HarmoDT: Harmony Multi-Task Decision Transformer for Offline Reinforcement Learning

Shengchao Hu [* 1 2] **Ziqing Fan** [* 1 2] **Li Shen** [3 4] **Ya Zhang** [1 2] **Yanfeng Wang** [1 2] **Dacheng Tao** [5]

Abstract

The purpose of offline multi-task reinforcement learning (MTRL) is to develop a unified policy applicable to diverse tasks without the need for online environmental interaction. Recent advancements approach this through sequence modeling, leveraging the Transformer architecture's scalability and the benefits of parameter sharing to exploit task similarities. However, variations in task content and complexity pose significant challenges in policy formulation, necessitating judicious parameter sharing and management of conflicting gradients for optimal policy performance. In this work, we introduce the Harmony Multi-Task Decision Transformer (HarmoDT), a novel solution designed to identify an optimal harmony subspace of parameters for each task. We approach this as a bi-level optimization problem, employing a meta-learning framework that leverages gradient-based techniques. The upper level of this framework is dedicated to learning a task-specific mask that delineates the harmony subspace, while the inner level focuses on updating parameters to enhance the overall performance of the unified policy. Empirical evaluations on a series of benchmarks demonstrate the superiority of HarmoDT, verifying the effectiveness of our approach.

1. Introduction

Offline reinforcement learning (RL) (Levine et al., 2020) enables the learning of policies directly from an existing offline dataset, thus eliminating the need for interaction

---
[*]Equal contribution [1]Shanghai Jiao Tong University, China [2]Shanghai AI Laboratory, China [3]Sun Yat-sen University, China [4]JD Explore Adademy, China [5]Nanyang Technological University, Singapore. Correspondence to: Li Shen <mathshenli@gmail.com>.
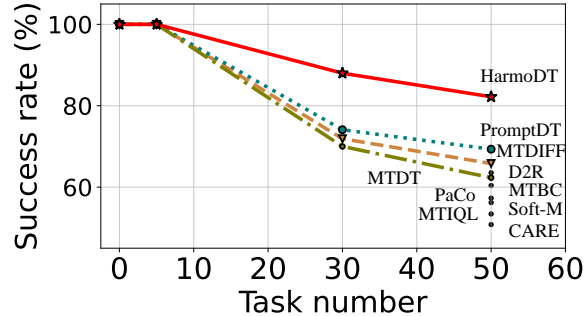
Figure 1. Illustration of a comparative analysis of success rates across various task numbers within the Meta-World benchmark, focusing on prevalent MTRL algorithms. An in-depth exploration of these results refers to Section 5.

with the actual environment. Despite the promising developments of offline RL in various robotic tasks, its successes have been largely confined to individual tasks within specific domains, such as locomotion or manipulation (Fu et al., 2020; Kumar et al., 2020). Drawing inspiration from human learning capabilities, where individuals often acquire new skills by building upon existing ones and spend less time mastering similar tasks, there's a growing interest in the potential of training a set of tasks with inherent similarities in a more cohesive and efficient manner (Lee et al., 2022). This perspective leads to the exploration of multi-task reinforcement learning (MTRL), which seeks to develop a versatile policy to address a variety of tasks.

Recent developments in Offline RL, such as the Decision Transformer (Chen et al., 2021) and Trajectory Transformer (Janner et al., 2021), have abstracted offline RL as a sequence modeling (SM) problem, showcasing their ability to transform extensive datasets into powerful decision-making tools (Hu et al., 2022). These models are particularly beneficial for multi-task RL challenges, offering a high-capacity framework capable of accommodating task variances and assimilating extensive knowledge from diverse datasets. Additionally, they open up possibilities for integrating advancements (Brown et al., 2020) from language modeling into MTRL methodologies. However, the application of these high-capacity sequential models to MTRL presents considerable algorithmic challenges. As indicated by Yu

et al. (2020b), simply employing a shared network backbone for all diverse robot manipulation tasks can lead to severe gradient conflicts. This situation arises when the gradient direction for a particular task starkly contrasts with the majority consensus direction. Such unregulated sharing of parameters and their optimization under conflicting gradient conditions can contravene the foundational goals of MTRL, degrading performance relative to task-specific training methods (Sun et al., 2022). Furthermore, the issue of gradient conflict is exacerbated by an increase in the number of tasks (detailed in Section 3), underscoring the urgency for effective solutions to these challenges.

Existing works on offline MTRL generally address the problem in one of three ways (Sun et al., 2022): 1) developing shared structures for the sub-policies of different tasks, as explored in works by Calandriello et al. (2014); Yang et al. (2020); Lin et al. (2022); 2) optimizing task-specific representations to condition the policies, as discussed by Sodhani et al. (2021); Lee et al. (2022); He et al. (2023a); 3) addressing the conflicting gradients arising from different task losses during training, a focus of research by Yu et al. (2020a); Chen et al. (2020); Liu et al. (2021a). While these methods have demonstrated effectiveness in different scenarios, they often fall short of adequately addressing the occurrence of conflicting gradients that stem from indiscriminate parameter sharing (Guangyuan et al., 2022). In contrast, our innovative method, the Harmony Multi-Task Decision Transformer (HarmoDT), diverges from these traditional approaches. HarmoDT endeavors to identify a harmony parameter subspace within a single policy for each task, offering a novel solution to the challenges of offline MTRL.

To reduce the occurrence of the conflicting gradient, the idea of adopting distinct parameter subspaces for each task is straightforward. Empirical observations, depicted by Figure 2(a), affirm that the application of masks significantly mitigates conflicts, leading to considerable performance gains across various sparsity ratios[1], as contrasted with the non-mask baseline shown in Figure 2(b). Building upon these insights, our HarmoDT seeks to identify an optimal harmony subspace of parameters for each task by incorporating trainable task-specific masks during MTRL training. This approach is conceptualized as a bi-level optimization problem, employing a meta-learning framework to discern the harmony subspace mask via gradient-based techniques. At the upper level, we focus on learning a task-specific mask that delineates the harmony subspace, while at the inner level, we update parameters to augment the collective performance of the unified model under the guidance of the task-specific mask. Empirical evaluations of HarmoDT, conducted across a broad spectrum of tasks in both seen and unseen settings, demonstrate its efficacy against mul-

[1]Sparsity ratio refers to the percentage of inactive weights.
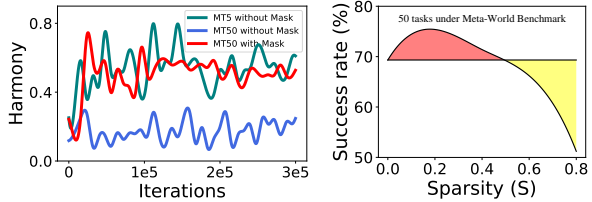
(a) Conflicting during MTRL. (b) Success with task masks.

Figure 2. Illustration of the harmony degree among trainable weights during training for policies with and without randomly initialized masks (left panel), and the success rates achieved when applying masks with varying sparsity levels (right panel).

tiple state-of-the-art algorithms. Additionally, we provide extensive ablation studies on various aspects, including scalability, model size, hyper-parameters, and visualizations, to comprehensively validate our approach.

In summary, our research makes three significant contributions to the field of MTRL:

- We rethink the challenges in MTRL from the perspective of sequence modeling, analyze gradient conflicts with increasing task numbers, and propose the harmony subspace using task-specific masks (Section 3).

- We model the problem as a bi-level optimization problem and introduce a meta-learning framework to find the optimal harmony subspace mask through gradient-based techniques (Section 4).

- We demonstrate the superior performance of HarmoDT through rigorous testing on a broad spectrum of benchmarks, establishing its state-of-the-art effectiveness in MTRL scenarios (Section 5).

2. Preliminary

2.1. Offline Reinforcement Learning

The goal of RL is to learn a policy $\pi_\theta(\mathbf{a}|\mathbf{s})$ maximizing the expected cumulative discounted rewards $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(\mathbf{s}_t, \mathbf{a}_t)]$ in a Markov decision process (MDP), which is a six-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, d_0)$, with state space $\mathcal{S}$, action space $\mathcal{A}$, environment dynamics $\mathcal{P}(\mathbf{s}'|\mathbf{s}, \mathbf{a})$: $\mathcal{S} \times \mathcal{S} \times \mathcal{A} \to [0, 1]$, reward function $\mathcal{R}$: $\mathcal{S} \times \mathcal{A} \to \mathbb{R}$, discount factor $\gamma \in [0, 1)$, and initial state distribution d_0 (Sutton & Barto, 2018). The action-value or Q-value of a policy π is defined as $Q^\pi(\mathbf{s}_t, \mathbf{a}_t) = \mathbb{E}_{\mathbf{a}_{t+1}, \mathbf{a}_{t+2} \cdots \sim \pi}[\sum_{i=0}^{\infty} \gamma^i \mathcal{R}(\mathbf{s}_{t+i}, \mathbf{a}_{t+i})]$. In the offline setting (Levine et al., 2020), instead of the online environment, a static dataset $\mathcal{D} = \{(\mathbf{s}, \mathbf{a}, \mathbf{s}', r)\}$, collected by a behavior policy π_β, is provided. Offline RL algorithms learn a policy entirely from this static offline dataset $\mathcal{D}$, without any online interactions with the environment.

In the multi-task setting, different tasks can have different reward functions, state spaces, and transition functions. We consider all tasks to share the same action space with the same embodied agent. Given a specific task $\mathcal{T} \sim p(\mathcal{T})$, a task-specified MDP can be defined as $(\mathcal{S}^{\mathcal{T}}, \mathcal{A}^{\mathcal{T}}, \mathcal{P}^{\mathcal{T}}, \mathcal{R}^{\mathcal{T}}, \gamma, d_0^{\mathcal{T}})$. Instead of solving a single MDP, the goal of multi-task RL is to find an optimal policy that maximizes expected return over all the tasks: $\pi^* = \arg\max_{\pi} \mathbb{E}_{\mathcal{T} \sim p(\mathcal{T})} \mathbb{E}_{\mathbf{a}_t \sim \pi}[\sum_{t=0}^{\infty} \gamma^t r_t^{\mathcal{T}}]$. The static dataset $\mathcal{D}$ correspondingly is partitioned into per-task subsets as $\mathcal{D} = \cup_{i=1}^{N} \mathcal{D}_i$, where N is the number of tasks.

2.2. Prompt Decision Transformer

The integration of the Transformer (Vaswani et al., 2017) architecture in offline RL for SM has gained prominence in recent years. Studies in NLP reveal that Transformers pre-trained on extensive datasets exhibit notable few-shot or zero-shot learning capabilities within a prompt-based framework (Liu et al., 2023; Brown et al., 2020). Building on this, Prompt-DT adapts the prompt-based methodology to offline RL, facilitating few-shot generalization to novel tasks. Unlike NLP, where prompts are typically text-based and adapt to various tasks through blank-filling formats, Prompt-DT introduces trajectory prompts. These prompts consist of state, action, and return-to-go tuples $(\mathbf{s}^*, \mathbf{a}^*, \hat{r}^*)$, providing directed guidance to RL agents with few-shot demonstrations. Each element marked with the superscript $\cdot^*$ is relevant to the trajectory prompt. Note that the length of the trajectory prompt is usually shorter than the task's horizon, encompassing only essential information to facilitate task identification, yet inadequate for complete task imitation. During training with offline collected data, Prompt-DT utilizes $\tau_{i,t}^{input} = (\tau_i^*, \tau_{i,t})$ as input for each task $\mathcal{T}_i$. Here, $\tau_{i,t}^{input}$ consists of the K^*-step trajectory prompt τ_i^* and the most recent K-step history $\tau_{i,t}$, and is formulated as:

$$\tau_{i,t}^{input} = (\hat{r}_{i,1}^*, \mathbf{s}_{i,1}^*, \mathbf{a}_{i,1}^*, \ldots, \hat{r}_{i,K^*}^*, \mathbf{s}_{i,K^*}^*, \mathbf{a}_{i,K^*}^*,$$
$$\hat{r}_{i,t-K+1}, \mathbf{s}_{i,t-K+1}, \mathbf{a}_{i,t-K+1}, \ldots, \hat{r}_{i,t}, \mathbf{s}_{i,t}, \mathbf{a}_{i,t}). \quad (1)$$

The prediction head linked to a state token $\mathbf{s}$ is designed to predict the corresponding action $\mathbf{a}$. For continuous action spaces, the training objective aims to minimize the mean-squared loss:

$$\mathcal{L}_{DT} = \mathbb{E}_{\tau_{i,t}^{input} \sim \mathcal{D}_i} \left[\frac{1}{K} \sum_{m=t-K+1}^{t} (\mathbf{a}_{i,m} - \pi(\tau_i^*, \tau_{i,m}))^2 \right]. \quad (2)$$

3. Rethinking SM with MTRL

Recent works in offline RL conceptualize it as sequence modeling (SM), effectively transforming extensive datasets into potent decision-making systems. This approach is advantageous for multi-task RL, offering a high-capacity

model that accommodates task discrepancies and assimilates comprehensive knowledge from diverse datasets. However, the application of such high-capacity sequential models to multi-task RL introduces significant algorithmic challenges. In this section, we delineate the primary challenges, particularly conflicting gradients, and explore the concepts of parameter subspace and harmony, laying the groundwork for the motivation behind our method.

3.1. Conflicting Gradients

In a multi-task training context, the aggregate gradient, $\hat{\mathbf{g}}$, is computed across multiple tasks and is defined as

$$\hat{\mathbf{g}} = \mathbb{E}_{\mathcal{T}_i \sim p(\mathcal{T})} \nabla \mathcal{L}_{\mathcal{T}_i}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{g}_i(\theta), \quad (3)$$

where θ represents the trainable parameter vector and $\mathbf{g}_i$ is the gradient vector for task $\mathcal{T}_i$. In scenarios where tasks are diverse, the gradients $\mathbf{g}_i$ from different tasks may conflict significantly, a phenomenon known as gradient conflicts and negative transfer in multi-task learning (Guangyuan et al., 2022; Tang et al., 2023).

Definition 3.1 (Harmony Score on a Single Weight). The harmony score of the j-th element in the weights vector is estimated by calculating the corresponding coordinate of the element-wise product of the task gradient and the total gradient, denoted as $(\mathbf{g}_i \odot \hat{\mathbf{g}})_j = \mathbf{g}_{i,j} \times \hat{\mathbf{g}}_j$.

Definition 3.2 (Averaged Harmony Score). The overall harmony score across all weights is evaluated using $\frac{1}{NK} \sum_{i=1}^{N} \sum_{j=1}^{K} \frac{(\mathbf{g}_i \odot \hat{\mathbf{g}})_j}{|\mathbf{g}_{i,j}||\hat{\mathbf{g}}_j|}$, where K and N denote the number of weights and tasks. This score, ranging between -1 and 1, reflects the degree of alignment among tasks.

To substantiate the presence of conflicts in MTRL, we establish two metrics to measure harmony scores and conduct experiments utilizing the Prompt-DT method on 5 and 50 tasks from the Meta-world benchmark, recording the average harmony score. As illustrated in Figure 2(a), the averaged harmony score significantly diminishes with the escalation in the number of tasks, indicating pronounced conflicts among tasks and underscoring the imperative to address these conflicts in MTRL.

3.2. Parameter Subspace and Harmony

Parameter subspace, a concept prevalent in pruning-aware training (Alvarez & Salzmann, 2017), aims to maintain comparable performance while achieving a sparse model. In the context of MTRL via SM, pruning to preserve distinct parameter subspaces for each task markedly alleviates gradient conflicts. To validate this, we conduct experiments on 50 tasks from the Meta-World benchmark. Each task $\mathcal{T}_i$ is assigned a randomly initialized mask $M^{\mathcal{T}_i}$ with a specific
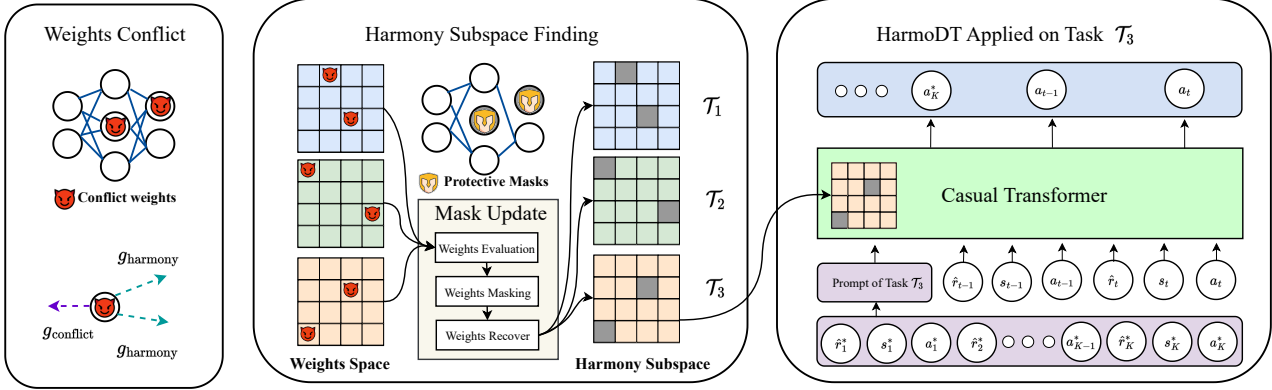
Figure 3. Illustration of the conflicting problem and the framework of our method HarmoDT to find a harmony subspace for each task. The left panel shows the conflicting phenomenon reflected by divergent task-specific gradients. The middle panel illustrates the procedure to find a harmony subspace for each task via the strategic learning of task masks. The right panel demonstrates the workflow of HarmoDT based on the DT architecture with prompts and learned harmony subspace of weights when handling a task, such as $\mathcal{T}_3$.

sparsity ratio S. During training, this mask modulates both the trainable parameters and gradients as follows:

$$\bar{\mathbf{g}}_i = \nabla \mathcal{L}_{\mathcal{T}_i}(\theta \odot \boldsymbol{M}^{\mathcal{T}_i}) \odot \boldsymbol{M}^{\mathcal{T}_i}, \ \ i = 1, 2, \ldots, N, \quad (4)$$

where $\odot$ represents element-wise multiplication, and $\boldsymbol{M}^{\mathcal{T}_i}$ is a binary vector. Intriguingly, as shown in Figure 2(b), applying the mask could result in enhanced performance across a wide range of sparsity ratios. This improvement, coupled with a significantly higher harmony score in multi-task settings as depicted in Figure 2(a), suggests that maintaining a subspace of parameters through the implementation of task-specific masks effectively mitigates the conflicts arising from unregulated parameter sharing.

3.3. Motivation

The complexity in MTRL is significantly amplified with increasing task numbers, largely attributed to escalating gradient conflicts. This challenge mainly stems from the unregulated sharing of parameters among tasks, intended to leverage similarities and enhance learning efficiency, but often leading to performance degradation. In response to these challenges, a nuanced approach involving task-specific masks has been proposed. These masks aim to maintain distinct parameter subspaces for each task, thereby ensuring that the learning process of one task does not adversely affect the others. While this strategy represents a step towards mitigating gradient conflicts, it introduces a new challenge: the determination of an optimal mask configuration for each task remains an open question. The complexity of this challenge is compounded by the dynamic and often non-linear nature of task interactions within the shared model space.

Addressing this intricate problem requires a sophisticated solution that can navigate the delicate balance between shared learning and task-specific adaptation. Our study proposes a bi-level optimization strategy situated within a meta-learning framework to address this issue. This approach leverages gradient-based techniques to meticulously explore and exploit the parameter space, aiming to identify a harmony parameter subspace for each task, thereby optimizing the overall MTRL performance.

4. Method: Find Harmony Subspace

To address the aforementioned problem, we introduce a meta-learning framework that discerns an optimal harmony subspace of parameters for each task, enhancing parameter sharing and mitigating gradient conflicts. This problem is formulated as a bi-level optimization, where we meta-learn task-specific masks to define the harmony subspace. Mathematically, we can express the problem as:

$$\max_{\mathbb{M}} \ \ \mathbb{E}_{\mathcal{T}_i \sim p(\mathcal{T})}[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}^{\mathcal{T}_i}(s_t, \pi(\tau_{i,t}^{input} | \theta^{*\mathcal{T}_i}))], \quad (5)$$

$$\text{s.t.} \ \ \theta^* = \arg\min_{\theta} \mathbb{E}_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{DT}(\theta, \mathbb{M}), \quad (6)$$

$$\text{where } \theta^{*\mathcal{T}_i} = \theta^* \odot \boldsymbol{M}^{\mathcal{T}_i}, \mathbb{M} = \{\boldsymbol{M}^{\mathcal{T}_i}\}_{\mathcal{T}_i \sim p(\mathcal{T})}, \quad (7)$$

where $\boldsymbol{M}^{\mathcal{T}_i}$ represents a binary task mask vector corresponding to $\mathcal{T}_i$, and $\mathbb{M}$ denotes the set of all task masks. The goal at the upper level is to learn a task-specific mask that identifies the harmony subspace for each task. Concurrently, at the inner level, the objective is to optimize the algorithmic parameters θ, maximizing the collective performance of the unified model under the guidance of the task-specific masks. The framework for our harmony subspace learning is depicted in Figure 3. Subsequent sections are meticulously dedicated to elucidating the methodology for selecting the harmony subspace, detailing the sophisticated update mechanism for task masks (refer to Section 4.1), and delineating the procedural intricacies of the algorithm (see Section 4.2).

Algorithm 1 HarmoDT

Input: A set of tasks $\mathbb{T} = \{\mathcal{T}_1, \ldots, \mathcal{T}_N\}$, maximum iteration E, episode length T, target return $\mathbf{G}^*$, learning rate η, hyper-parameters $\{\eta_{\max}, \eta_{\min}, s, t_m, \lambda, \text{thresh}\}$.

`// initializing stage`
Initialize the parameters of the network θ_0 and the set of task masks $\mathbb{M} = \{\boldsymbol{M}^{\mathcal{T}_1}, \ldots, \boldsymbol{M}^{\mathcal{T}_N}\}$ through ERK.

`// training stage`
$t = 1$.
while $t \leq E$ **do**
　　$\alpha_t = \lceil \eta_{max} + \frac{1}{2}(\eta_{min} - \eta_{max})(1 + \cos(2\pi \frac{t}{E}))\rceil$.
　　$\mathbb{M} = \text{Mask\_Update}(\mathbb{T}, \mathbb{M}, \theta_t, \lambda, \alpha_t)$.
　　while $t \bmod t_m \neq 0$ **do**
　　　　sample a task $\mathcal{T}_i$ from the set of tasks $\mathbb{T}$.
　　　　sample a batch of data τ_i from the dataset $\mathcal{D}_i$.
　　　　$\theta \leftarrow \theta - \eta \nabla \mathcal{L}_{\mathcal{T}_i}(\theta_t \odot \boldsymbol{M}^{\mathcal{T}_i}) \odot \boldsymbol{M}^{\mathcal{T}_i}$.
　　　　$t = t + 1$.
　　end while
　　$t = t + 1$.
end while

`// inference stage`
for $i = 1, \ldots, N$ **do**
　　Initialize history τ with zeros, desired reward $\hat{r} = \mathbf{G}_i^*$, prompt τ^*, the parameters $\theta_i \leftarrow \theta \odot \boldsymbol{M}^{\mathcal{T}_i}$.
　　$j = 0$.
　　for $j \leq T$ **do**
　　　　Get action $a = \text{Transformer}_{\theta_i}(\tau^*, \tau)[-1]$.
　　　　Step env. and get feedback $\mathbf{s}, \mathbf{a}, r, \hat{r} \leftarrow \hat{r} - r$.
　　　　Append $[\mathbf{s}, \mathbf{a}, \hat{r}]$ to recent history τ.
　　end for
end for

4.1. Mask Update

For a given sparsity S and task masks $\mathbb{M}$, we periodically assess the harmony subspace of trainable weights θ across all tasks. This process involves masking[2] α_t of the most conflicting parameters and subsequently recovering an equal number of previously masked parameters that have transitioned to harmony after the subsequent iterative training process. As delineated in Algorithm 2, this procedure encompasses three key steps: Weights Evaluation, Weights Masking, and Weights Recovery.

Weights Evaluation. During training, our aim is to iteratively identify a harmony subspace for each task by assessing trainable parameter conflicts and importance. This involves defining two metrics: the Agreement Score and the Importance Score, to gauge the concordance and significance of weights respectively.

Definition 4.1 (Agreement Score). For each task $\mathcal{T}_i$ with a set of task masks $\mathbb{M}$, the agreement score vector of all trainable weights is defined as follows: $A(\mathcal{T}_i) = \bar{\mathbf{g}}_i \odot \frac{1}{N} \sum_{i=1}^{N} \bar{\mathbf{g}}_i$, where $\bar{\mathbf{g}}_i$ is defined in Equation 4.

---

[2]When the term "mask" is used as a verb, it refers to the action of rendering the corresponding parameter inactive or modifying the mask vector by changing the value from 1 to 0.

Algorithm 2 Mask Update

Input: A set of tasks $\mathbb{T}$, a set of task masks $\mathbb{M}$, trainable weights vector θ_t and hyper-parameters $\{\lambda, \alpha_t\}$.

for $i = 1, \ldots, N$ **do**
　　$\bar{\mathbf{g}}_i = \nabla \mathcal{L}_{T_i}(\theta \odot \boldsymbol{M}^{\mathcal{T}_i}) \odot \boldsymbol{M}^{\mathcal{T}_i}$.
　　$\mathbf{g}_i = \nabla \mathcal{L}_{T_i}(\theta)$.
end for
$\hat{\mathbf{g}} = \frac{1}{N} \sum_{i=1}^{N} \bar{\mathbf{g}}_i$.
for $i = 1, \ldots, N$ **do**
　　`// Weights Evaluation`
　　Calculate $H(\mathcal{T}_i)$ with λ, $\mathbf{g}_i$ and $\hat{\mathbf{g}}$ as Sec. 4.1.
　　`// Weights Masking`
　　$\boldsymbol{M}^{\mathcal{T}_i} = \boldsymbol{M}^{\mathcal{T}_i} - \text{ArgBtmK}_{\alpha_t}(H(\mathcal{T}_i))$.
　　`// Weights Recovery`
　　$\boldsymbol{M}^{\mathcal{T}_i} = \boldsymbol{M}^{\mathcal{T}_i} + \text{ArgTopK}_{\alpha_t}(\mathbf{g}_i \odot \hat{\mathbf{g}})$.
end for

Output: $\mathbb{M}$.

Definition 4.2 (Importance Score). This metric evaluates the significance of parameters for task $\mathcal{T}_i$. It can be measured either by the absolute value of the parameters $I_M(\mathcal{T}_i) = |(\theta^{\mathcal{T}_i})|$, indicating magnitude-based importance, or by the Fisher information $I_F(\mathcal{T}_i) = \left(\nabla \log \mathcal{L}_{\mathcal{T}_i}(\theta^{\mathcal{T}_i}) \odot \boldsymbol{M}^{\mathcal{T}_i}\right)^2$, reflecting the parameters' impact on output variability.

For task $\mathcal{T}_i$, $A(\mathcal{T}_i)$ reflects the gradient similarity between the task-specific and the average masked gradients, while $I_M(\mathcal{T}_i)_j$ and $I_F(\mathcal{T}_i)_j$ measure the j-th element's importance. Lower values of $A(\mathcal{T}_i)_j$ or $I_{M/F}(\mathcal{T}_i)_j$ indicate increased conflict or diminished importance regarding the j-th element of the trainable parameters for the respective task. The Harmony Score $H(\mathcal{T}_i)_j$ for the j-th parameter of task $\mathcal{T}_i$ is computed as a weighted balance between the Agreement and Importance Scores, moderated by a balance factor λ:

$$H(\mathcal{T}_i)_j = \begin{cases} A(\mathcal{T}_i)_j + \lambda I_{M/F}(\mathcal{T}_i)_j, & (\boldsymbol{M}^{\mathcal{T}_i})_j = 1, \\ \inf, & (\boldsymbol{M}^{\mathcal{T}_i})_j = 0. \end{cases} \quad (8)$$

Parameters that have been already masked (i.e., $(\boldsymbol{M}^{\mathcal{T}_i})_j = 0$) are assigned an infinite Harmony Score to prevent their re-selection due to the pre-existing conflicts.

Weights Masking. Employing the Harmony Score, we identify and mask the most conflicting and less significant weights within the harmony subspace as follows:

$$\boldsymbol{M}^{\mathcal{T}_i} = \boldsymbol{M}^{\mathcal{T}_i} - \text{ArgBtmK}_{\alpha_t}(H(\mathcal{T}_i)), \quad (9)$$

where α_t represents the number of masks altered in the t-th iteration, and $\text{ArgBtmK}_{\alpha_t}(\cdot)$ generates zero vectors matching the dimension of $\boldsymbol{M}^{\mathcal{T}_i}$, marking the positions of the top-α_t smallest values from $H(\mathcal{T}_i)$ with 1.

Weights Recover. To maintain a fixed sparsity ratio and recover weights that have transitioned from conflict to harmony, the following recovery process is applied:

$$M^{\mathcal{T}_i} = M^{\mathcal{T}_i} + \text{ArgTopK}_{\alpha_t}(\mathbf{g}_i \odot \frac{1}{N}\sum_{i=1}^{N}\bar{\mathbf{g}}_i), \quad (10)$$

where $\text{ArgTopK}_{\alpha_t}(\cdot)$ generates zero vectors matching the dimension of $M^{\mathcal{T}_i}$, setting the positions corresponding to the top-α_t largest values from $(\mathbf{g}_i \odot \frac{1}{N}\sum_{i=1}^{N}\bar{\mathbf{g}}_i)$ to 1. This step ensures the reintegration of previously conflicting weights that have harmonized after subsequent iterations.

4.2. Training and Inference

Algorithm 1 provides the meta-learning process for the task mask set $\mathbb{M}$ and the update mechanism for the trainable parameters θ of the unified model. Given a set of source tasks, we first initialize corresponding masks through the ERK technique (Evci et al., 2020) with a predefined sparsity ratio S for each task (See Appendix C for more details). In the inner loop, we optimize the parameters of the unified model under the guidance of the task-specific mask:

$$\theta_{t+1} = \theta_t - \eta\mathbb{E}_{\mathcal{T}_i \sim p(\mathcal{T})}\nabla\mathcal{L}_{\mathcal{T}_i}(\theta \odot M^{\mathcal{T}_i}) \odot M^{\mathcal{T}_i}. \quad (11)$$

Then, in the outer loop, we optimize the set of task masks through the procedure detailed in Section 4.1 to find the harmony subspace for each separate task. Considering the stability and efficiency of the updating, we adopt a warm-up and cosine annealing strategy (Liu et al., 2021b) to control the updating number α_t. Given the maximum iterations E, α_t in t-th iteration is defined as:

$$\alpha_t = \lceil \eta_{max} + \frac{1}{2}(\eta_{min} - \eta_{max})(1 + \cos(2\pi\frac{t}{E}))\rceil, \quad (12)$$

where $\lceil\cdot\rceil$ represents the round-up command, and η_{min} and η_{max} denote the lower and upper bounds, respectively, on the number of parameters that undergo changes during the mask update process.

In the inference stage, task IDs are accessible in the online test environment, following the methodology of He et al. (2023a). Accordingly, the task-specific mask is applied to the parameters, and the inference process is conducted in line with Xu et al. (2022). For unseen tasks that differ from training tasks but share identical states and transitions, we aggregate task-specific masks from training tasks to formulate a generalized model. The mask for unseen tasks is constructed as follows:

$$\hat{M}_j = \begin{cases} 0, & \sum_{i=1}^{N}M_{i,j} \leq \text{thresh}, \\ 1, & \sum_{i=1}^{N}M_{i,j} > \text{thresh}, \end{cases} \quad (13)$$

where $\hat{M}$ denotes the mask for the unseen task, and 'thresh' is a predefined threshold. This approach integrates insights from all training tasks, enhancing the model's adaptability to novel scenarios.

5. Experiment

In this section, we conduct extensive experiments to answer the following questions: (1) How does HarmoDT compare to other offline and online baselines in the multi-task regime? (2) Does HarmoDT mitigate the phenomenon of conflicting gradients and identify an optimal harmony subspace of parameters for each task? (3) Can HarmoDT[3] generalize to unseen tasks?

5.1. Environments and Baselines

Environment. Our experiments utilize the Meta-World benchmark (Yu et al., 2020b), featuring 50 distinct manipulation tasks with shared dynamics, requiring a Sawyer robot to interact with various objects. We extend tasks to a random-goal setting, consistent with recent studies (He et al., 2023b; Sun et al., 2022). Performance evaluation is based on the averaged success rate across tasks. Following He et al. (2023a), we employ two dataset compositions: a near-optimal dataset from SAC-Replay (Haarnoja et al., 2018) ranging from random to expert experiences, and a sub-optimal dataset with initial trajectories and a reduced proportion (50%) of expert data.

For unseen tasks, HarmoDT's performance is evaluated on distinct objectives from datasets used in prior works (Mitchell et al., 2021; Yu et al., 2020b; Xu et al., 2022; Hu et al., 2023b), specifically Cheetah-dir, Cheetah-vel, and Ant-dir, which challenge the agent to optimize direction and speed. Details on environment specifications and hyperparameters are available in the Appendix A and B.

Baselines. We compare our proposed HarmoDT with the following offline baselines. (i) **MTBC**: Extends Behavior Cloning for multi-task learning with network scaling and a task-ID conditioned actor; (ii) **MTIQL**: Adapts IQL (Kostrikov et al., 2021) with multi-head critics and a task-ID conditioned actor for multi-task policy learning; (iii) **MTDIFF-P** (He et al., 2023a): A diffusion-based method combining Transformer architectures and prompt learning for generative planning in multitask offline settings; (iv) **MTDT**: Extends DT (Chen et al., 2021) to multitask settings, utilizing a task ID encoding and state input for task-specific learning; (v) **Prompt-DT** (Xu et al., 2022): Builds on DT, leveraging trajectory prompts and reward-to-go for multi-task learning and generalization to unseen tasks. The results of these offline methods are directly replicated from He et al. (2023a), with the exception that the approaches marked with * are implemented by us.

Besides, we compare our method with four online RL methods: (vi) **CARE** (Sodhani et al., 2021): Utilizes metadata

---

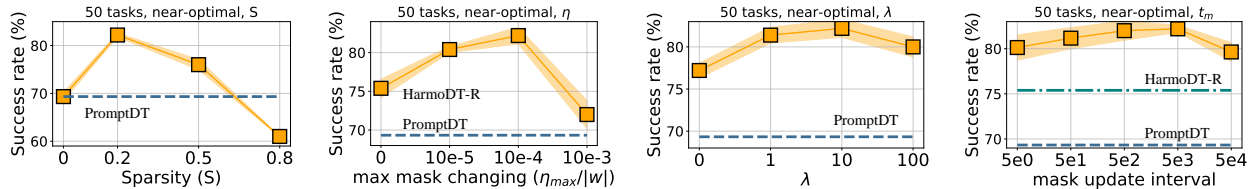[3]Our code is available at: https://github.com/charleshsc/HarmoDT

Figure 4. From the left to right, we illustrate the ablation results on the Meta-World benchmark with 50 tasks under the near-optimal case. Default values are listed as $\eta_{min} = 0$, η_{max} is 100 (about 1e-3% of total weights), S = 0.2, $\lambda = 10$ and $t_m = 5e3$. During each individual ablation, a single parameter is varied, with all other parameters maintained at their default values. Detailed results pertaining to additional settings are comprehensively documented in the Appendix E.

Table 1. Average success rate across 3 seeds on Meta-World MT50 with random goals (MT50-rand) under both near-optimal and sub-optimal cases. Each task is evaluated for 50 episodes. Approaches with * indicate baselines of our own implementation.

Method	Meta-World 50 Tasks		
#Partition	Near-optimal	Sub-optimal	Params
CARE (online)	$46.12_{\pm1.30}$	-	1.26 M
PaCo (online)	$54.31_{\pm1.32}$	-	3.39 M
Soft-M (online)	$53.41_{\pm0.72}$	-	1.62 M
D2R (online)	$63.53_{\pm1.22}$	-	1.40 M
MTBC	$60.39_{\pm0.86}$	$34.53_{\pm1.25}$	1.74 M
MTIQL	$56.21_{\pm1.39}$	$43.28_{\pm0.90}$	1.74 M
MTDIFF-P	$59.53_{\pm1.12}$	$48.67_{\pm1.32}$	5.32 M
MTDIFF-P-ONEHOT	$61.32_{\pm0.89}$	**$48.94_{\pm0.95}$**	5.32 M
MTDT	$20.99_{\pm2.66}$	$20.63_{\pm2.21}$	0.87 M
MTDT*	$65.80_{\pm1.02}$	$42.33_{\pm1.89}$	1.47 M
Prompt-DT	$45.68_{\pm1.84}$	$39.76_{\pm2.79}$	0.87 M
Prompt-DT*	**$69.33_{\pm0.89}$**	$48.40_{\pm0.16}$	1.47 M
HarmoDT-R (ours)	**$75.39_{\pm1.18}$**	**$53.80_{\pm1.07}$**	1.47 M
HarmoDT-M(ours)	**$80.33_{\pm0.97}$**	**$57.20_{\pm0.73}$**	1.47 M
HarmoDT-F(ours)	**$82.20_{\pm0.40}$**	**$57.20_{\pm0.68}$**	1.47 M

and a mixture of encoders for task representation; (vii) **PaCo** (Sun et al., 2022): Employs a parameter compositional approach for task-specific parameter recombination; (viii) **Soft-M** (Yang et al., 2020): Focuses on a routing network for the soft combination of modules; (ix) **D2R** (He et al., 2023b): Adopts disparate routing paths for module selection per task. The results of these methods are directly extracted from He et al. (2023b). Detailed descriptions of these baselines are summarized in the Appendix D.

5.2. Main Results

In this study, we benchmark HarmoDT and its variants against established baselines on 50 Meta-World tasks. The variants considered in this evaluation include **HarmoDT-R**, which maintains frozen task masks throughout the training process; **HarmoDT-F**, which utilizes fisher information $I_F(\mathcal{T}_i)$ for calculating weight importance; and **HarmoDT-M**, which employs magnitude information $I_M(\mathcal{T}_i)$ for determining weight importance. Note that the primary distinction between HarmoDT-F and HarmoDT-M resides in their re-

spective approaches to weight masking (Equation 9).

As shown in Table 1, HarmoDT-R, following Prompt-DT's structure, surpasses all other methods, achieving a 6.1% and 4.9% improvement in near-optimal and sub-optimal scenarios, respectively, compared to the best baseline. By employing fixed random masks, HarmoDT-R effectively competes with the current state-of-the-art techniques. Furthermore, the variants, HarmoDT-M and HarmoDT-F, enhance the performance of HarmoDT-R by identifying optimal harmony subspaces through task mask learning, resulting in substantial gains of 6.8% and 3.4% in near-optimal and sub-optimal cases, respectively. Our novel technique, HarmoDT, showcases its effectiveness in multi-task settings, encompassing both sub-optimal datasets that require the stitching of useful segments from suboptimal trajectories and near-optimal datasets where the emulation of optimal behaviors is crucial.

5.3. Further Analysis

This section delves into the scalability of task scale, model size, and hyper-parameters. It also examines the visualization of task masks for the harmony subspace and evaluates the performance on unseen tasks.

Scalability of Task Scale. We evaluated HarmoDT's scalability across varying task numbers in the Meta-World benchmark. As shown in Table 2 and Figure 1, HarmoDT consistently outperforms MTDIFF, MTDT, and Prompt-DT across all task numbers and cases, demonstrating promising superiority with increasing task count: 11% for 5 tasks, 11% for 30 tasks, and 8% for 50 tasks in sub-optimal settings.

Impact of Model Size. The influence of model size is pronounced in multi-task training scenarios. Table 3 delineates our ablation study on model size across 1e6 iterations. Models are characterized by their parameters (z M), layers (x), and head attentions (y), represented as (x, y, z). Results reveal that increasing model size markedly boosts performance for all evaluated methods. Significantly, our approach, HarmoDT, demonstrates consistent superiority over MTDT and Prompt-DT across a range of model sizes. This success is attributed to HarmoDT's effective establishment of a harmony subspace for each task.

Table 2. We randomly select 5, 30, 50 tasks from the Meta-World benchmark under both near-optimal and sub-optimal cases and record the average success rate across 3 seeds. Each task is evaluated for 50 episodes.

Method	Meta-World 5 Tasks		Meta-World 30 Tasks		Meta-World 50 Tasks	
#Partition	Near-optimal	Sub-optimal	Near-optimal	Sub-optimal	Near-optimal	Sub-optimal
MTDIFF	$\mathbf{100.0} \pm 0.0$	66.30 ± 2.31	67.52 ± 0.35	$\mathbf{54.21} \pm 1.10$	61.32 ± 0.89	$\mathbf{48.94} \pm 0.95$
MTDT	$\mathbf{100.0} \pm 0.0$	64.67 ± 5.25	71.89 ± 0.95	49.33 ± 2.05	65.80 ± 1.02	42.33 ± 1.89
Prompt-DT	$\mathbf{100.0} \pm 0.0$	$\mathbf{66.50} \pm 2.74$	$\mathbf{74.10} \pm 0.69$	53.78 ± 0.63	$\mathbf{69.33} \pm 0.89$	48.40 ± 0.16
HarmoDT-R	$\mathbf{100.0} \pm 0.0$	66.14 ± 3.00	82.65 ± 2.11	61.22 ± 2.16	75.39 ± 1.18	53.80 ± 1.07
HarmoDT-M	$\mathbf{100.0} \pm 0.0$	74.00 ± 4.79	84.67 ± 0.84	63.67 ± 0.73	80.33 ± 0.97	57.20 ± 0.73
HarmoDT-F	$\mathbf{100.0} \pm 0.0$	$\mathbf{77.67} \pm 3.25$	$\mathbf{88.00} \pm 0.96$	$\mathbf{65.00} \pm 0.92$	$\mathbf{82.20} \pm 0.40$	$\mathbf{57.20} \pm 0.68$

Table 3. Ablation study on the model size of MTDT, Prompt-DT, and our HarmoDT-F under near-optimal of Meta-World 30 tasks and 50 tasks. We denote the model with z M parameters and x layers of y head attentions as (x, y, z) in the table.

Case	Model	MTDT	Prompt-DT	HarmoDT-F
30 tasks	(1, 2, 0.48)	$35.78_{\pm 0.9}$	$42.10_{\pm 3.1}$	$59.33_{\pm 1.5}$
	(3, 4, 0.87)	$64.01_{\pm 1.4}$	$69.56_{\pm 1.0}$	$77.67_{\pm 1.6}$
	(6, 8, 1.47)	$71.89_{\pm 1.0}$	$74.10_{\pm 0.7}$	$88.00_{\pm 1.0}$
	(9, 16, 2.06)	$74.33_{\pm 0.7}$	$77.67_{\pm 1.7}$	$88.00_{\pm 1.6}$
50 tasks	(1, 2, 0.48)	$31.93_{\pm 3.4}$	$36.07_{\pm 2.5}$	$58.20_{\pm 1.9}$
	(3, 4, 0.87)	$58.13_{\pm 3.6}$	$61.93_{\pm 0.4}$	$72.80_{\pm 1.6}$
	(6, 8, 1.47)	$65.80_{\pm 1.0}$	$69.33_{\pm 0.9}$	$82.20_{\pm 0.4}$
	(9, 16, 2.06)	$68.33_{\pm 1.1}$	$71.00_{\pm 0.9}$	$83.33_{\pm 0.6}$

Table 4. Generalization ability to unseen tasks. Here we conduct experiments and record the cumulative reward of unseen tasks on three distinct datasets: Cheetah-dir, Cheetah-vel, and Ant-dir, which challenge the agent to optimize direction and speed.

Setting	MTDT	Prompt-DT	HarmoDT-F
Cheetah-dir	$662.40_{\pm 1.3}$	$935.3_{\pm 2.6}$	$\mathbf{958.5}_{\pm 1.5}$
Cheetah-vel	$-170.11_{\pm 5.7}$	$-127.7_{\pm 9.9}$	$\mathbf{-66.51}_{\pm 1.2}$
Ant-dir	$165.29_{\pm 0.4}$	$278.7_{\pm 38.7}$	$\mathbf{298.3}_{\pm 1.0}$
Average	219.2	362.1	$\mathbf{396.8}_{9.6\%\uparrow}$

Ablation on Hyper-parameters. This study introduces hyper-parameters η_{max} for cosine annealing (with $\eta_{min} = 0$), mask alteration frequency t_m, overall sparsity S, and balance controller λ. Comprehensive ablations are conducted to establish an empirical strategy for selecting these parameters. Figure 4 delineates the ablation study on 50 tasks of Meta-World benchmark in the near-optimal settings. Ablation results on other settings can be found in the Appendix E. Across a broad spectrum of hyper-parameter values, our approach consistently outperforms baselines. Based on these insights, recommended settings for hyper-parameters are sparsity ratio S = 0.2, η_{max} at approximately 0.001% of total weights, balance factor $\lambda = 10$ and mask changing interval $t_m = 5000$ rounds. These parameters collectively contribute to the superior performance of our method.
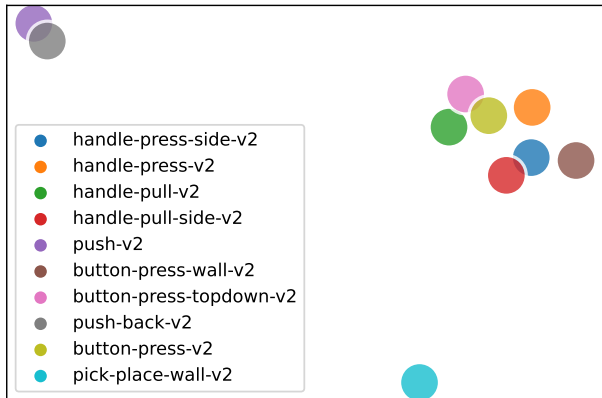

Figure 5. The t-SNE visualization of optimal subspace via masks learned by our HarmoDT on the 30 tasks of Meta-World benchmark. The figure illustrates the relational dynamics of task-specific masks, with a focus on 10 representative tasks from the total set.

Visualization of Mask. As shown in Figure 5, we use t-SNE (Van der Maaten & Hinton, 2008) to visualize the task masks post-training on 30 tasks from the Meta-World benchmark. Note that even small distances in the visualization can represent significant divergences in the original high-dimensional parameter space. The visualization effectively showcases the relational dynamics of the task masks; closely related tasks such as 'push-back-v2' and 'push-v2' are positioned in proximity, while disparate tasks like 'push-v2' and 'pick-place-wall-v2' are distinctly separated. This spatial arrangement underscores the efficacy of our HarmoDT in delineating a harmony subspace tailored for each task.

Ability to unseen tasks. Prompt-DT's proficiency with unseen tasks prompted us to assess HarmoDT's capabilities in similar scenarios. We employ a voting mechanism among all observed tasks to define a generalized subspace for unseen tasks, as delineated in Equation 13. This technique operates on a foundational assumption: parameters that are consistently identified as significant and harmonious across a range of tasks (surpassing a predefined threshold) are posited to hold universal value, potentially contributing

positively to task performance in novel scenarios. Comparative analysis involving HarmoDT, MTDT, and Prompt-DT is conducted on three distinct datasets: Cheetah-dir, Cheetah-vel, and Ant-dir. The results, presented in Table 4, affirm HarmoDT's comprehensive enhancements across all test cases. Notably, HarmoDT demonstrates an average reward of 396.8, surpassing Prompt-DT's 362.1 with a substantial 9.6% improvement. This outcome underscores the efficacy of our voting approach in addressing unseen tasks.

6. Conclusion

In this study, we introduce the Harmony Multi-Task Decision Transformer (HarmoDT), a novel approach designed to discern an optimal parameter subspace for each task, leveraging parameter sharing to harness task similarities while concurrently addressing the adverse impacts of conflicting gradients. By employing a bi-level optimization and a meta-learning framework, HarmoDT not only excels as a comprehensive policy in multi-task environments but also exhibits robust generalization capabilities to unseen tasks. Our rigorous empirical evaluations across a diverse array of benchmarks underscore HarmoDT's superior performance compared to existing baselines, establishing its state-of-the-art effectiveness in MTRL scenarios.

Limitation. We present an innovative approach to policy learning in multi-task offline RL, achieving state-of-the-art performance across various tasks. However, the efficacy of our approach depends on the model's capacity, as it employs a sparsification strategy for each task's parameter space. Moreover, tasks of differing complexity inherently require varying numbers of parameters for optimal performance. Our method, however, uses the same number of task-specific parameters across different tasks. Complex tasks could benefit from a denser parameter allocation, while simpler tasks might achieve peak efficiency with a sparser configuration.

Acknowledgements

Impact Statement

In domains such as healthcare and robotics, multi-task scenarios are commonplace. Our algorithm enhances the applicability of DT by extending them to complex multi-task environments, increasing their practical utility. However, the performance on certain tasks may deteriorate if they are provided with bad task masks, potentially due to malicious intent. Therefore, it's imperative to rigorously follow the algorithmic process during updates, treating each task with equal consideration. To date, our investigations have not uncovered any adverse societal impacts.

References

Ajay, A., Du, Y., Gupta, A., Tenenbaum, J., Jaakkola, T., and Agrawal, P. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

Alvarez, J. M. and Salzmann, M. Compression-aware training of deep networks. *Advances in neural information processing systems*, 30, 2017.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Calandriello, D., Lazaric, A., and Restelli, M. Sparse multi-task reinforcement learning. *Advances in neural information processing systems*, 27, 2014.

Chen, H., Lu, C., Ying, C., Su, H., and Zhu, J. Offline reinforcement learning via high-fidelity generative behavior modeling. *arXiv preprint arXiv:2209.14548*, 2022.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

Chen, Z., Ngiam, J., Huang, Y., Luong, T., Kretzschmar, H., Chai, Y., and Anguelov, D. Just pick a sign: Optimizing deep multitask models with gradient sign dropout. *Advances in Neural Information Processing Systems*, 33: 2039–2050, 2020.

D'Eramo, C., Tateo, D., Bonarini, A., Restelli, M., Peters, J., et al. Sharing knowledge in multi-task deep reinforcement learning. In *8th International Conference on Learning Representations*, pp. 1–11. OpenReview. net, 2020.

Evci, U., Gale, T., Menick, J., Castro, P. S., and Elsen, E. Rigging the lottery: Making all tickets winners. In *International Conference on Machine Learning*, pp. 2943–2952. PMLR, 2020.

Fan, Z., Wang, Y., Yao, J., Lyu, L., Zhang, Y., and Tian, Q. Fedskip: Combatting statistical heterogeneity with federated skip aggregation. In *2022 IEEE International Conference on Data Mining (ICDM)*, pp. 131–140. IEEE, 2022.

Fan, Z., Yao, J., Zhang, R., Lyu, L., Wang, Y., and Zhang, Y. Federated learning under partially disjoint data via manifold reshaping. *Transactions on Machine Learning Research*, 2023.

Fan, Z., Hu, S., Yao, J., Niu, G., Zhang, Y., Sugiyama, M., and Wang, Y. Locally estimated global perturbations are better than local perturbations for federated sharpness-aware minimization. In *International Conference on Machine Learning*, 2024a.

Fan, Z., Yao, J., Han, B., Zhang, Y., Wang, Y., et al. Federated learning with bilateral curation for partially class-disjoint data. *Advances in Neural Information Processing Systems*, 36, 2024b.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Fujimoto, S. and Gu, S. S. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pp. 2052–2062. PMLR, 2019.

Guangyuan, S., Li, Q., Zhang, W., Chen, J., and Wu, X.-M. Recon: Reducing conflicting gradients from the root for multi-task learning. In *The Eleventh International Conference on Learning Representations*, 2022.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

He, H., Bai, C., Xu, K., Yang, Z., Zhang, W., Wang, D., Zhao, B., and Li, X. Diffusion model is an effective planner and data synthesizer for multi-task reinforcement learning. *arXiv preprint arXiv:2305.18459*, 2023a.

He, J., Li, K., Zang, Y., Fu, H., Fu, Q., Xing, J., and Cheng, J. Not all tasks are equally difficult: Multi-task reinforcement learning with dynamic depth routing. *arXiv preprint arXiv:2312.14472*, 2023b.

Hu, S., Shen, L., Zhang, Y., Chen, Y., and Tao, D. On transforming reinforcement learning by transformer: The development trajectory. *arXiv preprint arXiv:2212.14164*, 2022.

Hu, S., Shen, L., Zhang, Y., and Tao, D. Graph decision transformer. *arXiv preprint arXiv:2303.03747*, 2023a.

Hu, S., Shen, L., Zhang, Y., and Tao, D. Prompt-tuning decision transformer with preference ranking. *arXiv preprint arXiv:2305.09648*, 2023b.

Hu, S., Fan, Z., Huang, C., Shen, L., Zhang, Y., Wang, Y., and Tao, D. Q-value regularized transformer for offline reinforcement learning. In *International Conference on Machine Learning*, 2024a.

Hu, S., Shen, L., Zhang, Y., and Tao, D. Learning multi-agent communication from graph modeling perspective. In *The Twelfth International Conference on Learning Representations*, 2024b.

Janner, M., Li, Q., and Levine, S. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34: 1273–1286, 2021.

Janner, M., Du, Y., Tenenbaum, J. B., and Levine, S. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

Kostrikov, I., Nair, A., and Levine, S. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 1179–1191, 2020.

Lee, K.-H., Nachum, O., Yang, M. S., Lee, L., Freeman, D., Guadarrama, S., Fischer, I., Xu, W., Jang, E., Michalewski, H., et al. Multi-game decision transformers. *Advances in Neural Information Processing Systems*, 35: 27921–27936, 2022.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Lin, Q., Liu, H., and Sengupta, B. Switch trajectory transformer with distributional value approximation for multi-task reinforcement learning. *arXiv preprint arXiv:2203.07413*, 2022.

Liu, B., Liu, X., Jin, X., Stone, P., and Liu, Q. Conflict-averse gradient descent for multi-task learning. *Advances in Neural Information Processing Systems*, 34:18878–18890, 2021a.

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., and Neubig, G. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

Liu, S., Yin, L., Mocanu, D. C., and Pechenizkiy, M. Do we actually need dense over-parameterization? in-time over-parameterization in sparse training. In *International Conference on Machine Learning*, pp. 6989–7000. PMLR, 2021b.

Meng, L., Wen, M., Le, C., Li, X., Xing, D., Zhang, W., Wen, Y., Zhang, H., Wang, J., Yang, Y., et al. Offline pre-trained multi-agent decision transformer. *Machine Intelligence Research*, 2023.

Mitchell, E., Rafailov, R., Peng, X. B., Levine, S., and Finn, C. Offline meta-reinforcement learning with advantage weighting. In *International Conference on Machine Learning*, pp. 7780–7791. PMLR, 2021.

Plappert, M., Andrychowicz, M., Ray, A., McGrew, B., Baker, B., Powell, G., Schneider, J., Tobin, J., Chociej, M., Welinder, P., et al. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*, 2018.

Sarafian, E., Keynan, S., and Kraus, S. Recomposing the reinforcement learning building blocks with hypernetworks. In *International Conference on Machine Learning*, pp. 9301–9312. PMLR, 2021.

Sodhani, S., Zhang, A., and Pineau, J. Multi-task reinforcement learning with context-based representations. In *International Conference on Machine Learning*, pp. 9767–9779. PMLR, 2021.

Sun, L., Zhang, H., Xu, W., and Tomizuka, M. Paco: Parameter-compositional multi-task reinforcement learning. *Advances in Neural Information Processing Systems*, 35:21495–21507, 2022.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Tang, A., Shen, L., Luo, Y., Ding, L., Hu, H., Du, B., and Tao, D. Concrete subspace learning based interference elimination for multi-task model fusion. *arXiv preprint arXiv:2312.06173*, 2023.

Van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, Z., Hunt, J. J., and Zhou, M. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

Xu, M., Shen, Y., Zhang, S., Lu, Y., Zhao, D., Tenenbaum, J., and Gan, C. Prompting decision transformer for few-shot policy generalization. In *international conference on machine learning*, pp. 24631–24645. PMLR, 2022.

Xu, Q., Zhang, R., Fan, Z., Wang, Y., Wu, Y.-Y., and Zhang, Y. Fourier-based augmentation with applications to domain generalization. *Pattern Recognition*, 139:109474, 2023.

Xu, Z., Wu, K., Che, Z., Tang, J., and Ye, J. Knowledge transfer in multi-task deep reinforcement learning for continuous control. *Advances in Neural Information Processing Systems*, 33:15146–15155, 2020.

Yamagata, T., Khalil, A., and Santos-Rodriguez, R. Q-learning decision transformer: Leveraging dynamic programming for conditional sequence modelling in offline rl. In *International Conference on Machine Learning*, pp. 38989–39007. PMLR, 2023.

Yang, R., Xu, H., Wu, Y., and Wang, X. Multi-task reinforcement learning with soft modularization. *Advances in Neural Information Processing Systems*, 33:4767–4777, 2020.

Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., and Finn, C. Gradient surgery for multi-task learning. *Advances in Neural Information Processing Systems*, 33: 5824–5836, 2020a.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., and Levine, S. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pp. 1094–1100. PMLR, 2020b.

Zhang, R., Fan, Z., Xu, Q., Yao, J., Zhang, Y., and Wang, Y. Grace: A generalized and personalized federated learning method for medical imaging. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 14–24. Springer, 2023a.

Zhang, R., Fan, Z., Yao, J., Zhang, Y., and Wang, Y. Domain-inspired sharpness aware minimization under domain shifts. In *The Twelfth International Conference on Learning Representations*, 2023b.

Zhou, Y., Li, H., Du, S., Yao, J., Zhang, Y., and Wang, Y. Low-rank knowledge decomposition for medical foundation models. *arXiv preprint arXiv:2404.17184*, 2024.

A. Detailed Environment

A.1. Meta-World

The Meta-World benchmark, introduced by Yu et al. (2020b), encompasses a diverse array of 50 distinct manipulation tasks, unified by shared dynamics. These tasks involve a Sawyer robot engaging with a variety of objects, each distinguished by unique shapes, joints, and connective properties. The complexity of this benchmark lies in the heterogeneity of the state spaces and reward functions across tasks, as the robot is required to manipulate different objects towards varying objectives. The robot operates with a 4-dimensional fine-grained action input at each timestep, which controls the 3D positional movements of its end effector and modulates the gripper's openness. In its original configuration, the Meta-World environment is set with fixed goals, a format that somewhat limits the scope and realism of robotic learning applications. To address this and align with recent advancements in the field, as noted in works by Sun et al. (2022); Yang et al. (2020), we have modified all tasks to incorporate a random-goal setting, henceforth referred to as MT50-rand. The primary metric for evaluating performance in this enhanced setup is the average success rate across all tasks, providing a comprehensive measure of the robotic system's adaptability and proficiency in varied task environments.

For the creation of the offline dataset, we follow the work by He et al. (2023a) and employ the Soft Actor-Critic (SAC) algorithm (Haarnoja et al., 2018) to train distinct policies for each task until they reach a state of convergence. Subsequently, we compile a dataset comprising 1 million transitions per task, extracted from the SAC replay buffer. These transitions represent samples observed throughout the training period, up until the point where each policy's performance stabilized. Within this benchmark, we have curated two distinct dataset compositions:

- **Near-optimal** dataset consisting of the experience (100M transitions) from random to expert (convergence) in SAC-Replay.

- **Sub-optimal** dataset consisting of the initial 50% of the trajectories (50M transitions) of the near-optimal dataset for each task, where the proportion of expert data decreases a lot.

A.2. Unseen Tasks

In our evaluation, we apply our approach to a diverse array of meta-RL control tasks, each offering distinct challenges to assess the performance and generalization capabilities of our model. The tasks are detailed as follows:

- **Cheetah-dir**: This task involves two distinct directions: forward and backward. The objective is for the cheetah agent to achieve high velocity in the assigned direction. The evaluation encompasses both training and testing sets, covering these two directions comprehensively to gauge the agent's performance effectively.

- **Cheetah-vel**: Here, the task defines 40 unique sub-tasks, each associated with a specific goal velocity, uniformly distributed between 0 and 3 m/s. The agent's performance is assessed based on the l_2 error relative to the target velocity, with a penalty for deviations. For testing, 5 of these tasks are selected, while the remaining 35 are used for training purposes.

- **Ant-dir**: This task comprises 50 different sub-tasks, each with a goal direction uniformly sampled in a two-dimensional plane. The agent, an 8-jointed ant, is incentivized to attain high velocity in the designated direction. Of these, 5 tasks are earmarked for testing, with the rest allocated for training.

By evaluating our approach on these diverse tasks, we can assess its performance and generalization capabilities across different control scenarios. The generalization ability of our approach is rigorously tested by examining the distribution of tasks between the training and testing sets, as outlined in Table 5. This experimental setup, as described in Section 5, adheres to the divisions specified, ensuring consistency in our evaluation and facilitating a thorough assessment of our approach's adaptability and effectiveness across varied control tasks.

B. Hyper-parameters and Resources

This section elaborates on the specifics of the training regimen implemented in our study. During the training phase, tasks are randomly selected for model refinement. The configuration for each training iteration is meticulously set, with a batch size of 8 and the utilization of the Adam optimizer, operating at a learning rate of 1e-4. The total number of training steps is

established at 10 million. We build our policy as a Transformer-based model, which is based on minGPT open-source code [4]. The specific model parameters and hyper-parameters utilized in our training process are outlined in Table 6.

Table 5. Training and testing task indexes when testing the generalization ability in meta-RL tasks

Cheetah-dir	
Training set of size 2	[0,1]
Testing set of size 2	[0.1]
Cheetah-vel	
Training set of size 35	[0-1,3-6,8-14,16-22,24-25,27-39]
Testing set of size 5	[2,7,15,23,26]
Ant-dir	
Training set of size 45	[0-5,7-16,18-22,24-29,31-40,42-49]
Testing set of size 5	[6,17,23,30,41]

Table 6. Hyper-parameters of HarmoDT in our experiments.

Parameter	Value
Number of layers	6
Number of attention heads	8
Embedding dimension	256
Nonlinearity function	ReLU
Batch size	8
Context length K	20
Dropout	0.1
Learning rate	1.0e-4
Total rounds	1e6
Sparsity S	0.2
minimum of mask changing η_{min}	0
maximum of mask changing η_{max}	100
balance factor λ	10
mask changing interval t_m	5000
threshold in Equation 13	25

Training Resources. We use NVIDIA GeForce RTX 3090 to train each model. The training duration for each model is typically observed to be 36 hours in the 50 tasks setting, while it takes approximately 24 hours in the 30 tasks setting. However, since each environment needs to be trained three times with different seeds, the total training time is usually multiplied by three.

C. ERK initialization

This section elucidates the utilization of the Erdős-Rényi Kernel (ERK), as proposed by Evci et al. (2020), for initializing the sparsity in each layer of the model. ERK tailors sparsity distinctively for different layers. In convolutional layers, the proportion of active parameters is determined by $\frac{n_{l-1}+n_l+w_l+h_l}{n_{l-1} \times n_l \times w_l \times h_l}$, where n_{l-1}, n_l, w_l, and h_l represent the number of input channels, output channels, and the kernel's width and height in the l-th layer, respectively. For linear layers, the active parameter ratio is set to $\frac{n_{l-1}+n_l}{n_{l-1} \times n_l}$, with n_{l-1} and n_l indicating the number of neurons in the $(l-1)$-th and l-th layers. ERK

---

[4] https://github.com/karpathy/minGPT

ensures that layers with fewer parameters maintain a higher proportion of active parameters.

D. Baselines

We compare our proposed HarmoDT with the following baselines.

   i. **MTBC**. We extend Behavior cloning (BC) to multi-task offline policy learning via network scaling and a task-ID conditioned actor that is similar to MTIQL.

   ii. **MTIQL**. We extend IQL (Kostrikov et al., 2021) with multi-head critic networks and a task-ID conditioned actor for multi-task policy learning. The TD-based baselines are used to demonstrate the effectiveness of conditional generative modeling for multi-task planning.

   iii. **MTDIFF-P** (He et al., 2023a). MTDIFF-P is a diffusion-based method that incorporates Transformer backbones and prompt learning for generative planning and data synthesis in multitask offline settings.

   iv. **MTDT**. We extend the DT architecture (Chen et al., 2021) to learn from multitask data. Specifically, MTDT concatenates an embedding z and a state s as the input tokens, where z is the encoding of task ID. In evaluation, the reward-to-go and task ID are fed into the Transformer to provide task-specific information.

   v. **Prompt-DT** (Xu et al., 2022). Prompt-DT built on DT aims to learn from multi-task data and generalize the policy to unseen tasks. Prompt-DT generates actions based on the trajectory prompts and reward-to-go.

In addition to offline methods, our analysis also encompasses a comparison with several online methodologies to provide a comprehensive evaluation of our approach. These include:

   vi. **CARE** (Sodhani et al., 2021). This method utilizes additional metadata alongside a combination of multiple encoders to enhance task representation, offering a nuanced approach to multi-task learning.

   vii. **PaCO** (Sun et al., 2022). PaCO introduces a parameter compositional strategy, ingeniously recombining task-specific parameters to foster a more flexible and adaptive learning process.

   viii. **Soft-M** (Yang et al., 2020).This approach is centered around the development of a routing network, which adeptly orchestrates the soft combination of various modules, thereby facilitating more dynamic learning pathways.

   ix. **D2R** (He et al., 2023b). D2R innovatively employs disparate routing paths, enabling the selection of varying numbers of modules tailored to the specific requirements of each task, thereby enhancing the model's adaptability and efficiency.

E. More Ablations

This section comprehensively details the ablation study conducted on key hyper-parameters within our experimental framework. These hyper-parameters include η_{max}, which is integral to the cosine annealing process (with a fixed $\eta_{min} = 0$), the mask alteration frequency denoted as t_m, the overall sparsity parameter S, and the balance controller λ. Figure 6 presents an in-depth analysis of these hyper-parameters' impact on the performance of our model across 30 and 50 tasks within the Meta-world benchmark. This evaluation spans both near-optimal and sub-optimal settings, providing a comprehensive understanding of the hyper-parameters' influence under varied conditions. Remarkably, our approach consistently surpasses baseline models across a diverse range of hyper-parameter values. From this extensive analysis, we derive optimal settings for these parameters: a sparsity ratio S set to 0.2, an $\eta_{max} = 100$ value approximating 0.001% of the total weight count, a balance factor λ fixed at 10, and a mask changing interval t_m established at 5000 rounds. These recommended configurations are grounded in empirical evidence and are instrumental in achieving enhanced performance in multi-task learning scenarios.

F. Related Work

F.1. Offline Reinforcement Learning

Offline RL algorithms learn a policy entirely from this static offline dataset $\mathcal{D}$, without online interactions with environment (Levine et al., 2020). This paradigm can be precious in case the interaction with the environment is expensive or high-risk
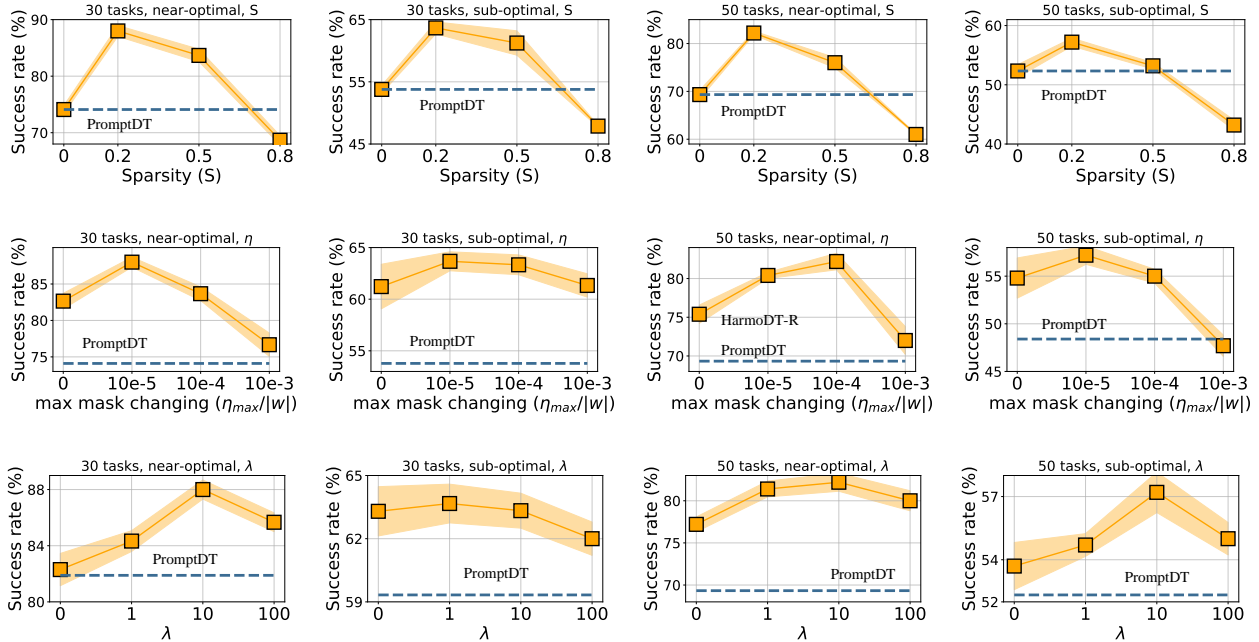
Figure 6. This figure presents an ablation study on critical hyper-parameters: sparsity ratio (S), maximum mask change (η_{max}), and balance ratio (γ). Displayed from left to right are the results for 30 tasks (near-optimal and sub-optimal) and 50 tasks (near-optimal and sub-optimal). Default settings are $\eta_{max} = 100$, S $= 0.2$, and $\lambda = 10$. Each ablation varies one parameter while others remain default.

(e.g., safety-critical applications). However as the learned policy might differ from the behavior policy, the offline algorithms must mitigate the effect of the *distribution shift*, which can result in a significant performance drop, as demonstrated in prior research (Fujimoto et al., 2019). Several previous works have utilized constrained or regularized dynamic programming to mitigate deviations from the behavior policy (Fujimoto & Gu, 2021; Kumar et al., 2020; Kostrikov et al., 2021).

Conditional sequence modeling approaches have been a promising direction for solving offline RL, which predicts subsequent actions from a sequence of past experiences, encompassing state-action-reward triplets. This paradigm lends itself to a supervised learning approach, inherently constraining the learned policy within the boundaries of the behavior policy and focusing on a policy conditioned on specific metrics for future trajectories (Chen et al., 2021; Hu et al., 2023a; 2024a; Yamagata et al., 2023; Hu et al., 2023b; 2024b; Meng et al., 2023).

Recently, there has been a growing interest in incorporating diffusion models into offline RL methods. This alternative approach to decision-making stems from the success of generative modeling, which offers the potential to address offline RL problems more effectively. Diffuser and its variants (Janner et al., 2022; Ajay et al., 2022; Chen et al., 2022; Wang et al., 2022) utilize diffusion-based generative models to represent policies or model dynamics, achieving competitive or superior performance across various tasks.

F.2. Multi-Task Reinforcement Learning

Multi-task RL aims to learn a shared policy for a diverse set of tasks, and there are many different approaches have been proposed in the literature (Xu et al., 2020; Yang et al., 2020; Sarafian et al., 2021; Sodhani et al., 2021; Zhou et al., 2024). One of the most straightforward approaches to MTRL is to formulate the multi-task model as a task-conditional one (Yu et al., 2020b), as commonly used in goal-conditional RL (Plappert et al., 2018). Conditional sequence modeling approaches, which consider handling multi-task problems, mainly rely on expert trajectories and entail substantial training expenses (Xu et al., 2022; Hu et al., 2023b; Lee et al., 2022). Diffusion model is also verified to have the potential to address the challenge of multi-task generalization in RL. MTDIFF (He et al., 2023a) extends the conditional diffusion model to be capable of solving multi-task decision-making problems and synthesizing useful data for downstream tasks.

Although these methods are simple and have shown some success in certain cases, one inherent limitation is the conflicting gradients among different data sources, phenomenon known as gradient conflicts and negative transfer in many fields, such as federated learning (Fan et al., 2022; 2024b; 2023; 2024a), domain generalization (Zhang et al., 2023a;b; Xu et al., 2023),

and multi-task learning (Yu et al., 2020a; Chen et al., 2020; Liu et al., 2021a). To mitigate conflicting gradient impacts in a multi-task context, several methodologies have been developed: PCGrad (Yu et al., 2020a) projects each task's gradient onto the orthogonal plane of another's, subsequently updating parameters using the mean of these projected gradients. Graddrop (Chen et al., 2020) employs a stochastic approach, randomly omitting certain gradient elements based on their conflict intensity. CAGrad (Liu et al., 2021a) manipulates gradients to converge towards a minimum average loss across tasks. In contrast, our method, HarmoDT, leverages gradient information in a fundamentally different manner. Instead of adjusting gradients post hoc as in these methods, we proactively utilize gradient data to inform the selective activation of parameters for each task through a masking mechanism. This direct intervention at the parameter level allows the model to update without the typical interferences found in gradient-level adjustments, fostering a more streamlined and potentially more efficacious optimization process.

On the other hand, D'Eramo et al. (2020) leverages the shared knowledge between multiple tasks by using a shared network followed by multiple task-specific heads. Yang et al. (2020) further extends this approach by softly sharing features (activations) from a base network among tasks, by generating the combination weight with an additional modularization network taking both state and task-id as input. Since the base and modularization networks take state and task information as input, there is no clear separation between task-agnostic and task-specific parts. PaCo (Sun et al., 2022) explores a compositional structure in the parameter space and distinguishes the task-agnostic and task-specific parts with different parameters, however, it still suffers the conflicting gradients within the shared parameters. Our method uses task-specific masks to find out the task-agnostic and task-specific parameters and dynamically update them to mitigate the conflicting gradients phenomenon, achieving state-of-the-art performance across various benchmarks.