Complex & Intelligent Systems (2025) 11:94
https://doi.org/10.1007/s40747-025-01785-0

ORIGINAL ARTICLE

# Demonstration and offset augmented meta reinforcement learning with sparse rewards

Haorui Li[1,2] · Jiaqi Liang[1] · Xiaoxuan Wang[1] · Chengzhi Jiang[1,2] · Linjing Li[1,2] · Daniel Zeng[1,2]

Received: 5 August 2024 / Accepted: 27 January 2025
© The Author(s) 2025

## Abstract

This paper introduces DOAMRL, a novel meta-reinforcement learning (meta-RL) method that extends the Model-Agnostic Meta-Learning (MAML) framework. The method addresses a key limitation of existing meta-RL approaches, which struggle to effectively use suboptimal demonstrations to guide training in sparse reward environments. DOAMRL effectively combines reinforcement learning (RL) and imitation learning (IL) within the inner loop of the MAML framework, with dynamically adjusted weights applied to the IL component. This enables the method to leverage the exploration strengths of RL and the efficiency benefits of IL at different stages of training. Additionally, DOAMRL introduces a meta-learned parameter offset, which enhances targeted exploration in sparse reward settings, helping to guide the meta-policy toward regions with non-zero rewards. To further mitigate the impact of suboptimal demonstration data on meta-training, we propose a novel demonstration data enhancement module that iteratively improves the quality of the demonstrations. We provide a comprehensive analysis of the proposed method, justifying its design choices. A comprehensive comparison with existing methods in various stages (including training and adaptation), using both optimal and suboptimal demonstrations, along with results from ablation and sensitivity analysis, demonstrates that DOAMRL outperforms existing approaches in performance, applicability, and robustness.

**Keywords** Meta learning · Reinforcement learning · Sparse reward · Suboptimal demonstration · One-shot learning · Imitation learning

## Introduction

Adapting to new environments is an essential ability for autonomous intelligent agents to be effectively applied to the real world. Meta-RL algorithms, represented by Model Agnostic Meta-Learning (MAML) [1], provide a solution to realize this capability. The core idea behind these algorithms is to train a distilled meta-policy across several similar tasks. This meta-policy represents commonalities of these tasks and can perform near optimally on each specific task by adapting to it using a small number of task-specific samples.

Compared to training each task from scratch, MAML's need for only a few adaptation samples is appealing. However, these samples should ideally represent the task as possible to facilitate meaningful adaptation and meta-learning. In sparse reward environments, obtaining non-zero rewards is challenging, making it difficult for collected samples to capture task-specific features. The optimization and adaptation of the meta-policy encounter significant challenges in such scenarios. This hinders the application of algorithms like MAML in real-world situations with sparse rewards.

✉ Jiaqi Liang
liangjiaqi2014@ia.ac.cn

Haorui Li
lihaorui2021@ia.ac.cn

Xiaoxuan Wang
xiaoxuan.wang@ia.ac.cn

Chengzhi Jiang
jiangchengzhi2021@ia.ac.cn

Linjing Li
linjing.li@ia.ac.cn

Daniel Zeng
dajun.zeng@ia.ac.cn

1 State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China

2 School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China

Over the years, many reinforcement learning (RL) methods have been developed to address the challenges of sparse reward environments. These methods primarily focus on improving exploration and can be broadly classified into two categories: rewarding novel states and rewarding diverse behaviors [2]. Approaches that reward novel states aim to create an intrinsic reward function that encourages agents to explore new states, compensating for the scarcity of external rewards. Methods within this category can be further divided into two types: those based on state representation prediction and those that focus on uncertainty about the environment. State representation prediction-based methods use prediction errors of states as intrinsic rewards. Examples include GAEX [3], which differentiates between real and fake states using Generative Adversarial Networks (GANs) [4], as well as approaches based on autoencoders [5–7] and RND-based methods [8]. Uncertainty-based methods encourage exploration by rewarding agents for uncertainty in the environment. These methods often rely on techniques like Bayesian inference [9], model ensembles [10], or information-theoretic approaches [11]. On the other hand, rewarding diverse behaviors treats exploration as the primary objective, helping agents overcome sparse rewards by encouraging exploration. Notable examples include Novelty Search [12], DIAYN [13], and SI [14]. However, both rewarding novel states and rewarding diverse behaviors face challenges in the meta-RL setting. Novel state reward methods struggle with meta-RL's need for generalization across tasks. As different tasks may have distinct transition dynamics, state prediction becomes computationally expensive. Furthermore, blind exploration of new states reduces the data efficiency of meta-RL algorithms. Rewarding diverse behaviors, while promoting exploration, often leads to overly random policies that are hard to adapt to specific tasks.

Some meta-RL methods take advantage of the unique characteristics of meta-learning to address challenges in sparse reward environments. These approaches can be grouped into three main categories: sparse reward densification, task commonality representation reuse, and improving meta-policy exploration through intrinsic rewards. Sparse reward densification methods aim to relabel sparse rewards before using them for meta-policy training. This can be achieved through pre-trained reward models [15] or techniques like HTR [16]. Task commonality representation reuse methods focus on identifying shared features across tasks, such as common patterns in task behaviors [17] or state-reward relationships [18]. These methods enable knowledge sharing across tasks. Some approaches also extract reusable skills from offline datasets, combining them with the meta-policy to improve task adaptation [19]. Methods that enhance meta-policy exploration use intrinsic rewards to encourage exploration during meta-training. Examples of such methods include MetaCURE [20] and Hyper-X [21]. However, while these methods improve performance in sparse reward settings, they often rely on complex network architectures or training processes, increasing the computational burden. This is particularly challenging in meta-RL, which already requires training on multi-task datasets.

In recent years, imitation learning (IL) has gained popularity as a solution to challenges in sparse reward environments. In standard RL, one common approach involves integrating demonstration data into the experience replay buffer, where samples from both demonstrations and agent experiences are mixed for replay [22, 23]. This allows IL to inform training while incorporating new agent-generated experiences. Another method involves directly training on demonstration data. For instance, [24] regularizes the policy loss using behavior cloning (BC) with expert data. DAPG [25] introduces an online fine-tuning algorithm that combines policy gradients with BC, and LOGO [26] employs a two-step guidance approach where demonstration data steers policy learning during the initial training phase. In meta-RL, direct IL on demonstration data is more common. This is primarily due to the challenges of collecting sufficient task-specific demonstration data to populate individual task buffers, which can be costly and data-inefficient. Direct imitation, on the other hand, requires fewer samples and allows for easier control over the degree of imitation. For instance, MIL [27] extends MAML to learn from raw pixel inputs and uses IL to guide the inner-loop updates. WTL [28] improves MAML's inner-loop optimization by performing IL on demo-trial pairs, while GMPS [29] enhances MAML's outer-loop optimization using expert demonstration data through supervised learning (SL). EMRLD [30] combines RL and IL objectives to refine MAML's inner-loop optimization for meta-policy training.

These demonstration-augmented RL and meta-RL methods typically rely on direct IL from demonstration data. Their performance is heavily influenced by the quality of the demonstration data, often assuming the data to be optimal. However, in real-world scenarios, the policies behind the demonstrations are frequently suboptimal, which limits the ability of these methods to learn high-performing meta-policies from real-world data. For example, demonstration trajectories produced by a technician operating a robotic arm may reflect personal preferences or operational habits, rather than being mathematically optimal. While such demonstrations may not represent the best possible trajectories, they can still offer valuable insights for sparse reward environments. In practice, the success of meta-RL in real-world tasks depends largely on its ability to handle data generated by non-expert policies. Therefore, finding efficient methods to leverage a small amount of potentially suboptimal data to improve the learning and adaptation of the meta-policy is crucial.

Recent work has sought to balance the use of demonstration data with exploration data collected through RL. For example, wPC [31] introduces a weight factor that adjusts the reliance on demonstration data based on its advantage. Imitation occurs only when the advantage of the demonstration is positive; otherwise, the method relies entirely on exploration data for updates. This approach allows the agent to selectively use demonstrations, improving task performance based on the quality of the available demonstration data. On a different note, MAML learns the meta-policy by adapting to individual tasks but can struggle with divergent task-specific update directions. To address this, NoRML [32] proposes adding a learnable parameter offset to the meta-policy network to enable more targeted exploration. By decoupling the meta-policy parameters from those fine-tuned for individual tasks, this method enhances MAML's performance in environments with varied transition dynamics.

This paper proposes a meta-RL algorithm, Demonstrations and Offset Augmented Meta-Reinforcement Learning (DOAMRL), designed to effectively utilize both optimal and suboptimal demonstration data in sparse reward environments. Building upon EMRLD [30], DOAMRL integrates dynamically weighted IL, a learnable parameter offset, and a novel demonstration data enhancement module. This approach combines the strengths of RL and IL to improve meta-policy exploration and guide meta-policy learning efficiently in sparse reward settings. DOAMRL follows a two-step structure similar to MAML [1]: (a) In the task adaptation phase, the current meta-policy is applied to each task and adapted based on the collected data to derive a task-specific policy. (b) In the meta-policy optimization phase, the task-specific policies are applied to the tasks to which they have been adapted. The samples collected from these tasks are then used to optimize the meta-policy that guided the adaptation process. Since algorithms like MAML achieve rapid learning primarily through feature reuse rather than fast adaptation [33], it is essential for the meta-policy to capture shared structures across tasks. To address this, DOAMRL introduces a learnable parameter offset during both the adaptation and meta-optimization phases, inspired by [34]. This offset is only optimized in the meta-optimization phase, in line with [32], and serves to promote more targeted exploration. It promotes more targeted exploration, calibrates the optimization direction for the meta-policy, and ensures more stable updates. In addition, DOAMRL features a novel demonstration enhancement module that iteratively improves the quality of demonstration data for each task. By combining RL, dynamically weighted IL, offset calibration, and demonstration enhancement within the two-step MAML framework, DOAMRL facilitates rapid adaptation and enhances performance in sparse reward environments, as illustrated in Fig. 1.

The main contributions of this paper are as follows:

1. A novel meta-RL algorithm, DOAMRL, is proposed, which integrates RL and dynamically weighted IL objectives within the inner loop of MAML. Unlike other MAML-based methods augmented with demonstrations, DOAMRL combines the exploration strengths of RL with the efficiency of IL. The learnable IL weights enable the agent to adapt its reliance on demonstrations at different training stages, effectively handling variations in demonstration quality.
2. Compared to previous work, DOAMRL offers a deeper exploration of the mechanisms influencing meta-RL. Specifically, we introduce a meta-optimized parameter offset that facilitates more targeted exploration. This offset calibrates the optimization direction of the meta-policy, promoting faster and more stable updates.
3. DOAMRL introduces a novel approach to enhance low-quality demonstration data. By periodically applying an enhanced meta-policy to generate high-return trajectories, the quality of the demonstration data is progressively refined, thereby further improving the meta-policy's performance.
4. Extensive experimental results in three sparse reward environments-Point2D Navigation, TwoWheeled Locomotion, and Half Forward Backward-demonstrate that DOAMRL consistently outperforms previous methods, both with optimal and suboptimal demonstrations. Notably, when using non-expert demonstrations, DOAMRL achieves at least a 29% performance improvement over state-of-the-art methods in all three environments. Additionally, we conduct a comprehensive analysis to further validate the effectiveness and computational efficiency of the proposed approach.

The remainder of this paper is arranged as follows. Section Related works introduces meta-RL, RL with demonstrations, and meta-RL with demonstrations. Section Preliminaries presents preliminaries about model-free RL and MAML on model-free parameterized policy gradient meta-RL. Section Methodology details the DOAMRL framework and a rationale analysis. Section Experiments provides comparative experiments and performance analysis in sparse reward environments. Section Conclusion concludes this work.

## Related works

Recently, deep learning has made significant advancements, demonstrating its remarkable ability to learn representations across various domains, including computer vision [35, 36], natural language processing [37], time series classification [38], and human activity recognition [39]. Meta-learning algorithms, often based on deep learning models, aim to develop a procedure that can learn to solve new tasks effi-

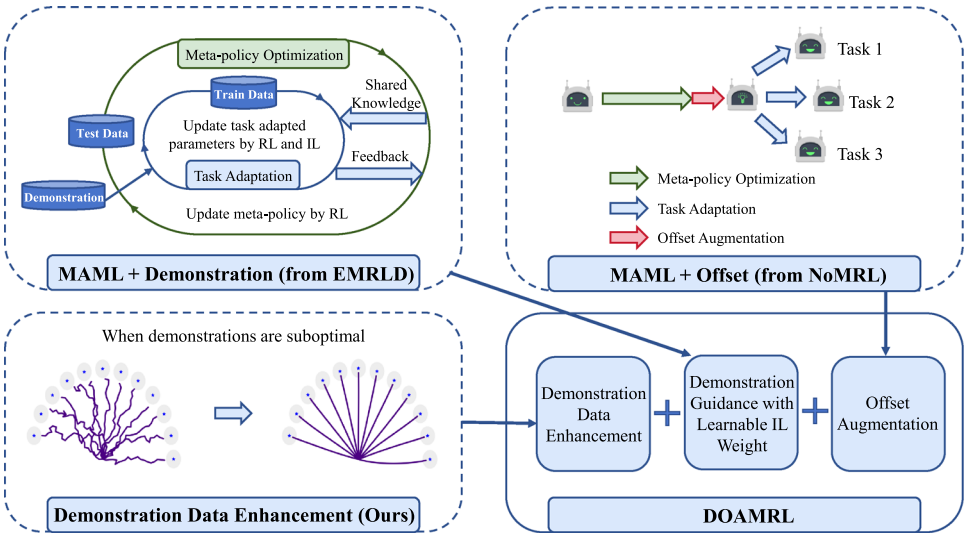



**Fig. 1** Main Contributions of DOAMRL. We integrate reinforcement learning (RL) and imitation learning (IL) objectives in the inner loop of MAML and dynamically learn the weight of the imitation learning objective. Additionally, we introduce a meta-optimized parameter offset to calibrate the update direction of the meta-policy. When provided demonstration data is of low quality, we employ a demonstration data enhancement module to augment the demonstrations, which helps train a meta-policy that performs better in sparse reward environments

ciently and effectively. They have demonstrated competitive performance across various domains, such as few-shot learning [40–43] and text classification [44]. In the context of RL, meta-RL [45], specifically model-free meta-RL, is particularly popular and widely applied. Previous model-free meta-RL algorithms can be classified as recursive-based [45–48], gradient-based [1, 47, 49], or a combination of both [50, 51].

MAML [1] is a widely utilized model-free parameterized policy gradient meta-RL method [52]. Despite this, these meta-RL variants may encounter challenges such as inaccurate gradient estimation and insufficient exploration. To address these issues, researchers have proposed a series of algorithms, including ProMP [53] and DiCE [54], to improve gradient estimation efficiency. Meanwhile, E-MAML [50, 55] and MAESN [49] were designed to enhance exploration in meta-RL. Sparse reward environments pose a challenge to meta-RL. HTR [56] attempted to overcome exploration difficulties in sparse reward environments by relabeling experience during meta-training, but its application was limited to goal-reaching tasks.

Demonstrations play a crucial role in providing task-specific domain knowledge to agents, thus facilitating learning. Existing research has explored various approaches to efficiently integrate demonstration and sampled data into an experience replay buffer [22, 23]. For instance, Nair et al. [24] incorporated expert data to introduce an auxiliary loss for RL, enhancing learning effectiveness. Another study, POfD [57], focused on training a discriminator to align generated data as closely as possible with expert data to ensure data quality during learning. DAPG [25] introduced an online fine-tuning algorithm that jointly utilized policy gradient and behavior cloning (BC), a fundamental class of algorithms in imitation learning, to enhance learning efficiently. More recently, the LOGO [26] method proposed a two-step guidance strategy, utilizing demonstration data to guide policy during the early stages of learning and support agent learning.

Existing research has demonstrated the efficiency of integrating demonstrations into meta-RL to enhance performance. Meta-imitation learning [27] extended MAML by incorporating imitation learning from expert video demonstrations. WTL [28] utilized demonstrations to generate an exploration algorithm and address tasks using a combination of exploration and demonstration data. GMPS [29] extended MAML by integrating supervised learning to optimize meta-policy using expert demonstration data. EMRLD [30] extended MAML by jointly utilizing RL and supervised learning on offline data to obtain a meta-policy that exhibited monotonic performance improvements, with requirements on offline data quality.

However, challenges remain in applying demonstrations to parameterized policy gradient meta-RL. Most existing approaches rely on expert policies or high-quality demonstration data, which limits their potential to outperform expert policies. Moreover, these algorithms often struggle when dealing with suboptimal demonstrations in real-world scenarios. This paper addresses these challenges by integrating RL and imitation learning loss in task adaptation to facilitate policy gradient computation. The weight of imitation learning is learned in the meta-optimization stage, allowing our method to leverage rewards and rapidly adapt to new tasks effectively. Moreover, we introduce a meta-optimized parameter offset to the initial MAML parameters, enhancing the meta-policy's expressive ability across multiple tasks and improving sampling efficiency and exploration. Additionally, we propose a simple iterative method to enhance demonstration data, effectively addressing the challenges posed by suboptimal demonstrations. Experimental results validate the superiority of our approach over prior work in sparse reward settings.

## Preliminaries

This section briefly introduces the two components of model-free parameterized policy gradient meta-RL: model-free RL and MAML.

### Model-free RL

We investigate the problem of an agent taking a series of actions in a stochastic environment to maximize its cumulative reward. This process can be represented using a Markov Decision Process (MDP) [58]. A MDP comprises five key elements: a finite set of states $\mathcal{S}$, a finite set of actions $\mathcal{A}$, a reward function $\mathcal{R}: \mathcal{S} \times \mathcal{A} \rightarrow [R_{min}, R_{max}]$, a transition function $\mathcal{P}: \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$, where $\Delta(\mathcal{S})$ denotes the set of probability distributions over $\mathcal{S}$, and $\gamma \in [0, 1)$ denotes a horizon discount factor. We use the following standard definitions of the value function $V^{\pi}$ of policy $\pi$, the state-action value function $Q^{\pi}$, and the advantage function $A^{\pi}$:

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid s_0 = s\right], \tag{1}$$

$$Q^{\pi}(s, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid s_0 = s, a_0 = a\right], \tag{2}$$

$$A^{\pi}(s, a) = Q^{\pi}(s, a) - V^{\pi}(s). \tag{3}$$

Our objective is to identify an optimal policy that maximizes the expected return of the policy $\pi_\theta$ in a given environment. The objective function for policy learning is defined as $\mathcal{L}(\theta) = \mathbb{E}_{s_0}[V^{\pi_\theta}(s_0)]$, where $s_0$ denotes the initial state. In model-free RL, policy gradient methods are commonly used to optimize a policy. By computing the gradient of the objective function, $\nabla_\theta \mathcal{L}(\theta)$, with respect to the policy, we can apply gradient ascent to maximize the objective and obtain the optimal policy. According to the policy gradient theorem [59], this gradient can be approximated as

$$\nabla_\theta \mathcal{L}(\theta) = -\mathop{\mathbb{E}}_{(s_t, a_t) \sim \pi_\theta}\left[A^{\pi_\theta}(s_t, a_t) \nabla_\theta \log \pi_\theta(a_t \mid s_t)\right], \tag{4}$$

where the advantage function $A^{\pi_\theta}(s_t, a_t)$ can be estimated using Generalized Advantage Estimation (GAE) [60] or alternative methods. A proof of Eq. (4) is provided in Appendix A.

### MAML on model-free parameterized policy gradient meta-RL

Meta-learning aims to learn a meta-policy capable of swift adaptation to specific tasks. Assuming that both training and testing tasks are from the same distribution, meta-learning optimizes the meta-policy for learning the underlying commonalities among tasks. This meta-policy should exhibit superior performance upon adaptation with a small amount of specific data.

As the most popular model-free meta-learning framework, MAML [1] addresses this challenge through a gradient-based approach, and its optimization process consists of two steps:

1. Outer loop: Optimize the initial neural network parameters to facilitate rapid adaptation to new tasks.
2. Inner loop: Given the parameters initialized in the outer loop, a series of gradient updates are executed using $k$ labeled data (termed $k$-shot learning) provided for adaptation in each task.

Considering a task distribution $p(\mathcal{T})$ and task-specific loss functions $\mathcal{L}_{\mathcal{T}_i}$ for each task $i$, MAML seeks an optimal meta-parameter $\theta$ capable of adapting to specific tasks using only a small amount of data through a single gradient descent step. This objective can be expressed as

$$\begin{aligned} &\arg\min_{\theta} \mathop{\mathbb{E}}_{i \sim p(\mathcal{T})}\left[\mathcal{L}_{\mathcal{T}_i}(\theta_i)\right] \\ &\text{s.t. } \theta_i = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\theta), \end{aligned} \tag{5}$$

where $\alpha$ is a hyperparameter. MAML's versatile formulation allows it to be applicable across diverse problem domains. When applying MAML to model-free RL, $\theta$ represents the meta-policy parameters $\pi_\theta$. If the gradient update rule in Eq. (5) is applied to model-free RL, with Monte-Carlo estimation replacing the expectation in Eq. (4), the adapted policy $\pi_{\theta_i}$ can be formulated as

$$\begin{aligned} \theta_i &= \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}\left(\theta, \mathcal{D}_i^{\text{train}}\right) \\ &= \theta + \alpha \sum_{(s_t, a_t) \in \mathcal{D}_i^{\text{train}}} A^{\pi_\theta}(s_t, a_t) \nabla_\theta \log \pi_\theta(a_t \mid s_t), \end{aligned} \tag{6}$$

where $\mathcal{D}_i^{\text{train}} = \left\{(s_1, a_1, r_1, \ldots, s_{H+1})_k : k = 1, \ldots, K\right\}$ is derived from $K$ trajectories generated by the meta-policy $\pi_\theta$ on task $i$. Throughout meta-training, trajectories generated by the fine-tuned policy $\pi_{\theta_i}$ for task $i$ contribute to computing the meta-objective. Through optimization of the meta-policy parameters $\theta$, MAML minimizes the expected loss across all tasks after adaptation. Furthermore, let $\mathcal{D}_i^{\text{test}}$ denote the transitions sampled from the adapted policy $\pi_{\theta_i}$, and the meta-policy updating process can be expressed as

$$\theta \leftarrow \theta - \beta \sum_{i \sim \mathcal{T}} \nabla_\theta \mathcal{L}_{\mathcal{T}_i}\left(\theta_i, \mathcal{D}_i^{\text{test}}\right), \tag{7}$$

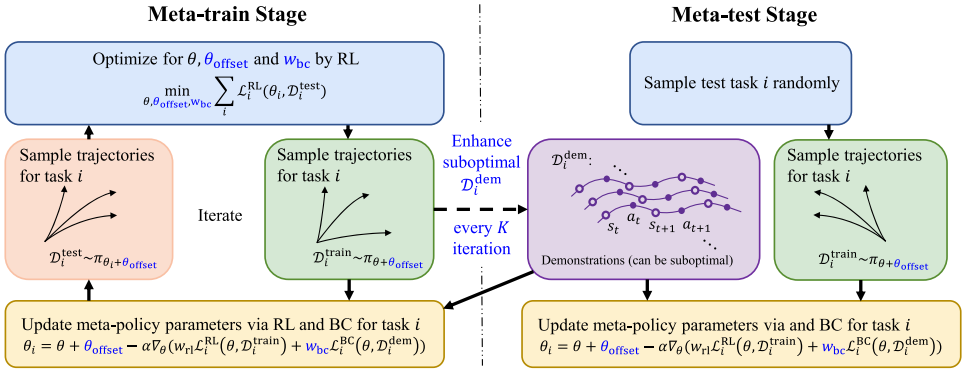



**Fig. 2** Overview of Demonstrations and Offset Augmented Meta-RL. This figure depicts the DOAMRL framework, which can be generalized across multiple tasks, and highlights the differences between EMRLD and DOAMRL. During the task adaptation stage, DOAMRL combines RL and imitation learning loss, allowing for the utilization of potentially suboptimal demonstration data. A parameter offset is introduced and updated during the meta-optimization stage. Additionally, the learnable imitation learning weight is optimized during the same stage. If demonstration data in $\mathcal{D}_i^{\text{dem}}$ is suboptimal, it is augmented every $K$ iteration. When adapted to a new task, policy gradients described in the task adaptation stage are utilized for rapid adaptation

where $\beta$ is a hyperparameter and

$$\mathcal{L}_{\mathcal{T}_i}\left(\theta_i, \mathcal{D}_i^{\text{test}}\right) = \mathcal{L}_{\mathcal{T}_i}\left(\theta - \alpha\nabla_\theta \mathcal{L}_{\mathcal{T}_i}\left(\theta, \mathcal{D}_i^{\text{train}}\right), \mathcal{D}_i^{\text{test}}\right). \tag{8}$$

The gradient of $\mathcal{L}_{\mathcal{T}_i}\left(\theta_i, \mathcal{D}_i^{\text{test}}\right)$ can be approximated using the policy gradient algorithm similar to Eq. (6).

## Methodology

This section begins with an overview of DOAMRL, followed by a detailed discussion of how demonstration data enhances its performance. It then explains the integration of the parameter offset for calibrating the meta-optimization direction and the iterative process for updating demonstration data. Finally, we analyze the computational complexity and the underlying rationale of DOAMRL.

### Overview

Our method consists of three key components: 1) Collecting demonstration data, which includes one expert and one non-expert trajectory for each training task; 2) Training a meta-policy to capture shared features across tasks; and 3) Meta-testing, where the learned policy is evaluated on new tasks. An overview of the DOAMRL method is illustrated in Fig. 2.

During the demonstration data collection phase, we utilize the majority of the demonstration data from EMRLD [30], while generating non-expert trajectories using a partially trained Soft Actor-Critic (SAC) [61]. This approach validates DOAMRL's effectiveness in handling suboptimal demonstration data, as discussed in Section Experimental setup. During meta-training, we alternate between task adaptation and meta-policy optimization, similar to MAML [1]. In the task adaptation phase, we combine RL with a dynamically weighted IL objective, allowing the meta-policy to utilize both expert and non-expert demonstrations better. This hybrid objective leverages RL's exploration advantages and IL's efficiency, enabling effective use of demonstration data at various stages of training. In the meta-policy optimization phase, we follow the approach from EMRLD [30], using TRPO [62] to optimize the meta-policy, ensuring stable updates with theoretical guarantees. We also introduce a learnable parameter offset to promote more targeted exploration of the meta-policy. To address the challenges posed by low-quality demonstration data, we incorporate a demonstration enhancement module, which iteratively improves the data quality by periodically replacing it with high-return trajectories generated by an increasingly effective meta-policy. Finally, during meta-testing, the meta-policy adapts to new tasks, generating task-specific policies based on the collected trajectories, which are refined through gradient descent.

### Demonstrations augmented meta-RL

Learning optimal control policies via RL in sparse reward environments has long been challenging. In such scenarios, agents typically only receive non-zero rewards upon task completion or nearing completion. During the early stages of training, the trajectories generated by the learning policy often need to provide valuable information. As gradients become increasingly homogeneous across various directions, the policy makes meaningless adjustments, significantly impacting sample efficiency and performance improvement.

In standard RL algorithms, demonstrations have been used widely to assist learning via supervised learning (SL) or imitation learning to tackle the challenges posed by sparse rewards. These demonstrations may come from expert policies [25] or non-expert data [26, 53]. The intuitive idea is that even if demonstration data lacks explicit reward information, it can still effectively guide the agent to explore those state-action regions likely to yield non-zero rewards.

Most gradient-based model-free meta-RL algorithms learn meta-parameters from scratch. The optimization of these algorithms heavily relies on rewards obtained from trajectories, acquired by rolling out the meta-policy and task-specific policies. However, because reward feedback is applied to both meta-parameter adaptation and updating, the learning process in meta-RL algorithms in sparse reward environments is more challenging than in standard RL algorithms.

The progress made in standard RL through the utilization of demonstration data offers valuable insights for employing a similar approach in meta-RL. Intuitively, demonstration data can positively impact meta-policy adaptation and meta-parameter optimization. Firstly, it assists the agent in adapting to new tasks via one or more gradient steps, which is particularly crucial in sparse reward environments. Secondly, it can facilitate meta-parameter updating by guiding task-specific policies to generate trajectories more likely to reach regions with non-zero rewards in sparse reward environments.

In the context of meta-RL with demonstrations, let $\mathcal{D}_i^{\text{dem}}$ denote the demonstration data corresponding to each task $i$, which can be suboptimal. We incorporate an auxiliary loss corresponding to BC to facilitate the gradient update. Specifically, we define the behavior cloning (BC) loss function for task $i$ as $\mathcal{L}_{\mathcal{T}_i}^{\text{BC}}\left(\theta, \mathcal{D}_i^{\text{dem}}\right) = -\sum_{(s,a)\in\mathcal{D}_i^{\text{dem}}} \log \pi_\theta(a \mid s)$. Formally, let $\mathcal{D}_i^{\text{train}}$ represent the training data obtained by rolling out the meta-policy on task $i$, and let $\theta_k$ represent the meta-policy parameters at the start of the $k$-th iteration. The adapted parameters $\theta_{k,i}$ for task $i$ can be expressed as

$$\begin{aligned}\theta_{k,i} &= \theta_k - \alpha(w_{\text{rl}}\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(\theta, \mathcal{D}_i^{\text{train}}) + w_{\text{bc}}\nabla_\theta \mathcal{L}_{\mathcal{T}_i}^{\text{BC}}(\theta, \mathcal{D}_i^{\text{dem}})) \\ &= \theta_k + \alpha\left(w_{\text{rl}} \sum_{(s,a)\in\mathcal{D}_i^{\text{train}}} \nabla_\theta \log \pi_\theta(a \mid s) A^{\pi_\theta}(s,a) \right. \\ &\quad \left. + w_{\text{bc}} \sum_{(s,a)\in\mathcal{D}_i^{\text{dem}}} \nabla_\theta \log \pi_\theta(a \mid s)\right)\Bigg|_{\theta=\theta_k}, \end{aligned} \tag{9}$$

where $w_{\text{rl}}$ and $\alpha$ are hyperparameters. Eq. (9) resembles Eq. (2) in [30], allowing us to leverage the strengths of RL and imitation learning. Our further goal is to ensure the agent is more appropriately guided by demonstrations based on the training progress and the quality of the demonstration data. Therefore, in our method, $w_{\text{bc}}$ is no longer treated as a hyperparameter but as a parameter that can be learned through gradient descent. $w_{\text{bc}}$ is updated using the Adam optimizer [63] during the meta-optimization phase as

$$\begin{aligned} w_{\text{bc}} &= w_{\text{bc}} - \beta_{w_{\text{bc}}} \sum_{i\sim\mathcal{T}} \nabla_{w_{\text{bc}}} \mathcal{L}_{\mathcal{T}_i}\left(\theta_i, \mathcal{D}_i^{\text{test}}\right) \\ &= w_{\text{bc}} - \beta_{w_{\text{bc}}} \sum_{i\sim\mathcal{T}} \sum_{(s,a)\in\mathcal{D}_i^{\text{test}}} \nabla_{w_{\text{bc}}} \\ &\quad \log \pi_{\theta_{k,i}}(a \mid s) A^{\pi_{\theta_{k,i}}}(s,a), \end{aligned} \tag{10}$$

where $\beta_{w_{\text{bc}}}$ is a hyperparameter indicating the learning rate of $w_{\text{bc}}$. We experimentally assess the impact of different initial values for $w_{\text{bc}}$ and different values for $w_{\text{rl}}$ on the meta-policy training in Section Experiments.

Next, we explore the utilization of demonstrations during the meta-policy optimization phase. Some research relies on the SL using demonstrations to update the meta-policy parameters [27, 29]. However, relying solely on SL when demonstrations are obtained from non-expert policies may restrict the meta-policy's performance. This contradicts our intention to fully exploit the potential of suboptimal demonstration data in sparse reward environments. Existing studies have also experimented with the simultaneous use of RL and SL, but have yet to yield superior performance [30]. Therefore, we update the meta-policy parameters using the RL gradient as

$$\begin{aligned} \theta_{k+1} &= \theta_k - \beta \sum_{i\sim\mathcal{T}} \nabla_\theta \mathcal{L}_{\mathcal{T}_i}\left(\theta_i, \mathcal{D}_i^{\text{test}}\right) \\ &= \theta_k - \beta \sum_{i\sim\mathcal{T}} \sum_{(s,a)\in\mathcal{D}_i^{\text{test}}} \nabla_{\theta_{k,i}} \log \pi_{\theta_{k,i}}(a \mid s) A^{\pi_{\theta_{k,i}}}(s,a), \end{aligned} \tag{11}$$

where $\mathcal{D}_i^{\text{test}}$ denotes the trajectories generated by the policy adapted to task $i$, and $\beta$ represents a hyperparameter. Given the higher likelihood of $\mathcal{D}_i^{\text{test}}$ reaching the non-zero rewards region of the state-action space, using it for RL gradient updates is reasonable.

For the meta-policy optimizer, we employ the trust-region policy optimization (TRPO) [62] for its advantage in maintaining stable policy updates. Formally, let us denote $\theta_{k+1}$ as the parameters of the meta-policy at the $(k+1)$-th iteration, and $\theta_k$ as the parameters at the $k$-th iteration. $\theta_{k+1}$ can be expressed as

$$\theta_{k+1} = \theta_k + \sqrt{\frac{2\delta}{g^T H^{-1} g}} H^{-1} g, \tag{12}$$

where $g$ represents the gradient of the expected difference between $\pi_{\theta_{k+1}}$ and $\pi_{\theta_k}$, and it can be calculated as

$$g = \nabla_{\theta_{k+1}} \mathop{\mathbb{E}}_{\substack{s\sim\nu^{\pi_{\theta_k}} \\ a\sim\pi_{\theta_k}(\cdot|s)}} \left[ \frac{\pi_{\theta_{k+1}}(a \mid s)}{\pi_{\theta_k}(a \mid s)} A^{\pi_{\theta_k}}(s,a) \right], \tag{13}$$

where $\nu^{\pi_{\theta_k}}$ represents the state visitation distribution of the policy $\pi_{\theta_k}$. $H$ represents the Hessian matrix of the average Kullback–Leibler (KL) divergence between $\pi_{\theta_k}$ and $\pi_{\theta_{k+1}}$, calculated as

$$H = \mathbf{H}\left[ \mathop{\mathbb{E}}_{s\sim\nu^{\pi_{\theta_k}}} \left[ \mathcal{D}_{KL}\left(\pi_{\theta_k}(\cdot \mid s), \pi_{\theta_{k+1}}(\cdot \mid s)\right) \right] \right]. \tag{14}$$

Subsequently, the meta-policy parameters can be updated using conjugate gradient and linear search, omitted for brevity.

## Offset learning

To overcome the limitations of a single gradient update step in adapting the exploratory meta-policy to a task-specific policy, we propose a practical approach to calibrating the optimization direction of the meta-policy. This approach aims to enhance the expressive capacity of the meta-policy and improve optimization efficiency. Specifically, inspired by the work [32], during the adaptation of the meta-policy to task $i$, we incorporate a meta-optimized parameter offset, denoted as $\theta_{\text{offset}}$, into the meta-policy parameters $\theta_k$ as

$$\theta_{k,i} = \theta_k + \theta_{\text{offset}} + \alpha \left( w_{\text{rl}} \sum_{(s,a)\in\mathcal{D}_i^{\text{train}}} \nabla_\theta \log \pi_\theta(a \mid s) A^{\pi_\theta}(s,a) + w_{\text{bc}} \sum_{(s,a)\in\mathcal{D}_i^{\text{dem}}} \nabla_\theta \log \pi_\theta(a \mid s) \right)\Big|_{\theta=\theta_k}. \tag{15}$$

As the parameter offset $\theta_{\text{offset}}$ is shared across all tasks, it does not influence the adaptation for a particular task. Task adaptation now relies on trajectories obtained by rolling out the policy $\pi_{\theta+\theta_{\text{offset}}}$, and the policy gradient is computed by differentiating regarding the meta-policy parameters $\theta$. The parameter offset is optimized jointly with the meta-policy as

$$\begin{aligned} \theta_{\text{offset}} &= \theta_{\text{offset}} - \beta \sum_{i\sim\mathcal{T}} \nabla_{\theta_{\text{offset}}} \mathcal{L}_{\mathcal{T}_i}\left(\theta_i, \mathcal{D}_i^{\text{test}}\right) \\ &= \theta_{\text{offset}} - \beta \sum_{i\sim\mathcal{T}} \sum_{(s,a)\in\mathcal{D}_i^{\text{test}}} \nabla_{\theta_{\text{offset}}} \log \pi_{\theta_{k,i}}(a \mid s) A^{\pi_{\theta_{k,i}}}(s,a). \end{aligned} \tag{16}$$

**Algorithm 1:** Demonstration Data Enhancement

**Input**: Original suboptimal demonstration $\tau_{origin}$ with return $R_{origin}$, replacement iteration threshold $K_{thres}$, replacement period $K$, and demonstration replacement return threshold $R_{thres}$

Initialize the enhanced demonstration data $\tau_{enhanced}$ with $\tau_{origin}$.
Initialize the enhanced demonstration data return $R_{enhanced}$ with $R_{origin}$.
**for** *iteration* $k = 0, 1, \ldots$ **do**
  **if** $k$ **mod** $K = 0$ *and* $k > K_{thres}$ **then**
    **for** $i \in \mathcal{T}$ **do**
      Sample trajectories using $\pi_{\theta_{k,i}+\theta_{\text{offset}}}$ on task $i$ and choose the trajectory $\tau_{candidate}$ with the highest return $R_{candidate}$.
      **if** $R_{candidate} > R_{thres}$ *and* $R_{candidate} > R_{enhanced}$ **then**
        Replace the old demonstration trajectory $\tau_{enhanced}$ with the trajectory $\tau_{candidate}$ for task $i$.
        Set $R_{candidate}$ to $R_{enhanced}$ for task $i$.
      **end**
    **end**
  **end**
**end**

**Output**: The enhanced demonstration data $\tau_{enhanced}$

## Demonstration data enhancement

**Assumption 1** *Meta-policies trained using demonstrations with higher average returns perform better than those trained using demonstrations with lower average returns.*

Based on Assumption 1, we propose a straightforward method to enhance the quality of demonstrations during the meta-policy training process, mainly when suboptimal demonstrations are utilized. The proposed method for enhancing demonstration data is outlined in Algorithm 1.

Specifically, there is a gradual performance improvement throughout the training process across different training tasks. Upon reaching the iteration threshold $K_{thres}$, we will roll out the task-specific policy for each training task in every $K$ iterations to gather a set of trajectories, which will be used to update the demonstration data. Establishing a $K_{thres}$ is instrumental in guaranteeing that trajectories generated by the meta-policy can address the training tasks. This significantly enhances the prospects of producing demonstration data of superior quality and meaningful guidance. The $K_{thres}$ value is estimated based on the average return curves during training. Selecting the value of $K$ is a tradeoff. If $K$ is too small, it can affect the stability of the meta-policy training, whereas if $K$ is too large, newly generated high-quality demonstration data may not be utilized timely to guide the training of the meta-policy, undermining its performance improvement. Setting $K$ to 20 in our experiments enables the meta-policy to achieve stable updates and perform well.

Next, we select the trajectory $\tau$ with the highest return from the collection of trajectories. Assume that the return $R_{candidate}$ of this trajectory exceeds both the return $R_{enhanced}$

**Algorithm 2:** Demonstrations and Offset Augmented Meta-RL (DOAMRL)

**Input**: The distribution over $N$ tasks $p(\mathcal{T})$, demonstration data $\mathcal{D}_i^{\text{dem}}$ for task $i = 1, \ldots, N$, adaptation learning rate $\alpha$, meta learning rate $\beta$, $w_{\text{bc}}$ learning rate $\beta_{w_{\text{bc}}}$

Randomly initialize the meta-parameter $\theta_0$
Initialize the parameters in $\theta_{\text{offset}}$ to zero.
**for** *iteration* $k = 0, 1, \ldots$ **do**
  **for** $i \in \mathcal{T}$ **do**
    Sample trajectories using $\pi_{\theta_k + \theta_{\text{offset}}}$ on task $i$ and store all transitions $(s_t, a_t, r_t, s_{t+1})$ as the training data set $\mathcal{D}_i^{\text{train}}$.
    Compute adapted parameters $\theta_{k,i}$ using $\mathcal{D}_i^{\text{train}}$ following Eq. (15).
    Sample trajectories using $\pi_{\theta_{k,i} + \theta_{\text{offset}}}$ on task $i$ as $\mathcal{D}_i^{\text{test}}$.
    **if** $\mathcal{D}_i^{dem}$ *is suboptimal* **then**
      Update $\mathcal{D}_i^{\text{dem}}$ with Algorithm 1.
    **end**
  **end**
  Update $w_{\text{bc}}$ using Adam optimizer with Eq. (10).
  Update $\theta_k$ using all $\mathcal{D}_i^{\text{test}}$ with Eq. (11).
  Update $\theta_{\text{offset}}$ using all $\mathcal{D}_i^{\text{test}}$ with Eq. (16).
**end**

**Output**: The trained meta-policy and the trained parameter $\theta_{\text{offset}}$

of the currently used demonstration data and the replacement return threshold $R_{thres}$ we have set. In such a scenario, we replace the old demonstration trajectory with this new trajectory to guide the meta-policy training process. The $R_{thres}$ serves two purposes. Firstly, it ensures that the demonstration data enhancement module can proceed even if the original demonstration trajectory lacks precise return information. Secondly, it guarantees that the selected new trajectory is high-quality and can effectively guide the meta-policy training process.

Through these two steps, we replace the old demonstration trajectory with a newly generated high-quality one to guide the training process. This approach helps to address the challenge posed by suboptimal demonstration data, as demonstrated in Section Experimental setup.

The proposed method is summarized in Algorithm 2.

## Computational complexity analysis

The computational complexity of DOAMRL is determined by both the inner and outer loops of the algorithm.

First, we discuss the complexity of the inner loop. In the inner loop, DOAMRL performs gradient updates on each task to quickly adapt the meta-policy. Let $N$ represent the number of tasks, $B = |\mathcal{D}_i^{\text{train}}|$ denote the size of each task's training set, and $K$ denote the number of gradient update steps per task. For DOAMRL, $K$ is equal to 1. Therefore, the complexity of the inner loop for a single task is $O(B)$, as each gradient update processes the training data. Consequently, the total complexity for all tasks in the inner loop is then $O(N \cdot B)$.

Then we discuss the complexity of the outer loop. In the outer loop, DOAMRL computes the meta-gradient based on the performance of the adapted meta-policy using a separate test set. The number of tasks in the outer loop, denotes as $N$, is equal to the number of tasks in the inner loop. Let $M$ represent the size of each task's test set, defined as $M = |\mathcal{D}_i^{\text{test}}|$, which is sampled during adaptation. The complexity of the outer loop involves calculating the gradient over the test set, resulting in a total complexity of $O(N \cdot M)$.

Finally, we can determine the overall complexity of DOAMRL. By combining complexities of the inner and outer loop, the total computational complexity of DOAMRL is $O(N \cdot (B + M))$, where $N$ is the number of tasks in both loops, $B$ is the training set size for each task, and $M$ is the test set size for each task.

Thus, the computational complexity of DOAMRL depends on the number of tasks and the sizes of the training and test sets for each task. Notably, DOAMRL reduces complexity compared to the original MAML [1], as it collects data for meta-optimizing only after a single gradient adaptation step per task. The incorporation of demonstration data and the introduction of $\theta_{\text{offset}}$ further contribute to this reduction. These modifications enable the method to focus on non-zero reward regions, which more accurately capture task characteristics, enabling learning a more effective meta-policy. Consequently, DOAMRL improves both performance and generalization, while requiring fewer adaptation steps and data samples.

## Rationale

Now, we analyze the benefits of incorporating demonstrations and the meta-optimized parameter offset into gradient-based meta-RL algorithms augmented with demonstration data. Specifically, we focus on how this integration can enhance DOAMRL's performance and sample efficiency in sparse reward environments.

Firstly, we analyze the rationale for incorporating demonstrations. Our work assumes that, during the early stages of training, actions proposed by demonstration data should outperform those generated by the meta-policy, thereby encouraging meaningful updates to the meta-policy, a concept also present in other RL algorithms guided by demonstrations [26, 30, 57]. Drawing from this assumption and referring to Theorem 3.2 in [30], if the TRPO algorithm is applied in the outer loop, minimizing the total variation distance between the adapted meta-policy and demonstration data can lead to enhanced meta-policy performance. However, as training processes, this theorem no longer holds. Therefore, we dynamically optimize the imitation learning weight at each meta-policy optimization phase, enabling the agent to utilize newly explored and improved actions, thus facilitating the enhanced performance of the meta-policy.

Subsequently, we provide a rationale for integrating a meta-optimized parameter offset. Firstly, setting all parameters in this offset to 0 would cause our proposed method to revert to a scenario without the offset, indicating that it enhances the model's capacity. Secondly, the meta-optimized parameter offset functions are akin to the residual connections in residual networks (ResNets) [64], promoting the stability of meta-policy updates. Moreover, in sparse reward environments where using rewards to guide policy optimization is challenging, the parameter offset modifies the direction of the meta-policy to the non-zero reward regions. Lastly, the parameter offset encourages the exploration of different behaviors, helping to prevent the algorithm from converging to local optima.

Finally, we demonstrate the rationale behind demonstration data enhancement. As described in Section Experimental setup, demonstrations of varying quality are collected by running the SAC [61] policy at different training stages in the corresponding environments. The return is a crucial metric that distinguishes the different training stages of the SAC policy. Thus, there is a strong correlation between the return and the quality of the demonstration data. By replacing the original low-return demonstrations with high-return trajectories generated by an improving meta-policy, we can effectively utilize Assumption 1. This approach provides more accurate and effective guidance for the meta-policy update direction in sparse reward environments, accelerating the update process and achieving better performance.

Based on the comprehensive analysis provided, incorporating demonstrations facilitates effective updates to the meta-policy, particularly during the initial training phases. At the same time, introducing a parameter offset allows for the calibration of the update direction of the meta-policy, ensuring stable policy updates and contributing to an expanded exploration. Demonstration data enhancement allows for iterative updates of the demonstration data even when only suboptimal demonstrations are provided, utilizing higher quality demonstration data to assist the meta-policy in rapidly updating in a better direction. Empirical experiments demonstrate that DOAMRL significantly enhances the performance and sample efficiency by incorporating these three orthogonal components.

## Experiments

We evaluate DOAMRL as a meta-RL algorithm and investigate its ability to learn an effective meta-policy in sparse reward environments using optimal or suboptimal demonstrations. We consider the following questions:

- Does DOAMRL outperform previous meta-RL approaches regarding performance and sample efficiency under the guidance of optimal and suboptimal demonstration?
- Can DOAMRL rapidly adapt to new tasks under the guidance of optimal and suboptimal demonstration?
- How does each component of DOAMRL impart the overall performance?
- How sensitive DOAMRL is to changes to hyperparameters?

To answer these questions, we conduct experiments on DOAMRL in simulated environments, including Point2D Navigation [1], TwoWheeled Locomotion [49], and HalfCheetah Forward Backward [65].

### Experimental setup

**Sparse Multi-task Environments:** We utilize three simulated environments, each defining specific tasks through distinct reward functions. These environments also allow generating unknown testing tasks by randomly modifying the reward functions.

- Point2D Navigation: This environment involves two-dimensional navigation, where an agent aims to reach the target position on the circumference of a semicircle with a radius of 2, using the fewest steps from the origin. The agent's actions $(a_x, a_y)$ are based on its current position $(x, y)$. The agent only receives a non-zero reward if the distance to the target location is less than a specific value. Training tasks consist of 12 fixed target positions, while testing tasks involve randomly sampled positions along the circumference of the semicircle.
- TwoWheeled Locomotion: Similar to Point2D Navigation, this is a target-reaching task with sparse rewards, but the agent is a robot with constraints on angular velocity, linear velocity, and turning radius. Training tasks involve 24 fixed target locations along the circumference of a semicircle with a radius of 2, while testing tasks involve randomly sampled positions.
- HalfCheetah Forward Backward: In this variant of the classic HalfCheetah task, the agent must move forward and backward along the $x$-axis. Rewards are provided when the agent moves a certain distance in the correct direction. Training and testing tasks remain consistent in this environment.

**Demonstration Data:** We train the algorithm by using only one demonstration for each task, where each demonstration is represented as a single trajectory with several $(s, a)$ transitions. In Point2D Navigation and TwoWheeled Locomotion, we utilize demonstration data provided in [30]. For the HalfCheetah Forward Backward task, we use the opti-

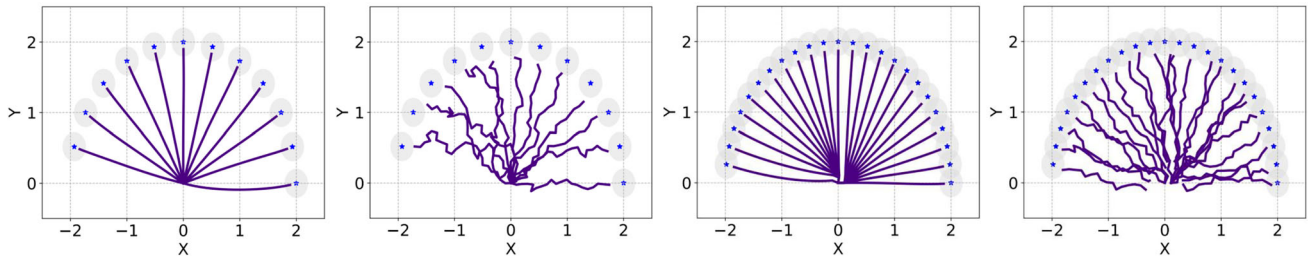



**Fig. 3** Illustration of optimal and suboptimal demonstrations of Point2D Navigation (leftmost) and TwoWheeled Locomotion (rightmost) tasks

mal data from [30] and an inexpert, partially trained SAC [61] policy to generate suboptimal demonstration data. The cumulative return of the non-expert demonstration data is only 10% of the return achieved by the SAC after training for 1 million timesteps. This scenario is realistic in real-world environments, where some demonstrations may be known to be relatively effective based on prior experience. Due to the diversity of reward functions in complex environments, accurately quantifying the gap between experiential data and optimal demonstrations is challenging. By using suboptimal demonstrations, we demonstrate the method's effectiveness and robustness in guiding meta-policy training with lower-quality experience data. The optimal and suboptimal demonstrations used in Point2D Navigation and TwoWheeled Locomotion are illustrated in Fig. 3.

**Baselines:** We compare DOAMRL with the following gradient-based meta-RL algorithms:

- MAML [1]: The original MAML algorithm for meta-RL.
- Meta-BC [27]: A variant of [27], using maximum likelihood loss during meta-optimization and adaptation phases.
- GMPS [29]: Guided meta-policy search, employing RL during adaptation and SL during meta-policy optimization.
- EMRLD [30]: Enhanced meta-RL with demonstrations integration, combining RL and SL during adaptation and RL during meta-optimization.
- PEARL [66]: Probabilistic embeddings for actor-critic meta-RL, a two-stage off-policy meta-RL method that combines task inference with the optimization of conditional policies based on task representations.
- GENTLE [67]: Generalizable task representation learning, a two-stage offline meta-RL method that utilizes task representation learning and offline meta-policy optimization over the learned representations.

**Experimental Details:** The experiments were conducted on an Ubuntu 18.04 system, equipped with an NVIDIA A100 GPU and an Intel(R) Xeon(R) Platinum 8375C CPU. The version of CUDA used was 11.2, and PyTorch 1.11.0 was employed for the experiments. Table 1 details the hyperparameters used in DOAMRL. Unless specified otherwise, these hyperparameters remain consistent with the values listed in Table 1. As our focus is on comparing the performance of DOAMRL with existing methods, rather than optimizing hyperparameters for the best possible performance, we adopt the same hyperparameter settings as those used in [30]. The values of the newly introduced parameters $K$ and $K_{thres}$, were selected through grid search. Results for Meta-BC and GMPS are obtained from the experiments reported in [30], while the EMRLD results were generating using the open-source code[1] provided by the authors. The MAML results were derived using the EMRLD project, with the BC module removed from the inner loop. Additionally, results for PEARL[2] [66] and GENTLE[3] [67] were obtained from the open-source code provided by their respective authors.

For the network implementation, we use a three-layer multi-layer perceptron (MLP) with a single hidden layer to model the meta-policy $\pi_\theta$, applying the ReLU activation function. The input dimension of the MLP varies depending on the environment's state space: 2 for Point2D Navigation, 3 for TwoWheeled Locomotion, and 20 for HalfCheetah Forward Backward. The output dimension corresponds to the action space, with dimensions of 2, 2, and 6 for the respective environments. The hidden layer contains 100 units. The network architecture for the parameter offset $\theta_{\text{offset}}$ is identical to that of the meta-policy.

## Performance evaluation

**Performance with optimal demonstration data:** Fig. 4 illustrates the training and testing performance of different algorithms utilizing optimal data. The upper part of Fig. 4 depicts the average return of different algorithms during the training process across three tasks. Compared to others, DOAMRL achieves the highest average return and demonstrates the fastest convergence rate, indicating its superior sample efficiency.

[1] https://github.com/DesikRengarajan/EMRLD.

[2] https://github.com/katerakelly/oyster.

[3] https://github.com/LAMDA-RL/GENTLE.

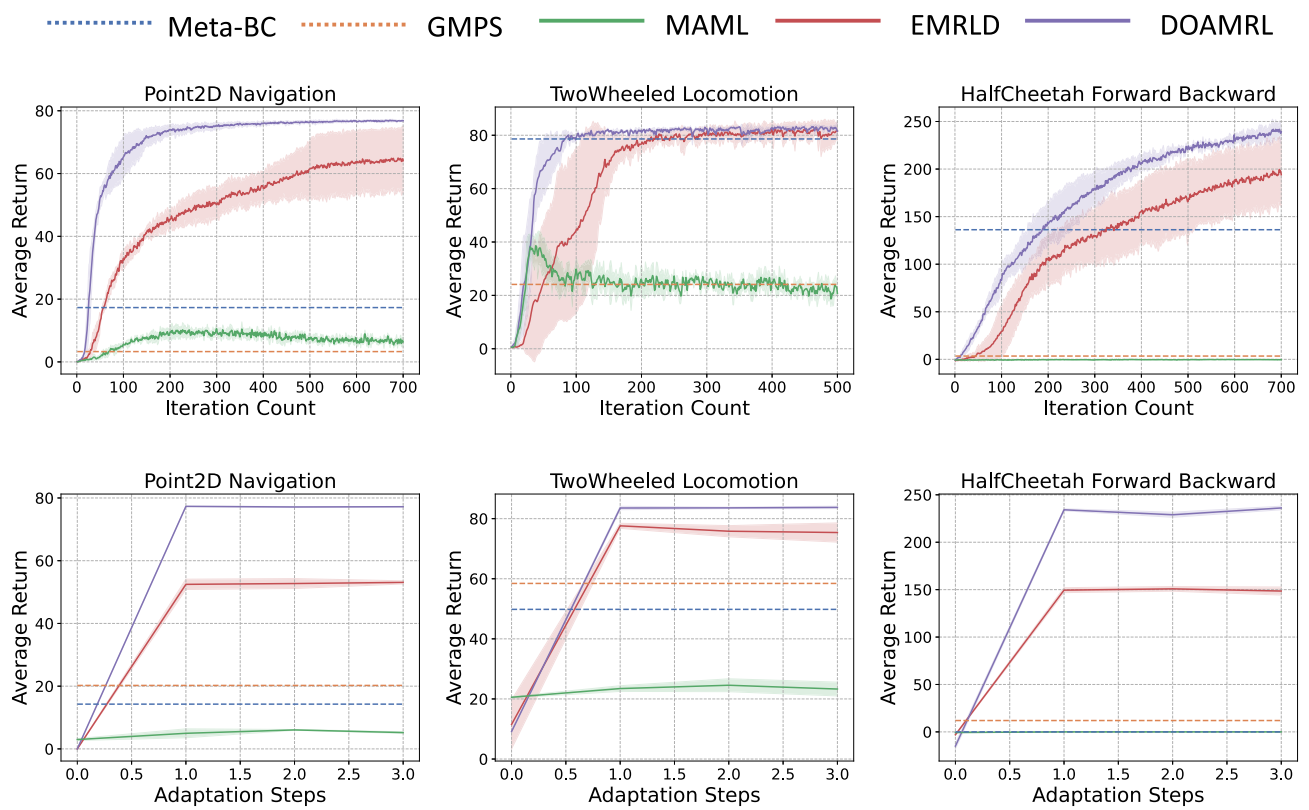



**Fig. 4** Average return curves during training (top) and testing (bottom) on Point2D Navigation, TwoWheeled Locomotion, and HalfCheetah Forward Backward with **optimal demonstrations**. The upper part shows training curves, where the solid line represents the mean value across three seeds, and the shaded region corresponds to the standard deviation. The lower part presents testing curves, where the solid line corresponds to the mean performance across all testing tasks, and the shaded region corresponds to the standard deviation

**Table 1** Hyperparameters used in DOAMRL, EMRLD, and MAML

| Hyperparameters | DOAMRL | EMRLD | MAML |
|---|---|---|---|
| Adapt Batch Size | 20 | 20 | 20 |
| Adapt Steps | 1 | 1 | 1 |
| Adaptation Learning Rate $\alpha$ | 0.01 | 0.01 | 0.01 |
| CPU Thread Number | 20 | 20 | 20 |
| GAE $\tau$ | 1 | 1 | 1 |
| $\gamma$ | 0.95 | 0.95 | 0.95 |
| Meta Learning Rate $\beta$ | TRPO | TRPO | TRPO |
| $w_{\text{bc}}$ | 1 | 1 | N/A |
| $w_{\text{rl}}$ | 0.2/1 | 0.2/1 | N/A |
| $w_{\text{bc}}$ Learning Rate | 0.0003 | N/A | N/A |
| $K$ | 20 | N/A | N/A |
| $K_{thres}$ | 40 | N/A | N/A |

The lower part of Fig. 4 presents the average return of different trained algorithms when adapting to testing tasks under a single trajectory guidance. DOAMRL rapidly adapts to testing tasks with only one adaptation step. In contrast, MAML and GMPS exhibit suboptimal performance due to the difficulty in obtaining sufficient reward feedback in sparse reward environments. The limited data constrains the GMPS adaptation and Meta-BC training. Introducing the offset facilitates the exploration of DOAMRL, while employing learnable imitation learning weight enables DOAMRL to utilize the provided demonstration data more appropriately. This contributes to DOAMRL outperforming EMRLD in both performance and sample efficiency.

**Performance with suboptimal demonstration data:** Fig. 5 illustrates different algorithms' training and testing performance when utilizing suboptimal data. As depicted in Fig. 5, DOAMRL outperforms all baselines and rapidly adapts to unseen tasks. This superior performance can be attributed to the effective utilization of both RL and imitation learning in DOAMRL, enabling it to overcome the challenges posed by suboptimal data. In contrast, Meta-BC and GMPS rely overly on data optimality, resulting in a notable performance decline. Meanwhile, MAML struggles to learn again due to the constraints of sparse rewards. Our proposed method, DOAMRL, surpasses EMRLD in performance and convergence speed. This improvement can be ascribed to the dynamic adjustment of the imitation learning

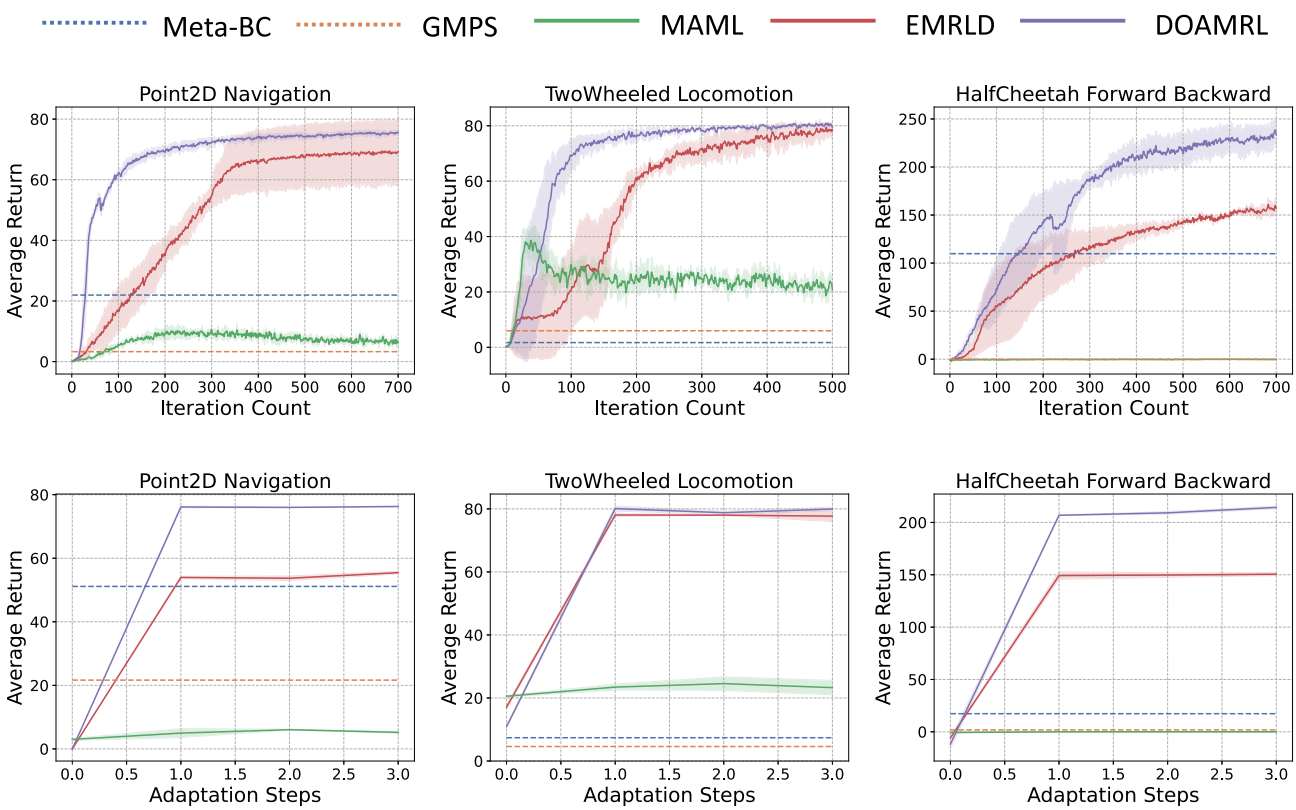



**Fig. 5** Average return curves during training (top) and testing (bottom) on Point2D Navigation, TwoWheeled Locomotion, and HalfCheetah Forward Backward with **suboptimal demonstrations**. The notions follow those of Fig. 4

weight and the enhancement of demonstration data, which enhance DOAMRL's ability to use suboptimal demonstration data for meta-policy training. Additionally, incorporating the parameter offset enhances exploration efficiency and update stability.

**Average return across all iterations:** Table 2 presents the average return of MAML, EMRLD, and DOAMRL across all iterations in three sparse reward environments. Each algorithm is trained in each environment with guidance from optimal and suboptimal demonstrations. Table 2 also compares the relevant improvements of DOAMRL with EMRLD. Regardless of the quality of demonstration data, DOAMRL achieves the highest average return across all three environments. Our proposed method demonstrates superior sample efficiency, rapid convergence, and the best asymptotic performance. Furthermore, under the guidance of suboptimal demonstrations, DOAMRL achieves a more significant improvement than EMRLD (in the Point2D Navigation environment, two kinds of demonstration data show similar improvements). This indicates that incorporating the learnable imitation learning weight and the meta-optimized parameter offset makes DOAMRL more robust in terms of the quality of demonstrations and aids it in more rapidly capturing tasks' commonalities.

**Average return over last 100 iterations:** Table 3 exhibits the average return of MAML, EMRLD, and DOAMRL under optimal and suboptimal demonstrations over the last 100 iterations in three sparse reward environments. At this stage, these three algorithms have essentially converged. DOAMRL consistently outperforms the other models in terms of asymptotic performance. Specifically, in the most challenging task, HalfCheetah Forward Backward, DOAMRL's performance under suboptimal data guidance achieves the most significant improvement. This indicates that the dynamic adjustment of the imitation learning weight and the demonstration data enhancement further improves the method's ability to leverage demonstration data for training the meta-policy. It also suggests that including the meta-optimized parameter offset improves the exploration and helps the policy overcome the limitations of demonstration data quality.

**Maximum average return across all iterations:** To better evaluate DOAMRL's performance, we compare it with two context-based meta-RL methods-PEARL and GENTLE-which differ from DOAMRL's framework, along with the baselines. Table 4 shows the maximum average return across all iterations for MAML, EMRLD, PEARL, GENTLE, and DOAMRL in three sparse reward environments. Only EMRLD and DOAMRL use demonstration data during train-

**Table 2** Average return and the performance improvement for DOAMRL compared to EMRLD **across all iterations**

| Environment | | MAML | EMRLD | DOAMRL | Improvement(%) w.r.t EMRLD |
|---|---|---|---|---|---|
| Point2D | Optimal data | 7.141 | 48.851 | **70.163** | +43.6% |
| Navigation | Suboptimal data | 7.141 | 49.508 | **67.758** | +36.9% |
| TwoWheeled | Optimal data | 24.435 | 66.124 | **76.433** | +15.6% |
| Locomotion | Suboptimal data | 24.435 | 53.547 | **69.320** | +29.5% |
| HalfCheetah | Optimal data | −0.419 | 147.486 | **169.290** | +14.8% |
| Forward Backward | Suboptimal data | −0.419 | 109.024 | **167.198** | +53.4% |

**Table 3** Average return and the performance improvement for DOAMRL compared to EMRLD **across the last 100 iterations**

| Environment | | MAML | EMRLD | DOAMRL | Improvement(%) w.r.t EMRLD |
|---|---|---|---|---|---|
| Point2D | Optimal data | 6.740 | 64.006 | **76.756** | +19.9% |
| Navigation | Suboptimal data | 6.740 | 68.832 | **75.280** | +9.4% |
| TwoWheeled | Optimal data | 22.551 | 81.130 | **82.505** | +1.7% |
| Locomotion | Suboptimal data | 22.551 | 77.133 | **80.005** | +3.7% |
| HalfCheetah | Optimal data | −0.313 | 220.502 | **235.559** | +6.8% |
| Forward Backward | Suboptimal data | −0.313 | 154.649 | **229.383** | +48.3% |

**Table 4** Maximum average return **across all iterations**

| Environment | | MAML | EMRLD | PEARL | GENTLE | DOAMRL |
|---|---|---|---|---|---|---|
| Point2D | Optimal | 11.138 | 65.483 | 27.451 | 0.001 | **77.057** |
| Navigation | Suboptimal | 11.138 | 70.230 | 27.451 | 0.001 | **75.816** |
| TwoWheeled | Optimal | 38.896 | 83.005 | **84.198** | 1.696 | 83.762 |
| Locomotion | Suboptimal | 38.896 | 80.365 | **84.198** | 1.696 | 81.160 |
| HalfCheetah | Optimal | −0.022 | 201.548 | 110.651 | 0.001 | **243.029** |
| Forward Backward | Suboptimal | −0.022 | 162.453 | 110.651 | 0.001 | **239.303** |

Maximum values for each environment and demonstration data quality combination are highlighted in **bold**

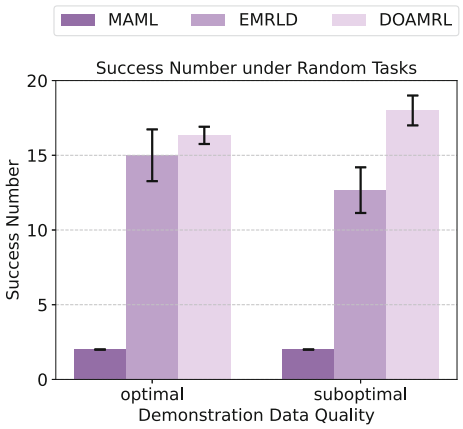



**Fig. 6** The number of successful arrivals after one-step adaptation using optimal and suboptimal demonstration data at 20 randomly selected target points in TwoWheeled Locomotion. Error bars indicate the standard deviation across three independent experiments

ing, which means that the maximum average return for the other methods remains unchanged regardless of the quality of demonstration data. DOAMRL achieves the highest average return in two of the three environments: Point2D Navigation and HalfCheetah Forward Backward. Additionally, it performs competitively in TwoWheeled Locomotion. In contrast, PEARL and GENTLE struggle to train effective context encoders for task representations due to the sparse nature of rewards in these environments. As a result, they produce poor task representations, which complicates the process of subsequent policy optimization. Additionally, the fully offline training of the meta-policy in GENTLE exacerbates the challenge by restricting the estimation of out-of-distribution state values. This limitations, in turn, hinders the learning of effective policies across all three environments. Among the three environments, TwoWheeled Locomotion has the lowest reward sparsity, which facilitates task inference and policy training for PEARL. This helps explain PEARL's strong performance in this environment, while its results in the other two environments are weaker. Overall, the comparison of DOAMRL with both similar and different framework meth-

ods highlights its effectiveness and robustness in handling sparse reward environments.

**Adaptability analysis:** We access the ability of various trained algorithms to new tasks unseen during training. Using the TwoWheeled Locomotion environment as a case study, Fig. 6 illustrates the number of successful target localization achieved by different algorithms across three independent experiments, each involving 20 random target points. The results demonstrate that the DOAMRL algorithm outperforms other baselines significantly, highlighting its superiority in sparse reward environments.

**Ablation analysis:** To validate the impact of DOAMRL components on its performance, take the Point2D Navigation environment as an example. Figure 7 illustrates the algorithm's performance when the demonstration data enhancement component (DE), the learnable imitation learning weight (LW), and the parameter offset (OF) are removed, respectively. The experimental results reveal that demonstration data guidance can significantly enhance the performance of the original MAML. Using the learnable imitation learning weight and the parameter offset facilitates meta-policy learning under different qualities of demonstration guidance. Additionally, incorporating the demonstration data enhancement module further improves performance and sampling efficiency under the suboptimal demonstration data guidance. Ultimately, the algorithm achieves the best performance when all these components are used together, confirming the critical role of each component in improving DOAMRL's overall performance.

**Sensitivity analysis:** Finally, we conduct a sensitivity analysis to assess the robustness of the proposed method to critical hyperparameters $w_{\mathrm{rl}}$ and $w_{\mathrm{bc}}$. It is important to note that $w_{\mathrm{bc}}$ refers to the initial value of the learnable imitation learning weight in DOAMRL. Using the Point2D Navigation environment as an example, Fig. 8 presents the average return of DOAMRL using optimal demonstration data across three random seeds in Point2D Navigation. The shaded region represents the standard deviation. Initially, we set $w_{\mathrm{bc}} = 1$ and vary $w_{\mathrm{rl}}$ from 0.2 to 1 in increments of 0.2. Similarly, we then fix $w_{\mathrm{rl}} = 1$ and vary $w_{\mathrm{bc}}$ from 0.2 to 1. All other hyperparameters are consistent with the standard configuration in Table 1.

The experimental results demonstrate that DOAMRL exhibits considerable robustness to variations in $w_{\mathrm{bc}}$ and $w_{\mathrm{rl}}$. Mainly, DOAMRL is more sensitive to changes in $w_{\mathrm{bc}}$ than $w_{\mathrm{rl}}$, emphasizing the crucial role of demonstration data in enabling the agent to obtain meaningful rewards in sparse

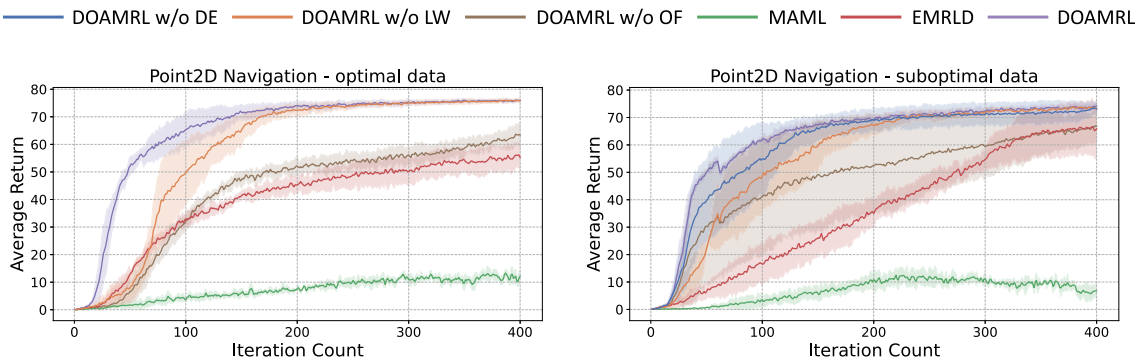
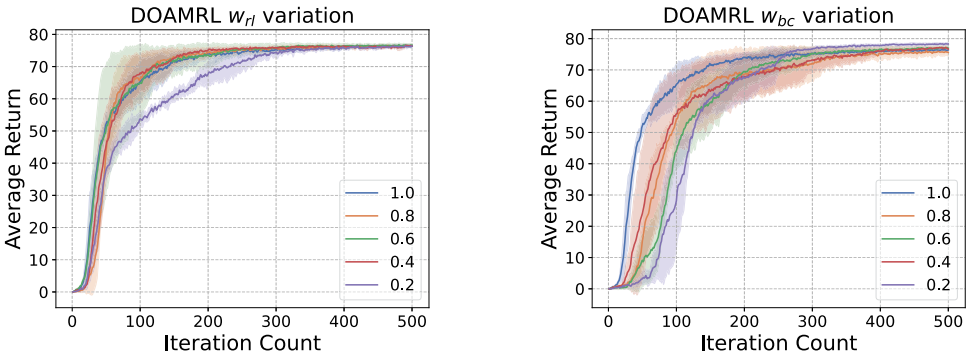



**Fig. 7** Ablation experiments on DOAMRL for Point2D Navigation environment by removing demonstration data enhancement component (DE), the learnable imitation learning weight (LW), and the parameter offset (OF) in the task adaptation phase with optimal (left) and suboptimal (right) demonstration data




(a) (b)

**Fig. 8** Sensitivity analysis for DOAMRL in Point2D Navigation when the demonstration data is optimal. **a** Keeping $w_{\mathrm{bc}} = 1$, $w_{\mathrm{rl}}$ is varied from 0.2 to 1 in steps of 0.2 for DOAMRL. **b** Keeping $w_{\mathrm{rl}} = 1$, $w_{\mathrm{bc}}$ is varied from 0.2 to 1 in steps of 0.2 for DOAMRL

reward environments. This further substantiates the rationality and efficiency of employing the meta-optimized imitation learning weight and updating the demonstration data through the demonstration data enhancement module when suboptimal demonstrations are initially provided.

## Conclusion

This paper presents DOAMRL, a novel meta-RL algorithm that enhances performance in sparse reward environments. The core of DOAMRL lies in its distinctive two-stage training approach. Initially, in the task adaptation phase, it skillfully combines RL and imitation learning. Subsequently, during both the adaptation and meta-optimization phases, a parameter offset is integrated into the initial MAML parameters, which is only updated in the meta-training phase. By incorporating the parameter offset, the optimization direction of the meta-policy can be calibrated, thereby incentivizing the agent to explore more extensively to learn more effective representations of task-specific features in each specific task. Meanwhile, by updating the imitation learning weight used in the task adaptation phase through meta-optimization, the agent can adjust its reliance on demonstrations based on the training stages and the quality of the demonstration data. Moreover, through the demonstration data enhancement mechanism, the agent can optimize the demonstration data even when initially guided by suboptimal demonstrations, and use higher-quality demonstrations to guide the meta-policy training in later training stages. Experimental results across three representative sparse reward environments demonstrate that, even with suboptimal demonstrations, DOAMRL can still gather informative samples to mitigate the adverse impacts of sparse rewards on learning. DOAMRL outperforms previous algorithms, thus validating its effectiveness.

A potential application of the DOAMRL method is in real-world robotic systems. When transitioning from simulation to real-world environments, model discrepancies can affect performance. By using trajectories from the simulated environment as demonstration data and combining them with real-world trajectories for DOAMRL adaptation, the algorithm's performance can be improved when applied to real-world tasks.

DOAMRL inherits the limitations common to gradient-based meta-RL methods like MAML, such as high data collection costs due to its on-policy nature and the requirement for gradient updates on test tasks. To address these challenges, exploring context-based meta-RL frameworks [66, 68] could offer insights into handling sparse rewards with demonstration data, without the need for gradient updates on test tasks. Additionally, investigating off-policy or offline meta-RL approaches using demonstrations could provide valuable solutions for applying meta-RL in real-world sparse reward scenarios.

## Appendix A Proof of Eq. (4)

In this section, we provide the proof of Eq. (4) based on the policy gradient theorem [59].

**Lemma 1** (Policy Gradient Theorem, [59]). *For any MDP, whether in the average-reward or start-state formulation, the policy gradient is given by*

$$\nabla_\theta \mathcal{L}(\theta) = \sum_s d^{\pi_\theta}(s) \sum_a Q^{\pi_\theta}(s,a) \nabla_\theta \pi_\theta(a \mid s), \tag{A1}$$

*where* $d^{\pi_\theta}(s) = \sum_{t=0}^{\infty} \gamma^t \Pr\{s_t = s \mid s_0, \pi_\theta\}$ *represents the discounted weighting of states encountered starting at* $s_0$*, and* Pr *denotes the probability.*

***Proof of Eq. (4)*** Since $d^{\pi_\theta}(s) = \sum_{t=0}^{\infty} \gamma^t \Pr\{s_t = s \mid s_0, \pi_\theta\}$ represents the discounted state distribution, it does not qualify as a "standard distribution". Specifically, the sum of $d^{\pi_\theta}(s)$ over all states $\sum_s d^{\pi_\theta}(s) = \sum_s \sum_{t=0}^{\infty} \gamma^t \Pr\{s_t = s \mid s_0, \pi_\theta\} = \sum_{t=0}^{\infty} \gamma^t \sum_s \Pr\{s_t = s \mid s_0, \pi_\theta\} = \sum_{t=0}^{\infty} \gamma^t = \frac{1}{1-\gamma}$ does not sum to 1. Therefore, to treat $d^{\pi_\theta}(s)$ as an expectation, we must normalize it into a proper probability distribution, denoted $D^{\pi_\theta}(s)$, i.e.,

$$\begin{aligned}
\nabla_\theta \mathcal{L}(\theta) &= \sum_s d^{\pi_\theta}(s) \sum_a Q^{\pi_\theta}(s,a) \nabla_\theta \pi_\theta(a \mid s) \\
&= \sum_s d^{\pi_\theta}(s) \sum_a \pi_\theta(a \mid s) Q^{\pi_\theta}(s,a) \frac{\nabla_\theta \pi_\theta(a \mid s)}{\pi_\theta(a \mid s)} \\
&= \sum_s d^{\pi_\theta}(s) \sum_a \pi_\theta(a \mid s) Q^{\pi_\theta}(s,a) \nabla_\theta \log \pi_\theta(a \mid s) \\
&= \frac{1}{1-\gamma} \sum_s (1-\gamma) d^{\pi_\theta}(s) \\
&\quad \sum_a \pi_\theta(a \mid s) Q^{\pi_\theta}(s,a) \nabla_\theta \log \pi_\theta(a \mid s) \\
&= \frac{1}{1-\gamma} \sum_s D^{\pi_\theta}(s) \sum_a \pi_\theta(a \mid s) Q^{\pi_\theta}(s,a) \nabla_\theta \log \pi_\theta(a \mid s) \\
&= \frac{1}{1-\gamma} \mathop{\mathbb{E}}_{\substack{s \sim D^{\pi_\theta} \\ a \sim \pi_\theta(a|s)}} \left[ Q^{\pi_\theta}(s,a) \nabla_\theta \log \pi_\theta(a \mid s) \right] \\
&\propto \mathop{\mathbb{E}}_{\substack{s \sim D^{\pi_\theta} \\ a \sim \pi_\theta(a|s)}} \left[ Q^{\pi_\theta}(s,a) \nabla_\theta \log \pi_\theta(a \mid s) \right].
\end{aligned} \tag{A2}$$

□

We now consider $\mathbb{E}_{s\sim D^{\pi_\theta}, a\sim\pi_\theta(a|s)}\left[V^{\pi_\theta}(s)\nabla_\theta \log \pi_\theta(a \mid s)\right]$:

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\substack{s\sim D^{\pi_\theta}\\ a\sim\pi_\theta(a|s)}}\left[V^{\pi_\theta}(s)\nabla_\theta \log \pi_\theta(a \mid s)\right] \\
&= \mathop{\mathbb{E}}_{s\sim D^{\pi_\theta}}\left[V^{\pi_\theta}(s) \mathop{\mathbb{E}}_{a\sim\pi_\theta(a|s)}\left[\nabla_\theta \log \pi_\theta(a \mid s)\right]\right] \qquad \text{(A3)}\\
&\overset{(a)}{=} \mathop{\mathbb{E}}_{s\sim D^{\pi_\theta}}\left[V^{\pi_\theta}(s)\cdot 0\right] \quad = 0.
\end{aligned}$$

Here, we get (a) from the fact from [69]. Starting with Eq. (A2), we obtain

$$\begin{aligned}
\nabla_\theta \mathcal{L}(\theta) &\propto \mathop{\mathbb{E}}_{\substack{s\sim D^{\pi_\theta}\\ a\sim\pi_\theta(a|s)}}\left[Q^{\pi_\theta}(s,a)\nabla_\theta \log \pi_\theta(a \mid s)\right] \\
&= \mathop{\mathbb{E}}_{\substack{s\sim D^{\pi_\theta}\\ a\sim\pi_\theta(a|s)}}\left[Q^{\pi_\theta}(s,a)\nabla_\theta \log \pi_\theta(a \mid s)\right] \\
&\quad - \mathop{\mathbb{E}}_{\substack{s\sim D^{\pi_\theta}\\ a\sim\pi_\theta(a|s)}}\left[V^{\pi_\theta}(s)\nabla_\theta \log \pi_\theta(a \mid s)\right] \\
&= \mathop{\mathbb{E}}_{\substack{s\sim D^{\pi_\theta}\\ a\sim\pi_\theta(a|s)}}\left[(Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s))\nabla_\theta \log \pi_\theta(a \mid s)\right] \\
&= \mathop{\mathbb{E}}_{\substack{s\sim D^{\pi_\theta}\\ a\sim\pi_\theta(a|s)}}\left[A^{\pi_\theta}(s,a)\nabla_\theta \log \pi_\theta(a \mid s)\right]. \qquad \text{(A4)}
\end{aligned}$$

Equation (A4) shows that while $\mathcal{L}(\theta)$ and $\mathbb{E}_{s\sim D^{\pi_\theta}, a\sim\pi_\theta(a|s)}\left[A^{\pi_\theta}(s,a)\nabla_\theta \log \pi_\theta(a \mid s)\right]$ are not exactly equal, they are proportional. In practice, the proportional constant is often ignored, as its impact can be compensated by adjusting the learning rate. Since we use gradient descent for gradient ascent, we include a negative sign in front of the result from Eq. (A4). Finally, by abbreviating $\mathbb{E}_{s\sim D^{\pi_\theta}, a\sim\pi_\theta(a|s)}$ as $\mathbb{E}_{(s_t,a_t)\sim\pi_\theta}$, we arrive at Eq. (4), thus completing the proof.

**Funding** This work was supported in part by the National Natural Science Foundation of China under Grant 72293573 and Grant 72293575, and in part by the Strategic Priority Research Program of Chinese Academy of Sciences under Grant XDA27030100 and the Excellent Youth Program of State Key Laboratory of Multimodal Artificial Intelligence Systems MAIS2024310.

## Declarations

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

**Open Access** This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by-nc-nd/4.0/.

## References

1. Finn C, Abbeel P, Levine S (2017) Model-agnostic meta-learning for fast adaptation of deep networks. In: Proceedings of the International Conference on Machine Learning, 70:1126–1135
2. Ladosz P, Weng L, Kim M, Oh H (2022) Exploration in deep reinforcement learning: a survey. Inf Fus 85:1–22
3. Hong W, Zhu M, Liu M, Zhang W, Zhou M, Yu Y, Sun P (2019) Generative adversarial exploration for reinforcement learning. In: Proceedings of the International Conference on Distributed Artificial Intelligence, pp. 1–10
4. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2020) Generative adversarial networks. Commun ACM 63(11):139–144
5. Stadie BC, Levine S, Abbeel P (2015) Incentivizing exploration in reinforcement learning with deep predictive models. arXiv preprint arXiv:1507.00814
6. Bougie N, Ichise R (2021) Fast and slow curiosity for high-level exploration in reinforcement learning. Appl Intell 51(2):1086–1107
7. Bougie N, Ichise R (2020) Towards high-level intrinsic exploration in reinforcement learning. In: Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI 2020, pp. 5186–5187
8. Burda Y, Edwards H, Storkey AJ, Klimov O (2019) Exploration by random network distillation. In: Proceedings of the International Conference on Learning Representations
9. Kolter JZ, Ng AY (2009) Near-bayesian exploration in polynomial time. In: Proceedings of the International Conference on Machine Learning 382:513–520
10. Pathak D, Gandhi D, Gupta A (2019) Self-supervised exploration via disagreement. In: Proceedings of TheInternational Conference on Machine Learning 97:5062–5071
11. Houthooft R, Chen X, Duan Y, Schulman J, Turck FD, Abbeel P (2016) VIME: variational information maximizing exploration. In: Proceedings of the Annual Conference on Neural Information Processing Systems, pp. 1109–1117
12. Lehman J, Stanley KO (2011) Abandoning objectives: evolution through the search for novelty alone. Evol Comput 19(2):189–223
13. Eysenbach B, Gupta A, Ibarz J, Levine S (2019) Diversity is all you need: learning skills without a reward function. In: Proceedings of the International Conference on Learning Representations
14. Gangwani T, Liu Q, Peng J (2019) Learning self-imitating diverse policies. In: Proceedings of the International Conference on Learning Representations
15. Pong VH, Nair A, Smith LM, Huang C, Levine S (2022) Offline meta-reinforcement learning with online self-supervision. In: Proceedings of the International Conference on Machine Learning 162:17811–17829
16. Packer C, Abbeel P, Gonzalez JE (2021) Hindsight task relabelling: experience replay for sparse reward meta-RL. In: Proceedings of the Annual Conference on Neural Information Processing Systems, pp. 2466–2477
17. Guo Y, Wu Q, Lee H (2022) Learning action translator for meta reinforcement learning on sparse-reward tasks. In: Proceeding of the AAAI Conference on Artificial Intelligence, pp. 6792–6800
18. Hua Y, Wang X, Jin B, Li W, Yan J, He X, Zha H (2021) HMRL: hyper-meta learning for sparse reward reinforcement learning

problem. In: Proceeding of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 637–645
19. Nam T, Sun S-H, Pertsch K, Hwang SJ, Lim JJ (2022) Skill-based meta-reinforcement learning. In: Proceedings of the International Conference on Learning Representations
20. Zhang J, Wang J, Hu H, Chen T, Chen Y, Fan C, Zhang C (2021) MetaCURE: meta reinforcement learning with empowerment-driven exploration. In: Proceedings of the International Conference on Machine Learning, vol. 139, pp. 12600–12610
21. Zintgraf LM, Feng L, Lu C, Igl M, Hartikainen K, Hofmann K, Whiteson S (2021) Exploration in approximate hyper-state space for meta reinforcement learning. In: Proceedings of the International Conference on Machine Learning 139:12991–13001
22. Vecerik M, Hester T, Scholz J, Wang F, Pietquin O, Piot B, Heess N, Rothörl T, Lampe T, Riedmiller M (2017) Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. arXiv preprint arXiv:1707.08817
23. Hester T, Vecerík M, Pietquin O, Lanctot M, Schaul T, Piot B, Horgan D, Quan J, Sendonaris A, Osband I, Dulac-Arnold G, Agapiou JP, Leibo JZ, Gruslys A (2018) Deep Q-learning From Demonstrations. In: Proceeding of the AAAI Conference on Artificial Intelligence, pp. 3223–3230
24. Nair A, McGrew B, Andrychowicz M, Zaremba W, Abbeel P (2018) Overcoming exploration in reinforcement learning with demonstrations. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 6292–6299
25. Rajeswaran A, Kumar V, Gupta A, Vezzani G, Schulman J, Todorov E, Levine S (2018) Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. In: Proceeding of the Robotics: Science and Systems
26. Rengarajan D, Vaidya G, Sarvesh A, Kalathil DM, Shakkottai S (2022) Reinforcement learning with sparse rewards using guidance from offline demonstration. In: Proceeding of the International Conference on Learning Representations
27. Finn C, Yu T, Zhang T, Abbeel P, Levine S (2017) One-shot visual imitation learning via meta-learning. In: Proceeding of the Annual Conference on Robot Learning, 78:357–368
28. Zhou A, Jang E, Kappler D, Herzog A, Khansari M, Wohlhart P, Bai Y, Kalakrishnan M, Levine S, Finn C (2020) Watch, try, learn: meta-learning from demonstrations and rewards. In: Proceeding of the International Conference on Learning Representations
29. Mendonca R, Gupta A, Kralev R, Abbeel P, Levine S, Finn C (2019) Guided meta-policy search. In: Proceedings of the Annual Conference on Neural Information Processing Systems, pp. 9653–9664
30. Rengarajan D, Chaudhary S, Kim J, Kalathil D, Shakkottai S (2022) Enhanced meta reinforcement learning via demonstrations in sparse reward environments. In: Proceedings of the Annual Conference on Neural Information Processing Systems
31. Peng Z, Han C, Liu Y, Zhou Z (2023) Weighted policy constraints for offline reinforcement learning. In: Proceedings of the AAAI Conference on Artificial Intelligence 37:9435–9443
32. Yang Y, Caluwaerts K, Iscen A, Tan J, Finn C (2019) NoRML: no-reward meta learning. In: Proceedings of the International Conference on Autonomous Agents and MultiAgent Systems, pp. 323–331
33. Raghu A, Raghu M, Bengio S, Vinyals O (2020) Rapid learning or feature reuse? Towards understanding the effectiveness of MAML. In: Proceeding of the International Conference on Learning Representations
34. He K, Zhang X, Ren S, Sun J (2016) Identity mappings in deep residual networks. In: Proceedings of the European Conference on Computer Vision, vol. 9908, pp. 630–645
35. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, Uszkoreit J, Houlsby N (2021) An image is worth 16x16 words: transformers for image recognition at scale. In: Proceeding of the International Conference on Learning Representations
36. Li Z, Tucker R, Snavely N, Holynski A (2024) Generative image dynamics. In: Proceeding of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 24142–24153
37. Rafailov R, Sharma A, Mitchell E, Manning CD, Ermon S, Finn C (2023) Direct preference optimization: your language model is secretly a reward model. In: Proceedings of the Annual Conference on Neural Information Processing Systems
38. Xiao Z, Xu X, Xing H, Zhao B, Wang X, Song F, Qu R, Feng L (2024) DTCM: deep transformer capsule mutual distillation for multivariate time series classification. IEEE Trans Cogn Dev Syst 16(4):1445–1461
39. Xiao Z, Tong H, Qu R, Xing H, Luo S, Zhu Z, Song F, Feng L (2023) CapMatch: semi-supervised contrastive transformer capsule with feature-based knowledge distillation for human activity recognition. IEEE Transactions on Neural Networks and Learning Systems, 1–15
40. Koch G, Zemel R, Salakhutdinov R, et al (2015) Siamese neural networks for one-shot image recognition. In: Proceedings of the International Conference on Machine Learning on Deep Learning Workshop, 2:1–30
41. Vinyals O, Blundell C, Lillicrap T, Kavukcuoglu K, Wierstra D (2016) Matching networks for one shot learning. In: Proceedings of the Annual Conference on Neural Information Processing Systems, pp. 3630–3638
42. Snell J, Swersky K, Zemel RS (2017) Prototypical networks for few-shot learning. In: Proceedings of the Annual Conference on Neural Information Processing Systems, pp. 4077–4087
43. Nichol A, Schulman J (2018) Reptile: a scalable metalearning algorithm. arXiv preprint arXiv:1803.02999
44. Harrison J, Sharma A, Pavone M (2018) Meta-learning priors for efficient online bayesian regression. In: Proceedings of the Workshop on the Algorithmic Foundations of Robotics 14:318–337
45. Duan Y, Schulman J, Chen X, Bartlett PL, Sutskever I, Abbeel P (2016) $RL^2$: fast reinforcement learning via slow reinforcement learning. arXiv preprint arXiv:1611.02779
46. Mishra N, Rohaninejad M, Chen X, Abbeel P (2018) A simple neural attentive meta-learner. In: Proceeding of the International Conference on Learning Representations
47. Stadie BC, Yang G, Houthooft R, Chen X, Duan Y, Wu Y, Abbeel P, Sutskever I (2018) Some considerations on learning to explore via meta-reinforcement learning. arXiv preprint arXiv:1803.01118
48. Wang JX, Kurth-Nelson Z, Tirumala D, Soyer H, Leibo JZ, Munos R, Blundell C, Kumaran D, Botvinick M (2016) Learning to reinforcement learn. arXiv preprint arXiv:1611.05763
49. Gupta A, Mendonca R, Liu Y, Abbeel P, Levine S (2018) Meta-reinforcement learning of structured exploration strategies. In: Proceedings of the Annual Conference on Neural Information Processing Systems, pp. 5307–5316
50. Houthooft R, Chen Y, Isola P, Stadie BC, Wolski F, Ho J, Abbeel P (2018) Evolved policy gradients. In: Proceedings of the Annual Conference on Neural Information Processing Systems, pp. 5405–5414
51. Sung F, Zhang L, Xiang T, Hospedales T, Yang Y (2017) Learning to learn: meta-critic networks for sample efficient learning. arXiv preprint arXiv:1706.09529
52. Beck J, Vuorio R, Liu EZ, Xiong Z, Zintgraf L, Finn C, Whiteson S (2023) A survey of meta-reinforcement learning. arXiv preprint arXiv:2301.08028
53. Rothfuss J, Lee D, Clavera I, Asfour T, Abbeel P (2019) ProMP: proximal meta-policy search. In: Proceeding of the International Conference on Learning Representations
54. Foerster JN, Farquhar G, Al-Shedivat M, Rocktäschel T, Xing EP, Whiteson S (2018) DiCE: the infinitely differentiable monte

carlo estimator. In: Proceedings of the International Conference on Machine Learning 80:1524–1533
55. Al-Shedivat M, Bansal T, Burda Y, Sutskever I, Mordatch I, Abbeel P (2018) Continuous adaptation via meta-learning in nonstationary and competitive environments. In: Proceeding of the International Conference on Learning Representations
56. Packer C, Abbeel P, Gonzalez JE (2021) Hindsight task relabelling: experience replay for sparse reward Meta-RL. In: Ranzato M, Beygelzimer A, Dauphin YN, Liang P, Vaughan JW (eds) Proceedings of the Annual Conference on Neural Information Processing Systems, pp. 2466–2477
57. Xue Z, Peng Z, Li Q, Liu Z, Zhou B (2023) Guarded policy optimization with imperfect online demonstrations. In: Proceeding of the International Conference on Learning Representations
58. Sutton RS, Barto AG (2018) Reinforcement learning: an introduction
59. Sutton RS, McAllester DA, Singh S, Mansour Y (1999) Policy gradient methods for reinforcement learning with function approximation. In: Proceedings of the Annual Conference on Neural Information Processing Systems, pp. 1057–1063
60. Schulman J, Moritz P, Levine S, Jordan MI, Abbeel P (2016) High-dimensional continuous control using generalized advantage estimation. In: Proceeding of the International Conference on Learning Representations
61. Haarnoja T, Zhou A, Abbeel P, Levine S (2018) Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: Proceedings of the International Conference on Machine Learning 80:1856–1865
62. Schulman J, Levine S, Abbeel P, Jordan MI, Moritz P (2015) Trust region policy optimization. In: Proceedings of the International Conference on Machine Learning 37:1889–1897
63. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. In: Proceedings of the International Conference on Learning Representations
64. He K, Zhang X, Ren S, Sun J (2016) Identity mappings in deep residual networks. In: Proceedings of the European Conference on Computer Vision 9908:630–645
65. Todorov E, Erez T, Tassa Y (2012) Mujoco: a physics engine for model-based control. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5026–5033
66. Rakelly K, Zhou A, Finn C, Levine S, Quillen D (2019) Efficient off-policy meta-reinforcement learning via probabilistic context variables. In: Proceedings of the International Conference on Machine Learning 97:5331–5340
67. Zhou R, Gao C, Zhang Z, Yu Y (2024) Generalizable task representation learning for offline meta-reinforcement learning with data limitations. In: Proceeding of the AAAI Conference on Artificial Intelligence, pp. 17132–17140
68. Gao Y, Zhang R, Guo J, Wu F, Yi Q, Peng S, Lan S, Chen R, Du Z, Hu X, Guo Q, Li L, Chen Y (2023) Context shift reduction for offline meta-reinforcement learning. In: Proceedings of the Annual Conference on Neural Information Processing Systems, pp. 80024–80043
69. Williams RJ (1992) Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach Learn 8:229–256

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.