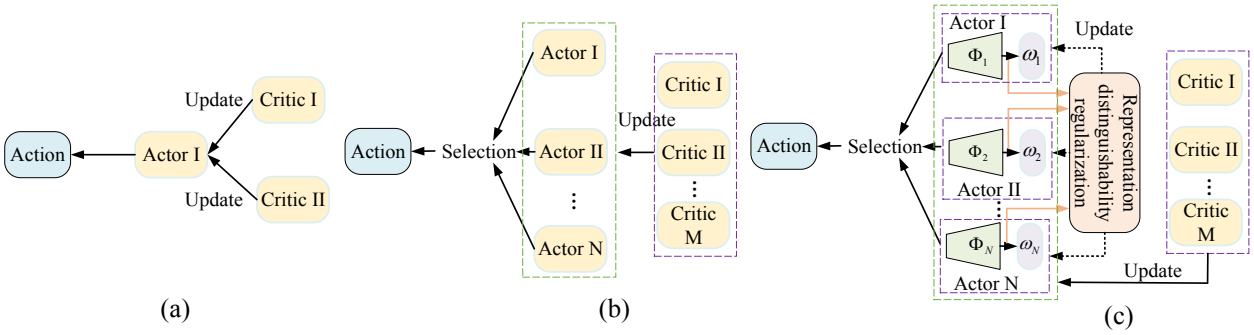


Fig. 1: The framework of our approach: (a) shows regular DRL methods, with the double-critic and single-actor method as an example; (b) illustrates existing multi-actor DRL methods; and (c) demonstrates multi-actor DRL methods integrated with our approach.

Here, β is a hyperparameter used to control the proportion of $\mathscr{L}_i$ in the actor loss, as $\mathscr{L}_o(\pi_i)$ and $\mathscr{L}_i$ may not be on the same scale. The gradient of the original loss $\nabla_{\pi_i(s_t)}\mathscr{L}_o(\pi_i)$ is typically computed in existing multi-actor DRL methods as

$$\nabla_{\pi_i(s_t)}\mathscr{L}_o(\pi_i) = \frac{1}{|\mathscr{D}_0|}\sum_{(s_t,a_t)\in\mathscr{D}_0}\left(\nabla_{\pi_i(s_t)}Q_{\theta^i}(s_t,\pi_i(s_t))\nabla_{\phi_i}\pi_i(s_t)\right) \tag{7}$$

The pseudocode of our method is provided in Algorithm 1.

Algorithm 1 Training procedure of our method

Initialize actors $\pi_1,\pi_2,...,\pi_N$ and critics, along with their target networks $\pi_1',\pi_2',...,\pi_N'$, and replay buffer $\mathscr{D}$
Initialize a random process for action exploration
Initialize the maximum number of training steps M
$t=1, p=1$
repeat
  repeat
    Select action a_t using the current best actor
    Execute the selected action a_t and observe the resulting reward r_t and the new state s_{t+1}
    Store transition (s_t,a_t,r_t,s_{t+1}) in the buffer $\mathscr{D}$
    Compute the next Q-value $\hat{Q}(s_t,a_t)$ using a mini-batch $\mathscr{D}_0$ sampled from the buffer.
    Update critic θ^Q by minimizing the loss: $\frac{1}{|\mathscr{D}_0|}\sum_{(s_t,a_t)\in\mathscr{D}_0}\left(\hat{Q}(s_t,a_t)-Q_\theta(s_t,a_t)\right)^2$
    Compute the representation distinguishability regularization $\mathscr{L}_p$ for actor π_p
    Update actor π_p: $\nabla_{\pi_p(s_t)}\left(\beta\mathscr{L}_p+\mathscr{L}_o(\pi_p)\right)$
    Every d steps, update target networks: $\pi_p'\leftarrow\tau\phi_p+(1-\tau)\pi_p'$
    $p++$
  until $p>N$
  $t++$
until $M-t=0$

C. Theoretical analysis

This section presents a theoretical analysis of our method. We begin by introducing an assumption and three lemmas that serve as the foundation for deriving Theorem 1. Based on Theorem 1, we explain how our approach improves upon existing multi-actor DRL methods.

Assumption 1: In DRL, the reward signal $r(s)$ is bounded such that its magnitude does not exceed a constant $R_{\max}$, which is mathematically expressed as $|r(s)|\le R_{\max}$.

Lemma 1: As discussed in the study [35], the objective function $J(\pi(s))$ in DRL is reformulated as $J(\pi(s)) = \frac{1}{1-\gamma}\mathbb{E}_{s\sim D(\pi(s))}[r(s)]$. Here, the term $\frac{1}{1-\gamma}$ is a constant for all policies, where γ denotes the discount factor.

Lemma 2: Let $D(\pi(s))$ represent the occupancy measure [35] for policy $\pi(s)$. In the study by [35], the expected reward $\mathbb{E}_{s\sim D(\pi(s))}[r(s)]$ is expressed as:

$$\mathbb{E}_{s\sim D(\pi(s))}[r(s)] = \int_{\mathscr{S}} r(s)\,D(\pi(s))(s)\,\mathrm{d}s \tag{8}$$

Lemma 3: In [35], it is proven that for any two actions $\pi_1(s)$ and $\pi_2(s)$ within the action space $\mathscr{A}$, there exists a positive constant K_c such that:

$$\int_{\mathscr{S}}|D(\pi_1(s))-D(\pi_2(s))|\,\mathrm{d}s \le K_c\max_{s\in\mathscr{S}}\|\pi_1(s)-\pi_2(s)\| \tag{9}$$

Theorem 1: The upper bound for the difference between the long-term cumulative rewards $J(\pi^*(s))$ of the optimal theoretical policy $\pi^*(s)$ and the policy $\pi(s)$ learned by multi-actor DRL is given by:

$$\begin{aligned}
&|J(\pi^*(s))-J(\pi(s))|\\
&\overset{(Lemma1)}{=}\frac{1}{1-\gamma}\left|\mathbb{E}_{s\sim D(\pi^*(s))}[r(s)]-\mathbb{E}_{s\sim D(\pi(s))}[r(s)]\right|\\
&\overset{(Lemma2)}{=}\frac{1}{1-\gamma}\left|\int_{\mathscr{S}} r(s)\left(D(\pi^*(s))-D(\pi(s))\right)\mathrm{d}s\right|\\
&\le\frac{1}{1-\gamma}\int_{\mathscr{S}}|r(s)\left(D(\pi^*(s))-D(\pi(s))\right)|\,\mathrm{d}s\\
&\overset{(Assumption1)}{\le}\frac{1}{1-\gamma}\int_{\mathscr{S}}(R_{\max})\left(|D(\pi^*(s))-D(\pi(s))|\right)\mathrm{d}s\\
&\overset{(Lemma3)}{\le}\frac{1}{1-\gamma}R_{\max}K_c\max_{s\in\mathscr{S}}\|\pi^*(s)-\pi(s)\|
\end{aligned} \tag{10}$$

This inequality demonstrates that the difference between the two policies directly affects the disparity in cumulative

reward J. Notably, the term $\frac{R_{\max}K_c}{1-\gamma}$, denoted as C, acts as a positive constant that remains consistent across all policies. Consequently, the performance gap between $J(\pi(s))$ and $J(\pi^*(s))$ is directly influenced by $\max_{s\in\mathscr{S}}\|\pi^*(s)-\pi(s)\|$.

Discussion: Since $\pi(s)$ is chosen from the optimal policy among multiple actors, it is defined as $\pi(s) = \max(\pi_1(s),\pi_2(s),\ldots,\pi_N(s))$. For example, existing approaches [3][5] can generate the current actions by selecting the policy with the highest Q-value from among the multiple actors. Let $\bar{\pi}(s)$ represent the average policy of these actors, and $\Delta\pi(s)$ represent the difference, or diversity, among them. Therefore, $\pi(s) = \max(\pi_1(s),\pi_2(s),\ldots,\pi_N(s)) = \bar{\pi}(s) + \Delta\pi(s)$. Thus, we have:

$$|J(\pi^*(s))-J(\pi(s))| \leq C\max_{s\in\mathscr{S}}\|\pi^*(s)-(\bar{\pi}(s)+\Delta\pi(s))\| \tag{11}$$

The equation above illustrates that as the diversity $\Delta\pi(s)$ among the actors increases, the upper bound $\max_{s\in\mathscr{S}}\|\pi^*(s)-(\bar{\pi}(s)+\Delta\pi(s))\|$ decreases. This reduction in the upper bound of the gap between the learned policy $\pi(s)$ and the optimal theoretical policy $\pi^*(s)$ encourages $\pi(s)$ to potentially converge more effectively towards $\pi^*(s)$. Furthermore, $\Delta\pi(s)$ is directly linked to the differences in the actors' representations, $\Delta\phi(s)$, which can be mathematically expressed as $\Delta\pi(s) \iff \Delta\phi^\top(s)w$. Thus, increasing the diversity in the actors' representations has a beneficial effect on the performance of the final learned policy.

IV. Experiment and Analysis

The experimental environment was configured with Torch (1.2.0), Gym (0.16.0), and Mujoco-py (1.50.0.1). The hardware setup included an Intel Core i7-9700 CPU, 64GB of RAM, and an NVIDIA RTX 2080 GPU. Each method was evaluated across five distinct random seeds.

A. Experiment preparation

(1) Baselines. Nine SOTA multi-actor DRL methods are used as baselines, which include five SOTA double-actor DRL methods: **TD Error-Driven Regularization (TDDR)** [20], **Selective Experience Replay Double-Actor DRL (SERDA)** [36], **Double-Actors Regularized Critics (DARC)** [5], **Generalized-activated Deep Double Deterministic Policy Gradients (GD3)** [37], and **Conservative Advantage Learning (CAL)** [6]; two SOTA ensemble policy methods (5 actors): **Centralized Cooperative Exploration Policy (CCEP)** [10] and **Trajectories-aware Ensemble Exploration (TEEN)** [11]; and two SOTA EDRL methods (10 actors): **Covariance Matrix Adaptation Evolution Strategy (CMAES)** [38] and **Cross-Entropy Regularized Policy Gradient (CEPG)** [39]. Additionally, the Twin Delayed Deep Deterministic Policy Gradient (TD3) and Soft Actor-Critic (SAC) algorithms are included as references.

(2) Benchmarks. Each method is evaluated on four MuJoCo tasks, including HalfCheetah-v2, Hopper-v2, Walker2d-v2, and Humanoid-v2, as well as four navigation tasks [40][41]: 2DNav-v1, 2DNav-v2, Reacher Navigation (ReacherNav), and Ant Navigation (AntNav). The ReacherNav and AntNav tasks are advanced versions of Reacher-v2 and Ant-v2, where the challenge is to navigate a two-jointed robotic arm or an ant-like robot from an initial position to a target. In contrast, the 2DNav-v1 and 2DNav-v2 tasks focus on guiding a point agent toward a goal in a 2D space, with 2DNav-v2 adding complexity by introducing obstacles. For all four navigation tasks, the variation in initial conditions implies differences in the starting position, target location, and obstacle placement. Fig. 2 presents a total of eight validation tasks.

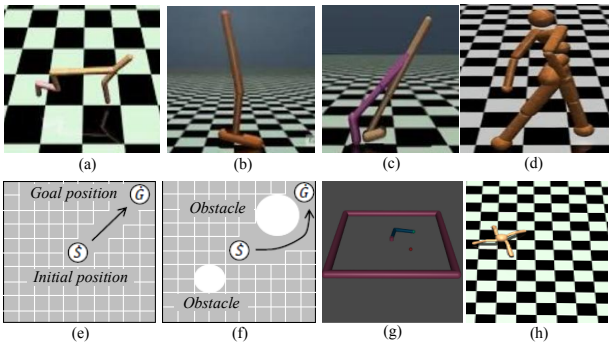


Fig. 2: The experimental setups: (a) HalfCheetah-v2, (b) Hopper-v2, (c) Walker2d-v2, (d) Humanoid-v2, (e) 2DNav-v1, (f) 2DNav-v2, where obstacles are depicted as white circles, with "S" marking the starting position and "G" representing the goal. Additionally, (g) showcases ReacherNav, and (h) presents AntNav.

(3) Metrics. The evaluation is conducted using two performance metrics: **episode return** and **final return**. **Episode return** refers to the average reward accumulated across a predefined number of episodes, typically represented as a curve that evolves over time. On the other hand, **final return** captures the average reward achieved during the final 100,000 steps of the 1 million-step training process. Higher values in both metrics indicate superior performance.

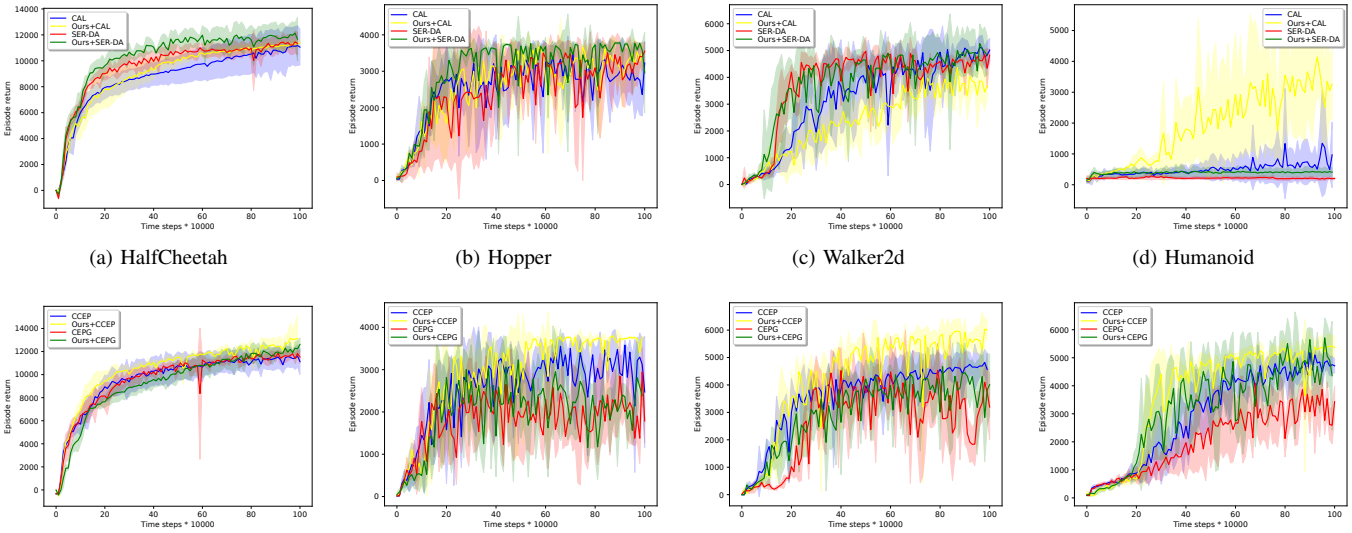
(4) Implementation Details: First, the implementation details of the underlying DRL algorithms are as follows: the DRL algorithms are structured using a three-layer multilayer perceptron (MLP), which consists of two hidden layers and one output layer. Both the actor and critic networks are designed with 256 neurons in each hidden layer, where the hidden layers use the ReLU activation function, while the output layer of the actor network adopts the Tanh activation function. For training, both networks use a fixed learning rate of 0.001 and are optimized via the Adam optimizer. A discount factor of 0.99 is applied, and the target networks are updated every two steps with a soft update rate of 0.005. During replay, mini-batches of 256 randomly selected samples are used, and the replay buffer has a capacity of 1,000,000 samples. For our method, the hyperparameter β is set to 0.001 based on sensitivity testing.

B. Verification under MuJoCo task

This section integrates our method with six baseline approaches: **CCEP**, **CEPG**, **SER-DA**, **DARC**, **GD3**, and **CAL**, evaluating them on four standard MuJoCo tasks. The results in Table I present the final return, with the "Mean"

TABLE I: Performance comparison under four MuJoCo tasks in terms of final return (Mean ± Standard deviation). The bolded results indicate that our method achieves a higher average return (Mean) than the baseline.

Methods	HalfCheetah	Hopper	Walker2d	Humanoid
DARC	11398.44 ± 346.19	3430.68 ± 304.6	4445.27 ± 582.1	612.89 ± 108.49
Ours+DARC	**11896.88** ± 269.44	**3522.07** ± 221.99	**4636.58** ± 395.24	**3402.25** ± 969.43
GD3	10156.11 ± 949.78	3393.59 ± 380.03	4587.67 ± 382.88	346.39 ± 284.42
Ours+GD3	**11394.33** ± 456.13	**3484.58** ± 219.73	4136.41 ± 306.29	**433.03** ± 147.55
CAL	10917.53 ± 1572.43	2851.58 ± 765.56	4907.03 ± 412.98	810.99 ± 821.01
Ours+CAL	**11215.66** ± 317.74	**3272.85** ± 460.72	3689.61 ± 728.23	**3316.52** ± 1355.31
SER-DA	11330.64 ± 687.75	3127.99 ± 609.58	4603.79 ± 238.99	201.38 ± 46.74
Ours+SER-DA	**11849.42** ± 1033.83	**3617.81** ± 235.91	**4899.76** ± 743.25	**415.23** ± 140.13
CEPG	11637.46 ± 388.98	2028.0 ± 574.72	3803.4 ± 1125.64	3210.62 ± 688.42
Ours+CEPG	**12059.32** ± 443.71	**2257.98** ± 464.89	**4390.41** ± 627.37	**4692.42** ± 1064.0
CCEP	11402.93 ± 883.11	3034.51 ± 576.76	4681.86 ± 491.65	4779.23 ± 348.26
Ours+CCEP	**12754.5** ± 1060.83	**3708.88** ± 46.6	**5652.91** ± 514.19	**5300.92** ± 360.19
TD3	10245.74 ± 1144.04	3305.19 ± 505.79	3913.44 ± 359.64	139.79 ± 108.6
SAC	11559.44 ± 506.73	2476.97 ± 710.49	4513.25 ± 545.31	4568.88 ± 304.43



(a) HalfCheetah (b) Hopper (c) Walker2d (d) Humanoid

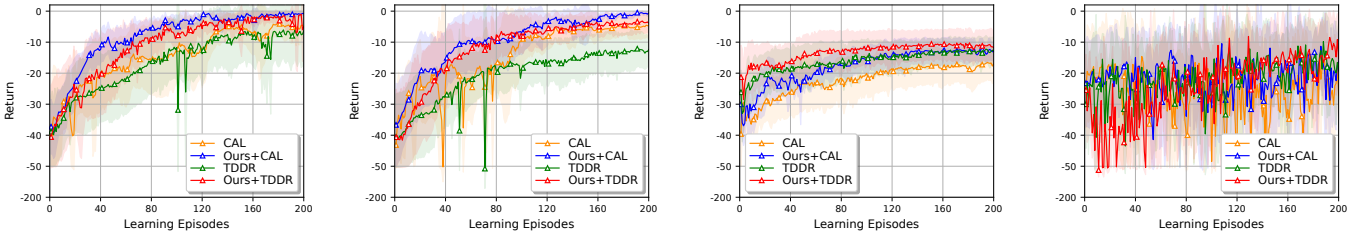
(e) HalfCheetah (f) Hopper (g) Walker2d (h) Humanoid

Fig. 3: Episode return on MuJoCo tasks. The x-axis denotes training steps, and the y-axis represents return. The thick curve indicates the average performance, with the shaded area reflecting the standard deviation across five random seeds.

representing the average return from five runs with different random seeds over the last 100,000 training steps, and the "Standard Deviation" showing the variation across these runs. Other tables follow the same presentation format. Additionally, Fig. 3 presents the episode return after integrating our method with CAL, SER-DA, CCEP, and CEPG. Each evaluation involves 1,000,000 training steps $T_{\max}$, with the first 10,000 steps reserved for random exploration and 10 test episodes conducted every 10,000 steps. The average return from these episodes generates the episode return curve. The results demonstrate that incorporating our method improves the performance (higher returns) of the baseline approaches in most cases, highlighting its effectiveness and generality. Existing multi-actor DRL methods either lack policy diversification (e.g., double-actor DRL methods), rely on random policy diversification (e.g., CEPG), or use manual indirect diversification methods (e.g., CCEP). Our approach fosters actor diversity by maximizing the representation differences among actors during training, which enables the optimal individuals within the actor population to more closely approximate the theoretical optimal policy. This makes our method a valuable complement to existing approaches, significantly enhancing their performance.

C. Verification under Navigation task

This section integrates our method with six baseline approaches: **TEEN**, **CMAES**, **CAL**, **DARC**, **GD3**, and **TDDR**, and evaluates them on four navigation tasks, focusing on the final return. The results in Table II show that our method significantly enhances the performance of the six baseline approaches on these more challenging navigation tasks, further confirming its effectiveness and generality. Fig. 4 illustrates the episode return after integrating our method with CAL and TDDR. Complex navigation tasks complicate the learning of effective policies. Similarly, existing multi-actor DRL methods either lack policy diversification (e.g., DARC, GD3, CAL, TDDR), rely on random policy diversification (e.g., CMAES), or attempt to indirectly promote actor diversity by using different samples for each actor (e.g., TEEN), but none effectively achieve actor diversity. In contrast, by maximizing the representation differences among actors, our approach promotes actor diversity, which aids DRL in exploring more optimal solutions, leading to improved performance.



(a) 2DNav-v1 (b) 2DNav-v2 (c) ReacherPos (d) AntNav

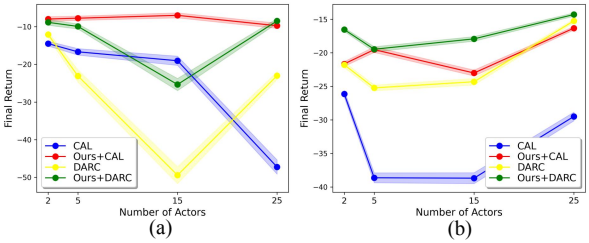
Fig. 4: Episode return on navigation tasks. This figure adopts the same presentation format as the previous figure.

TABLE II: Performance comparison under four navigation tasks in terms of final return (Mean ± Standard deviation).

Methods	2DNav-v1	2DNav-v2
TDDR	-16.542 ± 0.633	-20.731 ± 0.596
Ours+TDDR	$\mathbf{-10.957} \pm 0.697$	$\mathbf{-11.735} \pm 0.717$
CAL	-14.480 ± 0.630	-14.478 ± 0.696
Ours+CAL	$\mathbf{-7.960} \pm 0.648$	$\mathbf{-9.123} \pm 0.607$
DARC	-12.073 ± 0.670	-16.487 ± 1.364
Ours+DARC	$\mathbf{-8.837} \pm 0.767$	$\mathbf{-10.120} \pm 0.752$
GD3	-17.190 ± 0.521	-18.781 ± 0.499
Ours+GD3	$\mathbf{-14.478} \pm 0.449$	$\mathbf{-14.023} \pm 1.310$
TEEN	-59.701 ± 4.244	-146.084 ± 11.700
Ours+TEEN	$\mathbf{-31.717} \pm 2.929$	$\mathbf{-11.500} \pm 0.583$
CMAES	-20.149 ± 1.478	-16.104 ± 0.569
Ours+CMAES	$\mathbf{-16.914} \pm 0.588$	$\mathbf{-13.475} \pm 0.331$
TD3	-13.130 ± 0.810	-16.173 ± 1.330
SAC	-11.758 ± 0.829	-12.566 ± 1.255
Methods	ReacherPos	AntNav
TDDR	-16.302 ± 0.246	-24.558 ± 0.750
Ours+TDDR	$\mathbf{-13.290} \pm 0.208$	$\mathbf{-21.727} \pm 0.411$
CAL	-22.242 ± 0.351	-26.132 ± 0.490
Ours+CAL	$\mathbf{-17.694} \pm 0.397$	$\mathbf{-21.660} \pm 0.351$
DARC	-58.031 ± 3.686	-21.784 ± 0.407
Ours+DARC	$\mathbf{-20.321} \pm 0.666$	$\mathbf{-16.542} \pm 0.269$
GD3	-18.075 ± 0.351	-24.496 ± 0.616
Ours+GD3	$\mathbf{-14.618} \pm 0.295$	$\mathbf{-21.081} \pm 0.404$
TEEN	-17.095 ± 0.225	-22.448 ± 0.370
Ours+TEEN	$\mathbf{-13.418} \pm 0.248$	$\mathbf{-18.718} \pm 0.355$
CMAES	-16.380 ± 0.279	-26.403 ± 0.645
Ours+CMAES	$\mathbf{-14.421} \pm 0.231$	$\mathbf{-21.205} \pm 0.429$
TD3	-32.333 ± 2.154	-23.790 ± 0.382
SAC	-39.230 ± 0.970	-36.769 ± 0.826

TABLE III: The hyperparameter sensitivity analysis (Mean ± Standard deviation).

Methods	2DNav-v1	AntNav
0.01 (CAL)	-17.646 ± 0.876	-31.235 ± 0.676
0.001 (CAL)	$\mathbf{-7.960} \pm 0.648$	$\mathbf{-21.660} \pm 0.351$
0.0001 (CAL)	-12.959 ± 0.737	-33.940 ± 0.605
0.01 (DARC)	-58.705 ± 4.056	-18.626 ± 0.377
0.001 (DARC)	$\mathbf{-8.837} \pm 0.767$	$\mathbf{-16.542} \pm 0.269$
0.0001 (DARC)	-25.784 ± 1.448	-19.859 ± 0.387



(a) (b)

Fig. 5: Performance comparison across different numbers of actors: (a) on 2DNav-v1 and (b) on AntNav. The bold dot represents the mean of the final returns obtained from five random seeds, and the shaded area indicates the standard deviation.

D. Hyperparameter sensitivity testing

This section presents a sensitivity analysis of the hyperparameter β. We conduct experiments using the CAL and DARC methods as the baseline methods in the 2DNav-v1 and AntNav environments. The results, shown in Table III, evaluate the final return with β values set to 0.01, 0.001, and 0.0001. From the experimental results, it is evident that our method achieves the best performance when $\beta = 0.001$, so we select this value. Since the representation distinguishability regularization and the original actor loss may operate on different scales, we introduce β to adjust its proportion within the actor loss. When β is too large (e.g., 0.01), the higher weight causes the actor's original loss to be overshadowed, degrading policy performance. Similarly, when β is too small (e.g., 0.0001), the representation distinguishability regularization lacks sufficient influence, failing to promote actor diversity and thus degrading policy performance.

E. Performance comparison under more actors

This section presents the performance variation of our method as the number of actors increases. We use the CAL and DARC methods as the underlying DRL algorithms in the 2DNav-v1 and AntNav environments. Fig. 5 shows the performance improvements of our method over baseline approaches with 2, 5, 15, and 25 actors. From the results, we observe that the baseline methods exhibit varying performance with different numbers of actors. However, after integrating our method, the baseline approaches consistently achieve better performance across all actor configurations. Regardless of the number of actors, diversification helps avoid redundancy and promotes the generation of higher-quality actors. These high-quality actors elevate the performance ceiling, as the best actors are chosen for decision-making in multi-actor DRL methods, resulting in improved policy performance. Thus, our actor diversification method is essential for enhancing existing multi-actor DRL approaches.

V. CONCLUSION

This work develops a novel actor diversity method that can be seamlessly integrated into existing multi-actor DRL methods to enhance their performance. We calculate the inner product between each actor's representation vector and those of all other actors as an additional loss, which is combined with the original loss to form the final loss for each actor. This encourages actors to develop different understandings of the environment, fostering diversity and enabling better

policies. We demonstrate that our method improves nine SOTA multi-actor DRL methods across eight benchmark tasks, in terms of return. Future research will focus on enabling actors to exchange valuable learning experiences while preserving their diversity.

REFERENCES

[1] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4909–4926, 2021.

[2] B. Singh, R. Kumar, and V. P. Singh, "Reinforcement learning in robotic applications: a comprehensive survey," *Artificial Intelligence Review*, vol. 55, no. 2, pp. 945–990, 2022.

[3] L. Pan, Q. Cai, and L. Huang, "Softmax deep double deterministic policy gradients," *Advances in Neural Information Processing Systems*, vol. 33, pp. 11 767–11 777, 2020.

[4] B.-W. Guo, F. Chao, X. Chang, C. Shang, and Q. Shen, "Actor-critic with synthesis loss for solving approximation biases," *IEEE Transactions on Cybernetics*, vol. 54, no. 9, pp. 5323–5336, 2024.

[5] J. Lyu, X. Ma, J. Yan, and X. Li, "Efficient continuous control with double actors and regularized critics," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 7, 2022, pp. 7655–7663.

[6] Q. Li, W. Zhou, Z. Lu, and H. Li, "Simultaneous double q-learning with conservative advantage learning for actor-critic methods," *arXiv preprint arXiv:2205.03819*, 2022.

[7] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, "Evolution strategies as a scalable alternative to reinforcement learning," *arXiv preprint arXiv:1703.03864*, 2017.

[8] N. Kim, H. Baek, and H. Shin, "Pgps: Coupling policy gradient with population-based search," *OpenReview.net*, pp. 1–10, 2021.

[9] S. Khadka and K. Tumer, "Evolution-guided policy gradient in reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[10] C. Li, C. Gong, Q. He, X. Hou, and Y. Liu, "Centralized cooperative exploration policy for continuous control tasks," *arXiv preprint arXiv:2301.02375*, 2023.

[11] C. Li, C. Gong, Q. He, and X. Hou, "Keep various trajectories: promoting exploration of ensemble policies in continuous control," *Advances in Neural Information Processing Systems*, vol. 36, pp. 5223–5235, 2023.

[12] J. Liu and L. Feng, "Diversity evolutionary policy deep reinforcement learning," *Computational Intelligence and Neuroscience*, vol. 2021, no. 1, p. 5300189, 2021.

[13] J. Parker-Holder, A. Pacchiano, K. M. Choromanski, and S. J. Roberts, "Effective diversity in population based reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 18 050–18 062, 2020.

[14] J. Jiang, H. Piao, Y. Fu, Y. Hao, C. Jiang, Z. Wei, and X. Yang, "Phasic diversity optimization for population-based reinforcement learning," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 272–278.

[15] Y. Li, Y. Yang, W. Zhou, and T. Hospedales, "Feature-critic networks for heterogeneous domain generalization," in *International conference on machine learning*. PMLR, 2019, pp. 3915–3924.

[16] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[17] X. Li and Q. Liu, "Master-slave policy collaboration for actor-critic methods," in *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2022, pp. 1–7.

[18] B. Xie and B. Li, "Double actors and uncertainty-weighted critics for offline reinforcement learning," in *Proceedings of the 2023 7th International Conference on Innovation in Artificial Intelligence*, 2023, pp. 190–195.

[19] T. H. Nguyen and N. H. Luong, "Stable and sample-efficient policy search for continuous control via hybridizing phenotypic evolutionary algorithm with the double actors regularized critics," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2023, pp. 1239–1247.

[20] H. Chen, Z. Chen, A. Liu, and W. Fang, "Double actor-critic with td error-driven regularization in reinforcement learning," *arXiv preprint arXiv:2409.19231*, 2024.

[21] C. Xiao, H. Wang, Y. Pan, A. White, and M. White, "The in-sample softmax for offline reinforcement learning," *arXiv preprint arXiv:2302.14372*, 2023.

[22] S. Zhou, L. Feng, M. Mei, and M. Yao, "Dynamic resource management for federated edge learning with imperfect csi: A deep reinforcement learning approach," *IEEE Internet of Things Journal*, vol. 11, no. 18, pp. 30 400–30 412, 2024.

[23] K. Wang, R. Yang, Y. Zhou, W. Huang, and S. Zhang, "Design and improvement of sd3-based energy management strategy for a hybrid electric urban bus," *Energies*, vol. 15, no. 16, p. 5878, 2022.

[24] J. Chen, J. Zhang, N. Zhao, Y. Pei, Y.-C. Liang, and D. Niyato, "Joint device participation, dataset management, and resource allocation in wireless federated learning via deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 3, pp. 4505–4510, 2024.

[25] P. Hämäläinen, A. Babadi, X. Ma, and J. Lehtinen, "Ppo-cma: Proximal policy optimization with covariance matrix adaptation," in *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2020, pp. 1–6.

[26] L. Shao, Y. You, M. Yan, S. Yuan, Q. Sun, and J. Bohg, "Grac: Self-guided and self-regularized actor-critic," in *Conference on Robot Learning*. PMLR, 2022, pp. 267–276.

[27] Z.-W. Hong, T.-Y. Shann, S.-Y. Su, Y.-H. Chang, T.-J. Fu, and C.-Y. Lee, "Diversity-driven exploration strategy for deep reinforcement learning," *Advances in neural information processing systems*, vol. 31, 2018.

[28] H. Sheikh, M. Phielipp, and L. Boloni, "Maximizing ensemble diversity in deep reinforcement learning," in *International conference on learning representations*, 2022, pp. 1–13.

[29] T. Dai, Y. Du, M. Fang, and A. A. Bharath, "Diversity-augmented intrinsic motivation for deep reinforcement learning," *Neurocomputing*, vol. 468, pp. 396–406, 2022.

[30] K. Zhao, Y. Wang, Y. Chen, Y. Li, X. Niu *et al.*, "Efficient diversity-based experience replay for deep reinforcement learning," *arXiv preprint arXiv:2410.20487*, 2024.

[31] C. Li, T. Wang, C. Wu, Q. Zhao, J. Yang, and C. Zhang, "Celebrating diversity in shared multi-agent reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 3991–4002, 2021.

[32] Z. Liu, C. Yu, Y. Yang, Z. Wu, Y. Li *et al.*, "A unified diversity measure for multiagent reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 10 339–10 352, 2022.

[33] K. R. McKee, J. Z. Leibo, C. Beattie, and R. Everett, "Quantifying the effects of environment and population diversity in multi-agent reinforcement learning," *Autonomous Agents and Multi-Agent Systems*, vol. 36, no. 1, p. 21, 2022.

[34] M. Bettini, R. Kortvelesy, and A. Prorok, "Controlling behavioral diversity in multi-agent reinforcement learning," *arXiv preprint arXiv:2405.15054*, 2024.

[35] H. Xiong, T. Xu, L. Zhao, Y. Liang, and W. Zhang, "Deterministic policy gradient: Convergence analysis," in *Uncertainty in Artificial Intelligence*. PMLR, 2022, pp. 2159–2169.

[36] M. Xu, X. Chen, Z. Wen, W. Fu, and J. Wang, "A two-stage selective experience replay for double-actor deep reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 36, no. 9, pp. 16 864–16 878, 2025.

[37] J. Lyu, Y. Yang, J. Yan, and X. Li, "Value activation for bias alleviation: generalized-activated deep double deterministic policy gradients," *Neurocomputing*, vol. 518, pp. 70–81, 2023.

[38] O. S. Ajani, A. Kumar, and R. Mallipeddi, "Covariance matrix adaptation evolution strategy based on correlated evolution paths with application to reinforcement learning," *Expert Systems with Applications*, vol. 246, p. 123289, 2024.

[39] H. Guo, Z. Liu, R. Shi, W.-Y. Yau, and D. Rus, "Cross-entropy regularized policy gradient for multirobot nonadversarial moving target search," *IEEE Transactions on Robotics*, vol. 39, no. 4, pp. 2569–2584, 2023.

[40] T. Zhang, Z. Lin, Y. Wang, D. Ye, Q. Fu, W. Yang, X. Wang, B. Liang, B. Yuan, and X. Li, "Dynamics-adaptive continual reinforcement learning via progressive contextualization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 10, pp. 14 588–14 602, 2024.

[41] M. Xu, X. Chen, and J. Wang, "A novel topology adaptation strategy for dynamic sparse training in deep reinforcement learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 36, no. 7, pp. 12 271–12 283, 2025.