2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Offline Meta-Reinforcement Learning with Evolving Gradient Agreement

Jiaxing Chen[1], Weilin Yuan[2], Shaofei Chen[1,*], Furong Liu[1], Ao Ma[1], Zhenzhen Hu[1], Peng Li[1]
[1]*National University of Defense Technology, College of Intelligence Science and Technology, Changsha, China*
[2]*National University of Defense Technology, College of Information and Communication, Wuhan, China*
chenjiaxingmtn@nudt.edu.cn, chensf005@163.com

***Abstract*— Meta-Reinforcement Learning (Meta-RL) is a machine learning paradigm aimed at learning reinforcement learning policies that can quickly adapt to unseen tasks with few-shot data. Nevertheless, applying Meta-RL to real-world applications faces challenges due to the cost of data acquisition. To address this problem, offline Meta-RL has emerged as a promising solution, focusing on learning policies from pre-collected data that can effectively and rapidly adapt to unseen tasks. In this paper, we propose a new offline Meta-RL method called Meta-Actor-Critic with Evolving Gradient Agreement (MACEGA). MACEGA utilizes an evolutionary approach to estimate meta-gradients conductive to generalization across unseen tasks. During meta-training, gradient evolution is utilized to meta-update the value network and policies. Moreover, we use gradient agreement as an optimization objective for meta-learning, thereby enhancing the generalization ability of the meta-policy. We experimentally demonstrate the robustness of MACEGA in handling offline data quality. Furthermore, extensive experiments on various benchmarks provide empirical evidence that MACEGA outperforms previous state-of-the-art methods in generalizing to unseen tasks, thus demonstrating its potential for real-world applications.**

Offline meta-reinforcement learning, meta-reinforcement learning, evolving gradient, gradient agreement, generalization

I. Introduction

Meta-Reinforcement Learning (Meta-RL) is a machine learning approach that aims to learn a policy capable of adapting to new tasks within a task distribution, using few-shot data [1], [2]. However, the application of Meta-RL to real-world scenarios is challenging due to the complexity of the environment and the high costs associated with data generation. To address this problem, recent research has focused on offline Meta-RL [3]–[7] as a promising solution. Unlike traditional Meta-RL, offline Meta-RL learns policies from pre-collected data, enabling effective and fast adaptation to unseen new tasks.

Mitchell et al. [3] extended online Meta-RL to the offline setting, addressing the generalization problem from an optimization-based perspective. They proposed the Meta-Actor-Critic with Advantage Weighting (MACAW) algorithm, which utilizes Advantage Weighted Regression (AWR) [8] as the underlying reinforcement learning algorithm. MACAW leverages Model-Agnostic Meta-Learning (MAML) [9] method to optimize the agent's adaptability. However, MAML has some drawbacks in terms of computational efficiency [10]–[12], as it requires the computation of second-order gradients during the gradient update steps for meta-optimization, resulting in significant computational and memory burdens [11]. Furthermore, MACAW simply averages the gradients across all tasks for meta-learning, without considering the varying contributions of different task's gradients to the meta-gradient. This raises an important question: *Can an offline meta-reinforcement learning method be designed to consider multi-task gradient variability and eliminate the need to compute second-order gradients, thus enhancing an agent's ability to generalize across multiple new tasks?*

Evolutionary learning methods, such as Evolution Strategies [13], provide an alternative to traditional gradient-based methods that do not require the computation of higher-order derivatives. These methods have shown promising results in meta-learning, as demonstrated by recent advancements in meta reinforcement learning, including E-MAML [14], ES-MAML [15], and Evo-MAML [16]. These studies highlight the effectiveness of evolutionary-based meta-learning frameworks in adapting to new tasks. However, there is a lack of extensive research on the generalization capabilities of meta-agents in offline environments, as most existing works primarily focus on online meta reinforcement learning problems. Therefore, further investigation is needed to explore the generalization ability of meta-agents in offline settings.

Inspired by this, we propose a new offline meta-RL framework called Meta-Actor Critic with Evolving Gradient Agreement (MACEGA). MACEGA constructs meta-reinforcement learning agents offline, ensuring both generalization and robustness. An evolutionary approach is used to estimate the meta-gradients of the value network and the policy network. The multi-task gradient weights are automatically adjusted based on the difference in the degree of influence of each task gradient on the meta-gradient. The contributions of this work are as follows:

- We proposed a Meta-Actor-Critic with Evolving Gradient Agreement (MACEGA) method. First, meta-updating across tasks is achieved by using evolving gradients to estimate meta-gradients. Then, the degree of influence of multi-task gradients on the meta-gradient is evaluated, and automatic adjustment of the meta-gradient is achieved through gradient agreement, which improves the generalization ability of the agent in unseen new tasks;
- Our presented MACEGA has good robustness. The MACEGA method exhibits good robustness when

[*]Corresponding author

trained on offline datasets of different quality in MuJoCo's three types of environments;
- MACEGA exhibits good generalization performance. A series of experimental results demonstrate that it can effectively utilize online experience to adapt to new tasks quickly. Additionally, the policy's generalization ability outperforms other offline meta-reinforcement learning methods.

II. Preliminaries

A. Offline Meta-Reinforcement Learning

Reinforcement learning problems are commonly modeled using Markov Decision Processes (MDPs) to maximize the cumulative reward for an agent. A MDP is represented by a quadruple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r)$, which includes the state space $\mathcal{S}$, action space $\mathcal{A}$, state transition probability distribution $\mathcal{P}$, and reward function r. The agent interacts with the environment using two probability distributions: the state transition probability distribution $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ and the reward probability distribution $r_t = r(s_t, a_t, s_{t+1})$. The agent's objective is to maximize the expected return $\mathcal{R} = \sum_t \gamma^t r_t$, where $\gamma \in [0, 1]$ is the discount factor. In offline meta-reinforcement learning settings, a task $\mathcal{T} \in \{\mathcal{T}_i\}$ is defined as $(\mathcal{M}_i, \mu_i)$, consisting of a MDP $\mathcal{M}_i$ and a policy μ_i. The agent is provided with a pre-collected dataset D_i for each task $\mathcal{T}$, which includes trajectories sampled using μ_i. The agent is trained using a subset of training tasks $\mathcal{T}_{tr}$ and is expected to find the optimal policy in a set of test tasks $\mathcal{T}_{ts}$ that is disjoint from $\mathcal{T}_{tr}$. Mitchell et al. [3] proposed two different meta-testing approaches, offline testing and online fine-tuning testing. Firstly, in the offline setting, the agent uses a small batch of experiences sampled from the policy μ_{ts} to find the best-performing policy for solving $\mathcal{M}_{ts}$. Secondly, in the online fine-tuning setting, the agent can perform online data collection and learning after obtaining the offline data D_{ts}. Both of these meta-testing approaches were adopted in our experiments.

B. Model-Agnostic Meta-Learning

The MAML algorithm [9] and its variants [12], [15]–[18] are commonly used algorithms for addressing meta reinforcement learning (Meta RL) [12], [18], [19] and few-shot learning problems [10], [11], [20]. These algorithms employ a bi-level optimization approach aimed at achieving fast adaptation to unseen new tasks with a small number of update steps. Specifically, MAML performs meta-learning by first obtaining a set of initial policy parameters θ and then fine-tuning them through online adaptation for new tasks. In the inner loop, the meta-policy adapts to the new task through the update rule

$$\theta' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\theta), \tag{1}$$

where $\mathcal{L}_{\mathcal{T}_i}(\theta)$ represents the loss function for task $\mathcal{T}_i$ and α is the inner loop learning rate. In the outer loop, the meta-update is performed as

$$\theta \leftarrow \theta - \beta \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\theta'), \tag{2}$$

where β is the outer loop learning rate.

III. Method

To overcome the challenges associated with higher-order derivative estimation and inconsistent gradient direction in MACAW, we introduce a novel approach called Meta-Actor-Critic with Evolving Gradient Agreement (MACEGA). MACEGA is an offline meta-reinforcement learning algorithm that employs evolving gradient meta-learning and gradient agreement optimization to meta-learn the value function and initial policy parameters. In Section III-A, we provide an overview of the overall framework of MACEGA. We then proceed to explain the meta-training process in detail in Sections III-B and III-C.

A. MACEGA Architecture

The MACEGA method's meta-training process consists of an inner and outer loop structure, as illustrated in Fig. 1. In the inner loop, the value network and policy network are updated using gradient descent. The parameters of both networks are adjusted based on the calculated gradients [21]. Moving on to the outer loop, the gradient agreement method is employed to determine the discrepancy between the direction of multi-task gradients and the direction of the meta-gradient. The discrepancy is used as a weight to update the policy parameters, thereby computing the meta-gradients. The meta-tests for offline meta-reinforcement learning are categorized into two types. In offline testing, the meta-policy interacts with the offline test set to evaluate its performance. On the other hand, online fine-tuning tests require agents to interact with the real environment, enabling assessment of the online generalization ability of the offline policies. The MACEGA algorithm is summarized in Alg. 1.

B. Inner-Loop Process

The inner loop process of MACEGA consists of gradient updating and evolving gradient computation for the value network parameter φ and the policy parameter θ, and can be found in lines 3-10 of Algorithm 1. When given a batch of offline training data D^{tr}, MACEGA updates the value function through a single gradient step:

$$\varphi' = \varphi - \alpha_1 \nabla_\varphi \mathcal{L}_V(f_\varphi, D), \tag{3}$$

where the loss of the value function is defined in the AWR algorithm [8] as $\mathcal{L}_V(f_\varphi, D) := \mathbb{E}_{s,a\sim D}[(V_\varphi(s) - \mathcal{R}_D(s, a))^2]$, $\mathcal{R}_D(s, a)$ represents the Monte Carlo return associated with the action a taken by the agent in state s in the dataset D.

After adapting the value network, we compute the evolving gradient g_V^e of the value parameter φ' by the evolutionary method [21]. Specifically, we apply perturbations $\epsilon_P \sim \mathcal{N}(0, 1)$ to the adapted value parameter φ' to obtain P perturbed value parameters $\varphi_P = \varphi' + \epsilon_P$. Then, we compute the training losses of these perturbed variants and update the value parameter φ^* by affine combination to complete the evolutionary learning:

$$\varphi^* = w_1\varphi_1 + w_2\varphi_2 + \cdots + w_P\varphi_P, \tag{4}$$

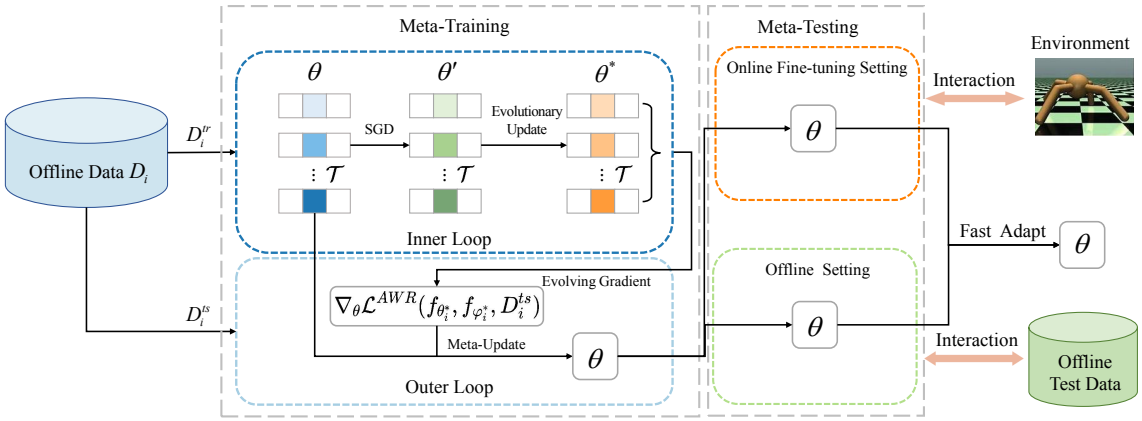

Fig. 1. The overview of MACEGA. In the meta-training phase, the policy parameter θ is updated using an evolutionary method, which follows a 1-step SGD update in the inner loop. The evolving policy parameter θ^* is calculated in the outer loop using the offline test data D^{ts} to compute the evolving gradient. The evolving gradient is utilized to perform the meta-updating across tasks. During the meta-testing phase, we adopt the MACAW [3] approach, which consists of meta-testing under offline conditions and online fine-tuning. The goal is to evaluate the generalization performance of the agent.

Algorithm 1: MACEGA Algorithm

Input: Tasks $\{\mathcal{T}_i\}$, offline data$\{D_i\}$
Hyperparameters: learning rates $\alpha_1, \alpha_2, \beta_1, \beta_2$, training iterations n, noise σ, temperature τ, number of perturbed models P

1 Randomly initialize value and policy parameters φ, θ
2 **for** n *steps* **do**
3 **for** *task* $\mathcal{T} \in \mathcal{T}_i$ **do**
4 Sample disjoint batches $D_i^{tr}, D_i^{ts} \sim D_i$
5 $\varphi_i' = \varphi - \alpha_1 \nabla_\varphi \mathcal{L}_V(f_\varphi, D_i^{tr})$
6 Evolving update value parameter φ_i^* with Eq. (4)
7 $\theta_i' = \theta - \alpha_2 \nabla_\theta \mathcal{L}_P(f_\theta, f_{\varphi_i'}, D_i^{tr})$
8 Evolving update policy parameter θ_i^* with Eq. (6)
9 **end**
10 $\varphi \leftarrow \varphi - \beta_1 \sum_i [\nabla_\varphi \mathcal{L}_V(f_{\varphi_i^*}, D_i^{ts})]$
11 $\omega_i = \frac{\sum_{j \in D_i^{ts}} (g_i^T g_j)}{\sum_{k \in D_i^{ts}} |\sum_{j \in D_i^{ts}} (g_k^T g_j)}$ for all $\mathcal{T}_i$
12 $\theta \leftarrow \theta - \beta_2 \sum_i [\omega_i \nabla_\theta \mathcal{L}^{AWR}(f_{\theta_i^*}, f_{\varphi_i^*}, D_i^{ts})]$
13 **end**

where the weights are defined as $w_1, w_2, \ldots, w_P = \text{softmax}([-\mathcal{L}_V(f_{\varphi_1}, D^{tr}), -\mathcal{L}_V(f_{\varphi_2}, D^{tr}), \ldots, -\mathcal{L}_V(f_{\varphi_P}, D^{tr})]/\tau)$. The temperature factor τ rescales the losses to adjust the scale of weight changes.

Then we update the initial policy parameter θ on a gradient step with a step size of α_2:

$$\theta' = \theta - \alpha_2 \nabla_\theta \mathcal{L}_P(f_\theta, f_{\varphi'}, D^{tr}), \tag{5}$$

the loss of the adapted policy is defined in the MACAW algorithm [3] as $\mathcal{L}_P(f_\theta, f_{\varphi'}, D) = \mathcal{L}^{AWR}(f_\theta, f_{\varphi'}, D) + \lambda \mathcal{L}^{ADV}(f_\theta, f_{\varphi'}, D)$, where $\mathcal{L}^{AWR}(f_\theta, f_{\varphi'}, D) := \mathbb{E}_{s,a \sim D}[-\log \mu_\theta(a|s) \exp(\frac{1}{T}(\mathcal{R}_D(s,a) - V_{\varphi'}(s)))]$ is the advantage-weighted regression loss, which is used to update the policy network, and $\mathcal{L}^{ADV}(f_\theta, f_{\varphi'}, D) := \mathbb{E}_{s,a \sim D}[(A_\theta(s,a) - (\mathcal{R}_D(s,a) - V_{\varphi'}(s)))^2]$ is the action advantage regression loss. $A_\theta(s,a)$ represents the advantage function estimated using the current policy's parameters θ.

After the policy performs gradient updating, we implement evolutionary updating on it. Similar to the evolutionary update process for value networks, the loss of P perturbed policy models $\theta_P = \theta' + \epsilon_P$ is calculated, and then the evolutionary learning of the policy parameter θ^* is accomplished by the following:

$$\theta^* = w_1 \theta_1 + w_2 \theta_2 + \cdots + w_P \theta_P, \tag{6}$$

where the weights are computed by $w_1, w_2, \ldots, w_P = \text{softmax}([-\mathcal{L}_P(f_{\theta_1}, f_{\varphi'}, D^{tr}), -\mathcal{L}_P(f_{\theta_2}, f_{\varphi'}, D^{tr}), \ldots, -\mathcal{L}_P(f_{\theta_P}, f_{\varphi'}, D^{tr})]/\tau)$. Note that here we use the gradient-updated value parameter φ' rather than the evolutionary-updated φ^*; the evolving parameter φ^* is only used for estimating the meta-gradient, and is not involved in parameter updating in the inner loop.

C. Outer-Loop Process

To facilitate rapid adaptation during meta-testing, we employ the evolving value gradient meta-update for the value parameters in the outer loop phase. To tackle the issue of inconsistent gradient direction in MAML, we utilize the gradient agreement method [22] as the meta-optimization objective for the policy. Additionally, we calculate a weighted average of the evolving policy gradient to regulate the extent to which different tasks contribute to the parameter update. This process can be observed in lines 10-12 of Algorithm 1.

In the outer loop, we evaluate the evolving parameters in the test data D^{ts} which is uncorrelated with the inner loop data D^{tr} and use the evolving gradient as the meta-gradient

$$\varphi \leftarrow \varphi - \beta_1 \sum_i [\nabla_\varphi \mathcal{L}_V(f_{\varphi^*}, D^{ts})] \tag{7}$$

Unlike the value network update, we use the gradient agreement method to optimize the initial policy parameters

in the outer loop, considering that the gradients of different tasks have different importance for the meta-policy update. Assuming that the gradient update vector for each task is represented by g_i, $\omega_i = \frac{\sum_{j \in D_i^{ts}} (g_i^T g_j)}{\sum_{k \in D_i^{ts}} | \sum_{j \in D_i^{ts}} (g_k^T g_j)}$. When the gradient of a task aligns with the average gradient of all tasks in a batch, the corresponding weight ω_i increases, indicating that this task contributes more to updating the model parameters. By employing the gradient agreement method, we effectively address the issue of negative adaptation in individual tasks during multi-task meta-learning and promote the generalization of the policy across all tasks.

IV. Experiments

We conducted experiments on three types of meta-RL tasks to evaluate the few-shot generalization performance of the proposed MACEGA algorithm. Our goal is to experimentally answer the following questions:

1) Can MACEGA achieve performance gains in the few-shot policy generalization compared to other strong baselines?
2) Can MACEGA effectively utilize online experience to fast adapt to new tasks during online adaptation meta-testing?
3) Can MACEGA show robustness to offline data quality?
4) Can the key design of MACEGA be conducive to improving the generalizability of the algorithm?

A. Environments

We adopt multi-task MuJoCo [23] control tasks to make comparisons as classical benchmarks commonly used in meta-RL [14]–[17], [19]. Experiments in Cheetah-Dir, Ant-Dir and Walker-Param strictly follow the datasets and settings in [3]. The agents in all three tasks are penalized with large control signals. The dataset for three control tasks contains the full replay buffer for training an RL agent with SAC [24].

TABLE I
OFFLINE META-REINFORCEMENT LEARNING ENVIRONMENTS SETTING.

Environment	Training tasks	Testing tasks	Training steps
Cheetah-Dir	2	2	2.5M
Ant-Dir	45	5	2M
Walker-Param	45	5	1M

B. Implementation and baselines

In offline meta-RL challenges, we put the following baselines to the test:

- MACAW [3]: combines the bi-level optimization-based meta-reinforcement learning algorithm MAML [9] with AWR [8] algorithm, and is currently the state-of-the-art optimization-based offline meta-reinforcement learning method;
- Offline MT+FT [3]: Multi-Task Offline RL with Fine-Tuning;
- CORRO [6]: proposes a task-represented contrastive learning framework robust to the distributional mismatch of agent action policies in offline training and testing. CORRO is the state-of-the-art offline meta-reinforcement learning algorithm based on context variables;
- FOCAL [4]: proposes a novel negative-power distance metric learning method to train the context encoder for task inference as an end-to-end offline meta-RL algorithm with high efficiency.

C. Results

In this section, we provide a comparative evaluation of our proposed method and analyze its generalization performance in both the offline meta-testing setting and the online fine-tuning setting. In Section IV-C-1), we assess the generalization performance of our proposed method under the online meta-test setting and compare it with previous offline meta-reinforcement learning algorithms. Furthermore, in Section IV-C-2), we examine the generalization performance of MACEGA in the online meta-test. Additionally, in Section IV-C-3), we evaluate the robustness of MACEGA in terms of data quality. Finally, in Section IV-C-4), we demonstrate the effectiveness of the evolving gradient approach and the gradient agreement method in improving the generalization ability of the offline meta-reinforcement learning agent through ablation experiments.

1) Offline Meta-Testing Performance: To evaluate the effectiveness of MACEGA in offline adaptation to new tasks, we conducted the first experiment in three MuJoCo environments. Fig. 2 illustrates the training performance of the five algorithms and Tab. II presents the final performance of these algorithms. The results depicted in Fig. 2 demonstrate that MACEGA exhibits competitive sample efficiency across all three environments. However, in the Walker-Param environment, MACEGA initially falls short of MACAW's performance. Nevertheless, after approximately *10k* training steps, MACEGA outperforms MACAW in terms of generalization performance. The higher learning difficulty of the Walker-Param environment explains why MACEGA requires more training time to adapt to it. In the Cheetah-Dir environment, which only consists of 2 training tasks, the meta-learning capability of MACEGA is not fully utilized due to the limited number of training tasks. In contrast, MACEGA demonstrates better generalization performance in the Ant-Dir and Walker-Param environments, where there are 45 training tasks available. The highlighted portion in Tab. II indicates the best generalization performance achieved in the current environments. MACEGA achieves state-of-the-art performance in all three offline control environments. While Offline MT+FT shows some learning results in the simpler Cheetah-Dir environment, it fails to adapt well to the more challenging Ant-Dir and Walker-Param environments. Compared to Offline MT+FT, MACEGA trains efficiently and exhibits relatively robust performance across all problems. It provides a promising approach to learning representations from the multi-task offline data that can be effectively

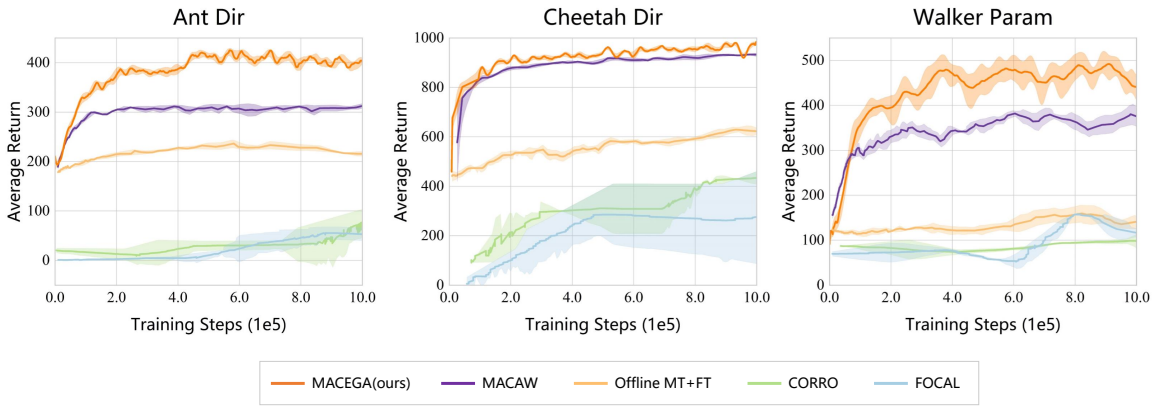

Fig. 2. Offline meta-testing performance of MACEGA against baselines run over three random seeds in unseen tasks. Shaded regions show one standard error of the average return of three seeds.

TABLE II

AVERAGE META-TESTING RETURNS FOR MACEGA RELATIVE TO OTHER FOUR BASELINES.

Algorithms	Cheetah-Dir	Ant-Dir	Walker-Param
FOCAL [4]	680.9±46.6	151.3±24.6	245.6±37.8
MACAW [3]	945.6±10.3	328.1±9.3	381.9±20.1
Offline MT+FT [3]	735.5±10.2	237.1±9.0	187.8±18.2
CORRO [6]	823.5±37.0	193.3±32.1	300.5±34.2
MACEGA (Ours)	**1031.8±30.7**	**415.8±12.2**	**566.2±17.9**

TABLE III

COMPARISON OF AVERAGE RETURNS BETWEEN MACEGA AND MACAW FOR TEST TASKS PERFORMED AFTER OFFLINE ADAPTATION (0 STEP) FOLLOWED BY ONLINE FINE-TUNING FOR 500 AND *1k* ONLINE INTERACTIONS.

	MACEGA (Ours)			MACAW		
Online Steps	0	500	*1k*	0	500	*1k*
Ant-Dir	409.3	423.1	441.9	251.9	263.1	278.3
Cheetah-Dir	33.7	390.9	552.1	-8.3	53.3	107.5
Walker-Param	548.7	532.5	566.7	324.9	276.4	289.4

adapted to new tasks during meta-testing.

2) Online Fine-Tuning Meta-Testing Performance: To assess the online meta-learning capability of MACEGA, we conducted experiments in three control environments to compare its performance with MACAW under online adaptation conditions. An ideal offline meta-reinforcement learning algorithm should be able to leverage both offline and online data during meta-testing. We compared the generalization performance of MACEGA and MACAW at 0 steps, 500 steps, and *1k* steps of online adaptation, as shown in Tab. III. In environments with fewer testing task scenarios, such as Cheetah-Dir, the generalization performance of both algorithms is relatively average, possibly due to the limited task diversity. However, in the more diverse task environments of Ant-Dir and Walker-Param, MACEGA exhibits a strong ability to adapt online. Notably, even in the challenging Walker-Param environment, where utilizing online experience is most difficult, MACEGA demonstrates fast adaptation to new tasks.

3) The Robustness of Offline Data Quality: To evaluate the robustness of MACEGA on offline datasets, we test the performance of the algorithm using three quality datasets. Tab. IV presents the average returns of MACEGA and MACAW for meta-reinforcement learning tasks on these offline datasets. The results in Tab. IV demonstrate that when the offline data are of medium quality, MACAW shows a slight decrease in performance of approximately 5% across the three environments compared to its performance with the expert data training. However, when random offline data are used, MACAW's performance decreases significantly, particularly in the Ant-Dir environment where the decrease reaches 16%. In contrast, the difference between the training performance of MACEGA and its performance with expert dataset is no more than 10%, regardless of whether the dataset is of medium quality or random. This indicates that MACEGA is capable of maintaining its robustness when trained with datasets of varying qualities, potentially due to the enhanced generalization provided by evolutionary meta-learning.

4) Ablation: To evaluate the effectiveness of evolving gradient and gradient agreement optimization, we conducted ablation experiments. These two components play a crucial role in the performance of MACEGA. The experiments involved removing either the evolving gradient or the gradient agreement part while keeping the other parts of the MACEGA framework unchanged. The two ablation variants are referred to as No EG (No Evolving Gradient) and No GA (No Gradient Agreement). In the No EG variant, the evolving gradient is removed, and meta-gradient estimation is conducted using SGD. In the No GA variant, the gradient agreement optimization is removed, and average meta-gradient computation is used as an alternative. Fig. 3 illustrates the performance of MACEGA and its two ablation variants in the Walker-Param environment.

TABLE IV

COMPARISON OF THE PERFORMANCE OF MACEGA AND MACAW WITH DIFFERENT QUALITIES OF OFFLINE DATA DURING THE META-TESTING PHASE. THE ↓ INDICATES THE DECREASE IN PERFORMANCE WITH OTHER DATA QUALITIES.

Environment	MACEGA (Ours)			MACAW		
	Expert	Medium	Random	Expert	Medium	Random
Ant-Dir	415.8±12.2	404.7±20.3 (↓ 2.7%)	389.6±12.0 (↓ 6.3%)	328.1±9.3	310.6±15.4 (↓ 5.3%)	275.5±28.7 (↓ 16.0%)
Cheetah-Dir	1031.8±30.7	1006.4±25.6 (↓ 2.4%)	998.7±32.4 (↓ 3.2%)	945.6±10.3	887.9±22.9 (↓ 6.1%)	851.2±29.6 (↓ 10.0%)
Walker-Param	566.2±17.9	560.2±13.7 (↓ 1.1%)	558.6±7.6 (↓ 1.3%)	381.9±20.1	365.8±22.7 (↓ 4.2%)	340.2±10.1 (↓ 10.9%)

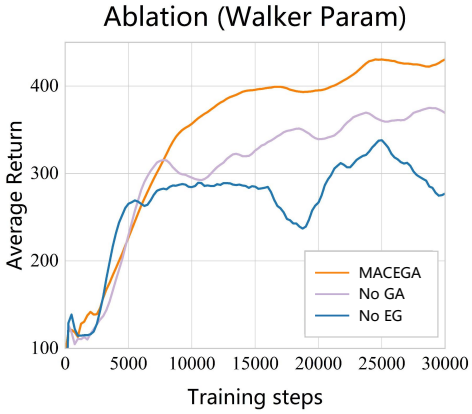

Fig. 3. Ablation study of the effect of components on MACEGA generalization performance.

The results indicate that although the two ablation variants showed similar performance to MACEGA in the first 8000 training steps, MACEGA outperformed the two ablation variants in subsequent training. Additionally, the No EG variant, which does not incorporate evolving gradient meta-learning, exhibited the worst asymptotic performance compared to the other two algorithms, suggesting that evolving gradient meta-learning is a crucial factor in improving the generalization ability of the model.

To examine the effectiveness of the gradient agreement optimization method in enhancing the generalization performance of each test task, ablation experiments were conducted in the Ant-Dir environment. Fig. 4 shows the generalization performance of MACEGA compared to MACAW under each test task during the meta-training process, as well as the performance when averaging all the test task returns. The black dashed line in the figure represents the case where the return is 400. In the three test tasks (Task 23, Task 30, and Task 41), MACEGA consistently maintains stable performance at a gain value of around 400. In contrast, MACAW's return, without utilizing gradient agreement optimization, only reaches a stable performance level of around 350. These results suggest that the weighted averaging of evolving gradients, facilitated by the gradient agreement optimization, can effectively enhance the generalization performance of the agents for each subtask.

V. RELATED WORK

A. Meta-RL

Existing approaches to meta-reinforcement learning can be categorized into two groups. One group is contextual meta-reinforcement learning methods, which use recurrent neural networks [25]–[27] to tune the network based on experience, or inference networks [19], [28], [29] to achieve meta-learning by learning contextual encoders. The other class is the meta-reinforcement learning method based on bi-level optimization [9], [15], [16] which implements meta-learning through upper-layer meta-level optimization, and then quickly adapts to unseen tasks in the lower layer of adaptation-level optimization. In previous work, the former class of methods tends to achieve higher asymptotic performance, while the latter class is usually robust to out-of-distribution tasks.

B. Offline Meta-RL

Offline meta-reinforcement learning aims to learn policies from pre-collected data to quickly adapt to new, unseen tasks. Recent research in this field can be categorized into two main groups. The first group extends from traditional online meta-reinforcement learning settings and includes methods such as context-based FOCAL [4], CORRO [6], and the meta-gradient optimization-based MACAW [3]. These approaches focus on improving generalization performance by leveraging meta-learning techniques. The second group addresses the generalization problem from a sequence modeling perspective. Prompt-DT [5] utilizes collected cues as prefixes to generalize tasks without the need for an explicit context encoder. However, this type of approach requires high-quality immediate hot-start data, which can be challenging to obtain for unseen tasks. To overcome this limitation, Ni et al. [7] proposed MetaDiffuser, which combines a context-based approach with a sequence modeling approach to achieve data robustness for offline meta-reinforcement learning agents. Our approach, MACEGA, belongs to the first category of work and combines the strengths of meta-gradient optimization and evolutionary optimization methods. Building upon the foundation laid by MACAW [3], MACEGA not only avoids the need for higher-order gradient computation and improves the generalization ability of the meta-policy, but also enhances the robustness of the agent to the quality of offline data.

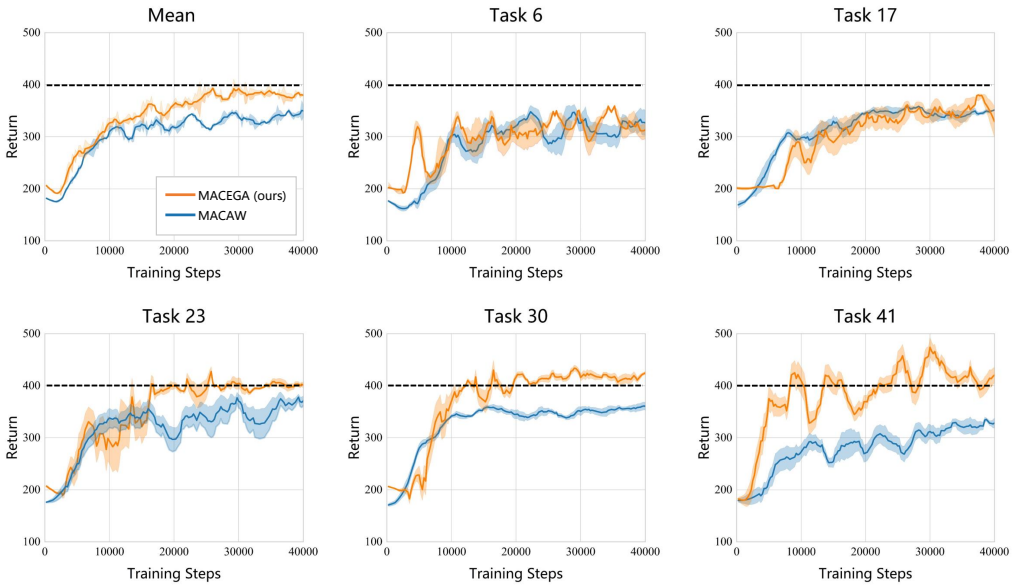

Fig. 4. Comparison of generalization performance of MACEGA and MACAW on five different meta-testing tasks.

VI. Conclusion

In this paper, we propose MACEGA, an offline meta-reinforcement learning approach that eliminates the need for higher-order gradient computation. Specifically, we utilize evolving gradient estimation to compute meta-gradients for meta-updating value networks. To further enhance the generalization performance of offline meta-reinforcement learning agents, we introduce the gradient agreement method as the meta-optimization objective for policy optimization. In addition, we incorporate weights to measure the contribution of each task to the meta-policy update. Our method outperforms previous baselines in terms of asymptotic performance on multiple benchmarks, achieving performance improvements ranging from 10% to 50% compared to MACAW. Moreover, our approach exhibits robustness, maintaining stable performance even when trained with random-quality offline data.

Acknowledgment

This research was funded by National Natural Science Foundation of China under Grant 62376280.

References

[1] J. Beck, R. Vuorio, E. Liu, et al. A Survey of Meta-Reinforcement Learning. ArXiv preprint arXiv:2301.08028, 2023.
[2] T. Yu, D. Quillen, Z. He, et al. Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning. In Conference on Robot Learning, 2019.
[3] E. Mitchell, R. Rafailov, X. Peng, et al. Offline meta-reinforcement learning with advantage weighting. In International Conference on Machine Learning, PMLR, 7780-7791, 2021.
[4] L. Li, R. Yang, and D. Luo. FOCAL: Efficient Fully-Offline Meta-Reinforcement Learning via Distance Metric Learning and Behavior Regularization. In International Conference on Learning Representations, 2020.
[5] M. Xu, Y. Shen, S. Zhang, et al. Prompting Decision Transformer for Few-Shot Policy Generalization. In International Conference on Machine Learning, 24631-24645, 2022.
[6] H. Yuan and Z. Lu. Robust task representations for offline meta-reinforcement learning via contrastive learning. In International Conference on Machine Learning, PMLR, 25747-25759, 2022.
[7] F. Ni, J. Hao, Y. Mu, et al. MetaDiffuser: Diffusion Model as Conditional Planner for Offline Meta-RL. In International Conference on Machine Learning, 2023.
[8] X. Peng, A. Kumar, G. Zhang, et al. Advantage weighted regression: Simple and scalable off-policy reinforcement learning. ArXiv preprint arXiv:1910.00177, 2019.
[9] C. Finn, P. Abbeel, and S. Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In International Conference on Machine Learning, 2017.
[10] A. Antoniou, H. Edwards, and A. Storkey. How to train your MAML. In International Conference on Learning Representations, 2019.
[11] A. Rajeswaran, C. Finn, S. Kakade, et al. Meta-Learning with Implicit Gradients. In Neural Information Processing Systems, 2019.
[12] H. Liu, R. Socher, and C. Xiong. Taming MAML: Efficient unbiased meta-reinforcement learning. In the International Conference on Machine Learning, PMLR, 4061–4071, 2019.
[13] T. Salimans, J. Ho, X. Chen, et al. Evolution Strategies as a Scalable Alternative to Reinforcement Learning. ArXiv preprint arXiv:1703.03864, 2017.
[14] B. Stadie, G. Yang, R. Houthooft, et al. Some Considerations on Learning to Explore via Meta-Reinforcement Learning. In International Conference on Learning Representations, 2018.
[15] X. Song, W. Gao, Y. Yang, et al. ES-MAML: Simple Hessian-Free MetaLearning. In International Conference on Learning Representations, 2020.
[16] J. Chen, W. Yuan, S. Chen, et al. Evo-MAML: Meta-Learning with Evolving Gradient. Electronics, 12(18), 2023.
[17] J. Rothfuss, D. Lee, I. Clavera, et al. ProMP: Proximal Meta-Policy Search. In International Conference on Learning Representations, 2019.
[18] Q. Fu, Z. Wang, N. Fang, et al. MAML2: Meta Reinforcement Learning via Meta-Learning for Task Categories. Frontiers of Computer Science, 17(4), 2022.
[19] K. Rakelly, A. Zhou, D. Quillen, et al. Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables. In the International Conference on Machine Learning, PMLR, 5331–5340, 2019.
[20] A. Ghadirzadeh, X. Chen, P. Poklukar, et al. Bayesian Meta-Learning for Few-Shot Policy Adaptation Across Robotic Platforms. IEEE/RSJ International Conference on Intelligent Robots and Systems, 1274-1280, 2021.

[21] O. Bohdal, Y. Yang, and T. Hospedales. EvoGrad: Efficient Gradient-Based Meta-Learning and Hyperparameter Optimization. In Neural Information Processing Systems, 2021.
[22] A. Eshratifar, D. Eigen, and M. Pedram. Gradient Agreement as an Optimization Objective for Meta-Learning. Statistics, 2018.
[23] E. Todorov, T. Erez, and Y. Tassa. MuJoCo: A physics engine for model-based control. In 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems.
[24] T. Haarnoja, A. Zhou, P. Abbeel, et al. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In International Conference on Machine Learning, PMLR, 1861–1870, 2018.
[25] Y. Duan, J. Schulman, X. Chen, et al. RL²: Fast Reinforcement Learning via Slow Reinforcement Learning. ArXiv preprint arXiv:1611.02779, 2016.
[26] R. Fakoor, P. Chaudhari, S. Soatto, et al. Meta-Q Learning. International Conference on Learning Representations, 2019.
[27] S. Rohani, S. Hedayatian and M. Baghshah. BIMRL: Brain Inspired Meta Reinforcement Learning. IEEE/RSJ International Conference on Intelligent Robots and Systems, 9048-9053, 2022.
[28] L. Zintgraf, K. Shiarlis, M. Igl, et al. VariBAD: A Very Good Method for Bayes Adaptive Deep RL via Meta-Learning. International Conference on Learning Representations, 2019.
[29] L. Wen, S. Zhang, H. Tseng, et al. Improved Robustness and Safety for Pre-Adaptation of Meta Reinforcement Learning with Prior Regularization. IEEE/RSJ International Conference on Intelligent Robots and Systems, 8987-8994, 2022.