Cost-aware Offline Safe Meta Reinforcement Learning with Robust In-Distribution Online Task Adaptation

Cong Guan
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University
Nanjing, China
guanc@lamda.nju.edu.cn

Ruiqi Xue
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University
Nanjing, China
ruiqixue@smail.nju.edu.cn

Ziqian Zhang
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University
Nanjing, China
zhangzq@lamda.nju.edu.cn

Lihe Li
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University
Nanjing, China
lilh@lamda.nju.edu.cn

Yi-Chen Li
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University
Nanjing, China
liyc@lamda.nju.edu.cn

Lei Yuan
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University;
Polixir Technologies
Nanjing, China
yuanl@lamda.nju.edu.cn

Yang Yu*
National Key Laboratory for Novel Software Technology, Nanjing University
School of Artificial Intelligence, Nanjing University;
Polixir Technologies
Nanjing, China
yuy@nju.edu.cn

ABSTRACT

Despite the gained prominence made by reinforcement learning (RL) in various domains, ensuring safety in real-world applications remains a significant challenge. Offline safe RL, which learns safe policies from pre-collected data, has emerged to address these concerns. However, existing approaches assume a single constraint mode and lack adaptability to diverse safety constraints. In real-world scenarios, we often find ourselves working with datasets gathered from various tasks, with the aim of developing a generalized policy capable of handling unknown tasks during testing. To deal with such offline safe *meta* RL problem, we introduce a novel framework called **COSTA**, which is designed to facilitate the learning of a safe generalized policy that can adapt and be transferred to unknown tasks during testing. COSTA addresses two key challenges in offline safe meta RL: First, it develops a cost-aware task inference module using contrastive learning to distinguish tasks based on safety constraints, mitigating the MDP ambiguity problem. Second, COSTA introduces a novel metric, Safe In-Distribution Score (SIDS), to assess the in-distribution degree of trajectories, out of the consideration of both reward maximization and cost constraint satisfaction. Trajectories collected with a safe exploration policy are filtered using SIDS for robust online task adaptation. Experimental results in a tailored benchmark suite within the Mujoco environments demonstrate that COSTA consistently balances safety and reward maximization, outperforming multiple baselines.

KEYWORDS

Offline Reinforcement Learning; Meta Learning; Safe Reinforcement Learning; Task Adaptation

*Yang Yu is the corresponding author.

This work is licensed under a Creative Commons Attribution International 4.0 License.

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

ACM Reference Format:
Cong Guan, Ruiqi Xue, Ziqian Zhang, Lihe Li, Yi-Chen Li, Lei Yuan, and Yang Yu. 2024. Cost-aware Offline Safe Meta Reinforcement Learning with Robust In-Distribution Online Task Adaptation. In *Proc. of the 23rd International*

Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024, IFAAMAS, 17 pages.

1 INTRODUCTION

Learning optimal policies in simulators via reinforcement learning (RL) has gained significant traction in various fields [38, 40] and exhibits tremendous potential across diverse real-world applications, e.g., sequential recommendation systems [30, 46] and robotic control [12, 15]. Nevertheless, safety concerns are usually raised in the real world where various constraints exist. For example, autonomous vehicles should pay extra attention to traffic regulations to avoid accidents such as high-speed collisions [16, 39]. In such cases, solely maximizing long-term returns might lead to illegal actions that induce damage to safety. Meanwhile, the need for trial-and-error of RL during training is unbearable in the real world, especially when exploratory behavior is costly or unsafe [43]. Consequently, offline safe RL (a.k.a. offline constrained RL) [22, 41], which learns a safe policy within a pre-collected dataset without additional interactions with the environment, has garnered significant attention in recent years.

Recent works on offline safe RL mostly model the environment as a Constrained Markov Decision Process (CMDP) [2] and solve the constraint optimization problem while maintaining a conservative policy to mitigate the distribution shift issue inherent in offline RL. Le et al. [18] estimate safety constraints via off-policy evaluation, yet it assumes a sufficient state-action coverage of the dataset and is limited to discrete action space. To evaluate constraint values accurately, Constraints Penalized Q-learning (CPQ) [41] learns an extra cost critic and makes out-of-distribution (OOD) actions unsafe. CPQ achieves impressive performance by penalizing unsafe actions and updating the policy exclusively on safe actions.

The aforementioned approaches train policies that maximize returns within a *singular* constraint mode for both training and testing. In practical situations, however, we may need to develop a policy that generalizes to various scenarios with *different* safety constraints and swiftly adapts to unknown tasks during testing. Some works on offline meta RL address this challenge by employing a context encoder to learn distinct task representations from various tasks and utilizing latent embeddings as augmentations to the policy's input [21, 44]. Navigating the intersection of offline safe RL and offline meta RL poses new non-trivial challenges. On one hand, learning a safe policy from an offline dataset requires a substantial number of trajectories sampled by the safe policy. However, varying safety constraints across different tasks inherently lead to distinct state-action distributions for these safe policies, increasing the risk of the context encoder overfitting to state-action distributions and overlooking task-specific cost information. This can lead to biased task inferences, a problem referred to as the MDP ambiguity [7, 20]. On the other hand, reducing safety costs necessitates a shorter *burn-in* or adaptation period during testing, making an efficient exploration policy and robust task inference indispensable.

For the mentioned issues, this paper studies the problem of learning a safe generalized policy from offline datasets collected across different tasks, with the capability to adapt and be transferred to unknown tasks during testing. Specifically, we propose a novel framework of **C**ost-aware **O**ffline **S**afe Meta RL with Robust In-Distribution Online **T**ask **A**daptation (**COSTA**). To learn a context encoder that distinguishes different tasks based on safety constraints rather than state-action distributions, we train a cost-aware task inference module using contrastive learning with positive and negative examples generated through supervised cost relabeling. We also observe that in-distribution adaptation trajectories provide reliable task inferences and improved policy evaluation. Accordingly, COSTA introduces a novel metric named as **S**afe **I**n-**D**istribution **S**core (**SIDS**) to assess the in-distribution degree of trajectories. SIDS strikes a balance between safety satisfaction and reward maximization. We collect trajectories using a safe but efficient exploration policy and employ SIDS to filter out OOD trajectories, thereby achieving a robust and swift online task adaptation. Experiments in our developed benchmarks, tailored to offline safe meta RL, justify that COSTA consistently balances safety satisfaction and reward maximization with robust online adaptation, outperforming all baselines.

Our main contributions are threefold as follows:

- To the best of our knowledge, this is the first time that offline safe meta RL problems are formally formulated and explored.
- We propose a novel framework for learning a safe generalized policy from offline datasets collected across different tasks, which incorporates a cost-aware context encoder and robust safe in-distribution online adaptation.
- Additionally, we develop an offline safe meta RL benchmark suite within the Mujoco environments and verify that COSTA consistently learns to maximize rewards while satisfying safety constraints and outperforms all baselines.

2 RELATED WORK

Offline Safe RL. Some recent works have focused on offline safe RL problems, considering the expense of trial-and-error in the real world. Constrained Batch Policy Learning [18] applies Fitted Q Iteration into policy optimization and evaluates the safe constraints via off-policy evaluation. COPO [27] and COptiDICE [19] both utilize stationary distribution correction (DICE) technique to derive a cost-feasible policy solution set. Xu et al. [41] propose to train an extra cost critic and disable the policy update on unsafe and OOD actions, which achieves impressive performance under safety constraints. They also explore the combination of offline RL methods and Lagrangian-based approaches to ensure safety. Sequential modeling approaches have also been applied to safe RL to flexibly handle different safety thresholds via cost-related tokens [23, 45]. Liu et al. [22] introduce a comprehensive benchmarking platform tailored to offline safe RL, allowing evaluation and comparison between different algorithms. However, none of the aforementioned methods takes meta setting into consideration.

Offline Meta RL. Context-based [21] meta RL framework [4] and online task adaptation are utilized in most offline meta RL methods. FOCAL [21] and MACAW [25] apply distance metric learning to derive representations that distinguish different tasks based on collected trajectories but neglect the influence of state-action distribution mismatch due to behavior policy. To tackle such an MDP ambiguity problem, Li et al. [20] and Dorfman et al. [6] assume

a known reward function and relabel the transitions to learn robust task representations. CORRO [44] uses mutual information maximization to derive the contrastive learning objective and dispenses with the assumption via generative modeling and reward randomization. IDAQ [14] proposes a novel adaptation frame by generating and distinguishing in-distribution episodes via an uncertainty quantification. Nevertheless, methods above all fail to handle small state-action overlap between tasks and can not guarantee safety and robustness during the adaptation period.

3 PROBLEM SETTING

We consider the problem of offline safe meta reinforcement learning in deterministic environments such as MuJoCo [36] under fully observable Constrained Markov Decision Process (CMDP) [2].

A CMDP can be defined as a tuple $\langle S, A, r, c, P, \gamma, b\rangle$, where S and A represent the state space and the action space, $r : S \times A \mapsto [-R_{\max}, R_{\max}]$ and $c : S \times A \mapsto \{0, 1\}$ denote the reward and cost functions. $P : S \times A \times S \mapsto [0, 1]$ is the transition probability function, $\gamma \in (0, 1)$ is the discount factor, and b represents the safety constraint limit. A policy $\pi : S \mapsto \Delta(A)$ maps states to action distributions. Under the given policy π, the cumulative reward can be expressed as $R(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t, a_t)\right]$, where $\tau = (s_0, a_0, s_1, a_1, \dots)$ denotes a trajectory and $\tau \sim \pi$ indicates that the distribution of trajectories is induced by policy π. Similarly, the cumulative cost is given by $C(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{\infty}\gamma^t c(s_t, a_t)\right]$. The objective of solving a CMDP is to learn a policy that maximizes cumulative reward while adhering to safety constraints, which can be represented as:

$$\max_{\pi} \quad R(\pi) \qquad s.t. \quad C(\pi) \le b. \tag{1}$$

Similar to previous offline meta RL formulations, we first assume a task distribution $\Delta(\mathcal{T})$, where each task $\mathcal{T}$ is a CMDP, and $\Delta(\cdot)$ is an unknown distribution. The agent cannot learn by interacting with the environment but only has access to a static dataset $\mathcal{D}$, which contains data from T different tasks sampled from $\Delta(\mathcal{T})$. For each task i, its collected dataset is $\mathcal{D}_i = \{(s_j, a_j, r_j, c_j, s'_j)\}_{j=1}^{|\mathcal{D}_i|}$, generated by the corresponding behavior policy $\beta_i(a|s)$. For simplification, we focus only on tasks that share the same state-action space, transition, and reward function while having different cost functions. Additionally, we assume that during the training process, each task in $\mathcal{D}$ is sampled with the same probability. In this scenario, the meta-training procedure is transformed into maximizing the average objective across all training tasks:

$$\hat{\theta}_{meta} = \arg\max_{\theta} \frac{1}{T}\sum_{i=1}^{T}\mathbb{E}[\mathcal{L}_i(\theta)], \tag{2}$$

where $\mathcal{L}_i(\theta)$ is the objective evaluated on transition samples drawn from task $\mathcal{T}_i$ and θ is the parameter of the learning policies.

In the online deployment phase, the meta-trained policy will be confronted with a test task, denoted as $\mathcal{T}_{\text{test}} \sim \Delta(\mathcal{T})$. To better simulate the real-world application scenarios, we assume that we do not know the exact task id of $\mathcal{T}_{\text{test}}$. Therefore, the ultimate objective of offline safe meta RL is to correctly identify the test task by interacting with the online environment while striking a balance between reward maximization and safety constraint satisfaction.

4 METHOD

This section gives a detailed description of our proposed COSTA, a novel algorithm for offline safe meta reinforcement learning. As visually depicted in Figure 1, Section 4.1 illustrates the process of COSTA's context encoder learning, Section 4.2 presents COSTA's procedure for safe online adaptation, while Section 4.3 introduces COSTA's overall algorithm.

4.1 Cost-aware Context Learning

In order to enable the policy to make distinct decisions for specific tasks, we here introduce a context-based policy architecture like the context-based meta-learning approaches [8, 28]. After introducing context-based approaches to offline safe meta RL, the offline context of the problem can be formulated as solving a task-augmented CMDP (TA-CMDP). The formal definitions will be provided in Appendix B. Specifically, we employ an inference network $q_\phi(z|e)$, parameterized by ϕ, to extract task-relevant representations based on the given context e. Subsequently, we incorporate an attention-based deterministic neural network as our chosen encoder architecture. In specific, the encoder $q_\phi = (v_\phi, m_\phi)$ consists of two distinct multi-layer perceptrons (MLP), where v_ϕ and softmax(m_ϕ) stand for attention value and attention weight, respectively. Given context $e = \{(s_k, a_k, c_k, s'_k)\}_{k=1}^{n}$, we can calculate the task information as:

$$z = \sum_{j=1}^{n} \text{softmax}(\{m_\phi(s_k, a_k, c_k, s'_k)\}_{k=1}^{n})_j \cdot v_\phi(s_j, a_j, c_j, s'_j). \tag{3}$$

As simple MLP networks cannot guarantee the distinguishment among different tasks, we here design auxiliary regularizers to train the context encoder to distinguish between different tasks based on their task-relevant features. On the one hand, we can maximize the distance between context embeddings of different tasks by employing the negative-power variant of the contrastive loss method derived from **distance metric learning (DML)** [31] like the application in offline mete RL setting [21]. In specific, given a pair of contexts e_i, e_j and their labels $y_i, y_j \in \{1, 2, \dots, T\}$, the formulation is expressed as follows:

$$\begin{aligned}\mathcal{L}_{\text{dml}}(e_i, e_j; q) = {} & \mathbb{1}\{y_i = y_j\}||z_i - z_j||_2^2 \\ & +\lambda_1 \mathbb{1}\{y_i \neq y_j\}\frac{1}{||z_i - z_j||_2^p + \epsilon},\end{aligned} \tag{4}$$

where $\mathbb{1}\{\cdot\}$ is the indicator function, $z_i = q_\phi(e_i)$, $z_j = q_\phi(e_j)$, λ_1 is a predefined coefficient, ϵ is a small constant to avoid division by zero, and we set $p = 2$ to better conform to local topology and similarity relationships.

The mentioned optimization object makes sense in the offline meta setting. Unfortunately, in offline safe meta RL, the unknown data-collection behavior policies $\{\beta_i\}_{i=1}^{T}$, satisfying safety constraints, can vary significantly across different tasks, resulting in a substantial divergence in state-action distributions between offline datasets. To solve this problem, we here introduce the **cost contrastive loss** to explicitly emphasize the distinctions existing between tasks within (s, a, c, s') at the cost level by prioritizing the task-specific features that truly matter, mitigating the risk of erroneous inferences resulting from different state-action distributions.

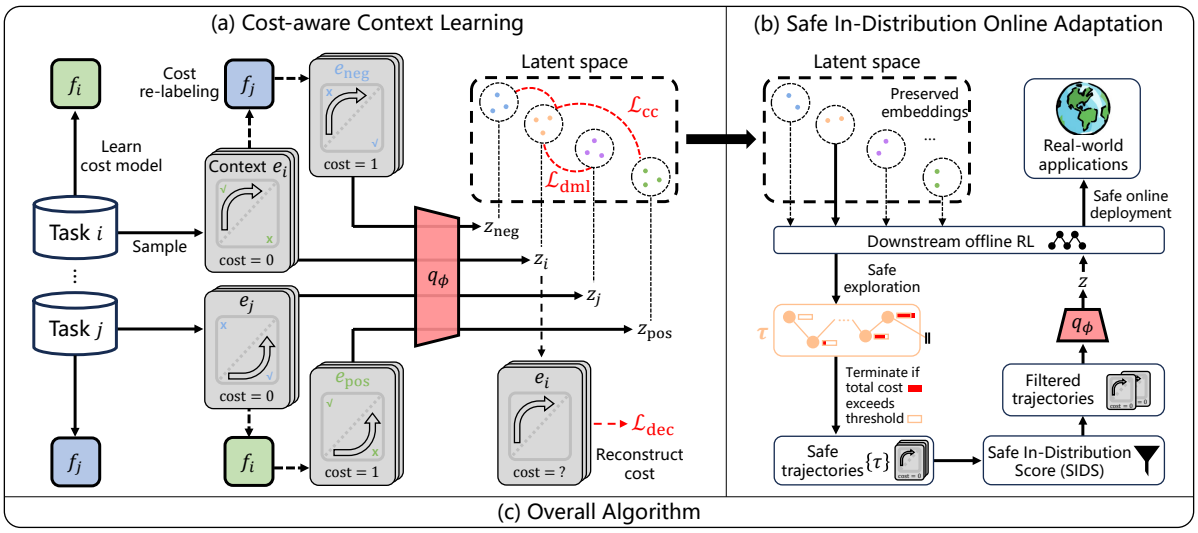


Figure 1: Structure of COSTA. (a) During context encoder learning, COSTA employs contexts from various tasks to conduct distance metric learning. Simultaneously, it utilizes a cost re-labeling method based on the cost model to generate negative and positive samples for cost contrastive learning. (b) During online adaptation, COSTA will utilize a small number of context embeddings saved during the offline training process to perform policy rollouts within the environment. The trajectories obtained from these rollouts will be truncated based on their cumulative cost. Finally, an in-distribution assessment method proposed by us will be used to select the optimal trajectories as context for evaluation.

Specifically, we first utilize supervised learning to acquire a cost model f_i for each task, where $i = 1, \ldots, T$. Due to the binary nature of the cost function, each model inherently operates as a binary classification model, which substantially simplifies the learning process when compared to conventional reward models. We first sample the target task $\mathcal{T}_{\text{tgt}} \in \mathcal{T}$ and its corresponding context $e = \{(s_k, a_k, c_k, s'_k)\}_{k=1}^n$ from the buffer. Afterward, we construct negative and positive samples, respectively:

- To ensure that the context encoder can distinguish between different tasks under the same state-action distribution, we generate negative samples by replacing only the cost value in the context $e^- = \{(s_k, a_k, c_k, s'_k)\}_{k=1}^n$. Specifically, we sample a different task $\mathcal{T}_{\text{neg}} \in \mathcal{T}$ and employ the trained cost model to re-label the cost as $c_k^- = f_{\mathcal{T}_{\text{neg}}}(s_k, a_k, s'_k)$. This finishes the generation of negative samples $e_{\text{neg}} = \{(s_k, a_k, c_k^-, s'_k)\}_{k=1}^n$.
- We then generate positive samples so that the context encoder can identify the target task based on the cost value even under different state-action distributions. we sample another task $\mathcal{T}_{\text{pos}} \in \mathcal{T}$ and its context $e^+ = \{(s_k, a_k, c_k, s'_k)\}_{k=1}^n$ from the offline dataset. We then re-label c_k using the cost model $f_{\mathcal{T}_{\text{tgt}}}$, resulting in $c_k^+ = f_{\mathcal{T}_{\text{tgt}}}(s_k, a_k, s'_k)$. Consequently, we obtain positive samples $e_{\text{pos}} = \{(s_k, a_k, c_k^+, s'_k)\}_{k=1}^n$.

Combining the generated samples and incorporating them into the contrastive loss of DML, we arrive at the following expression:

$$\mathcal{L}_{\text{cc}}(e, e_{\text{pos}}, e_{\text{neg}}; q) = ||z - z_{\text{pos}}||_2^2 + \lambda_2 \frac{1}{||z - z_{\text{neg}}||_2^p + \epsilon}, \tag{5}$$

where $z = q_\phi(e), z_{\text{pos}} = q_\phi(e_{\text{pos}}), z_{\text{neg}} = q_\phi(e_{\text{neg}})$, λ_2 is a predefined coefficient, ϵ is a small constant to avoid division by zero.

In addition to the contrastive loss discussed above, we further introduce an auxiliary **decoder reconstruction loss** to ensure that context embeddings contain sufficient information for predicting the cost associated with a transition. The loss function is defined as follows:

$$\mathcal{L}_{\text{dec}}(e; q, p) = -\frac{1}{n}\sum_{i=1}^{n}(c_i \log(p_\psi(s_i, a_i, s'_i, z)) + (1 - c_i)\log(1 - p_\psi(s_i, a_i, s'_i, z))), \tag{6}$$

where $z = q_\phi(e)$, and p_ψ represents a binary neural network parameterized by ψ, which takes (s, a, s', z) as input. Overall, the total loss function for training the context encoder is as follows:

$$\mathcal{L}_{\text{enc}} = \mathcal{L}_{\text{dml}} + \alpha_{\text{cc}}\mathcal{L}_{\text{cc}} + \alpha_{\text{dec}}\mathcal{L}_{\text{dec}}, \tag{7}$$

where α_{cc} and α_{dec} are hyper-parameters that control the balance between the three objectives. It is worth noting that the presence of substantial errors within the cost models leads to a notable reduction in the quality of negative samples. We will adjust α_{cc} based on the model training error in the experiments.

4.2 Safe In-Distribution Online Adaptation

The introduction of cost-aware context learning enables COSTA to mitigate the impact of the behavior policy distributions on task identification. However, the contexts out of distribution may lead to biased task inference during online task adaptation, and utilizing safe in-distribution trajectories as contexts for testing will greatly improve the robustness. Traditional online adaptation methods like Thompson sampling [33] are inefficient in generating in-distribution trajectories, especially in safe meta scenarios, where

different tasks might correspond to different distributions of trajectories due to significantly different safety constraints. Accordingly, we randomly sample m contexts from the offline buffer for each task and derive $T \times m$ context embeddings $\{\bar{z}^i_{1:m}\}^T_{i=1}$ as additional preserved information after finishing the offline training. The meta-trained policy is capable of generating in-distribution trajectories belonging to task $i \in \mathcal{T}$ by taking the embedding $\bar{z}^i_k$ as input, $\forall k = 1, ..., m$. It is worth noting that $m \leq 10$ is sufficient in the typical scenarios. For larger numbers in complex settings, we leave it for future work.

Although we could sequentially generate trajectories with diverse distributions by augmenting the embeddings into the input of the meta-trained policy, we still need to select those that better align with the test task's in-distribution trajectories. For instance, we can derive a safe and well-performed policy under task $i \in \mathcal{T}$ by taking $\bar{z}^i_k$ as input, $\forall k = 1, ..., m$, but it will fail to guarantee the safety under the different test task $j \in \mathcal{T}$. To address this issue, we first introduce a measurement for assessing the in-distribution degree of trajectories and then propose a trajectory truncation method to ensure safety during exploration.

Given a trajectory τ, assuming its reward return is $R(\tau)$ and its cost return is $C(\tau)$, then its **Safe In-Distribution Score (SIDS)** is defined as follows:

$$\text{SIDS}(\tau) = \begin{cases} R(\tau) & C(\tau) \leq b + \epsilon_1, R(\tau) \geq \delta \\ -C(\tau) & C(\tau) > b + \epsilon_1, R(\tau) \geq \delta \\ -K & R(\tau) < \delta \end{cases}, \tag{8}$$

where b is the cost threshold defined in CMDP, ϵ_1, δ, and K are adjustable hyperparameters, with the constraint that $K > \max_\tau C(\tau)$.

Equation (8) divides the overall SIDS value space into three exclusive regions, wherein trajectories conforming to task safety constraints and yielding superior reward returns are associated with elevated SIDS values. From another perspective, it implies that a trajectory is more likely to be in-distribution if it attains a higher cumulative reward before its cumulative cost violates safety constraints. Therefore, during the exploration process of online adaptation, once the cumulative cost of the trajectory induced by the policy exceeds $b + \epsilon_2$, where $\epsilon_2 \geq \epsilon_1$, we terminate the episode and truncate the trajectory immediately to ensure safe exploration.

To effectively evaluate such collected truncated trajectories with different lengths, we extend SIDS to $\text{SIDS}_{\text{trunc}}$ by replacing τ with $\tau_{1:t_{\text{tc}}}$, where t_{tc} is the variable satisfying the following conditions:

$$t_{\text{tc}} = \min\{\arg\min_t C(\tau_{1:t}) \geq b + \epsilon_2, |\tau|\}. \tag{9}$$

The detailed expression of $\text{SIDS}_{\text{trunc}}$ is provided in Appendix B. Among all truncated trajectories with a safety guarantee, we select the top l trajectories with the highest $\text{SIDS}_{\text{trunc}}$ to derive the final context embedding z^* for deployment.

4.3 Overall Algorithm

With the learned context encoder and a safety-guaranteed online adaptation procedure, we can apply COSTA to any downstream offline meta reinforcement learning method to learn a safe and efficient policy. In practice, we implement COSTA on CPQ [41], a typical offline safe RL paradigm, by incorporating the cost-aware context learning and safe in-distribution online adaptation into it. CPQ learns an extra cost critic Q_c to evaluate the cumulative cost value, and we concatenate the context embedding of the training task to its input to derive the optimization objective:

$$\min_{Q_c} E_{s,a,s' \sim \mathcal{D}, z \sim q_\phi(z|e)}[(Q_c(s,a,z) - \mathcal{B}^\pi Q_c(s,a,z))^2] - \alpha_{\text{cpq}} E_{s \sim \mathcal{D}, a \sim v, z \sim q_\phi(z|e)}[Q_c(s,a,z)], \tag{10}$$

where $\mathcal{B}^\pi$ represents the Bellman operator, v is a learned distribution to mimic OOD(Out-of-Distribution) action distribution, z is the context embedding of given context e, and α_{cpq} is an adjustable weight used to control the conservatism of the cost critic, q_ϕ is our context encoder. By optimizing Equation 10, the cost critic Q_c learns to estimate the cost value accurately and penalize OOD actions to achieve conservatism.

Additionally, as the cost critic is distorted, we update the policy π by only maximizing reward critic values Q_r. Considering the safety and conservatism issues, CPQ defines Constrained Penalized Bellman Operator $\mathcal{B}^\pi_p$ to make sure that the reward values of non-safe or OOD transitions are set to be zero:

$$\mathcal{B}^\pi_p Q_r(s,a,z) = r + \gamma \mathbb{E}_{a' \sim \pi}[\mathbb{1}(Q_c(s',a',z) \leq b) Q_r(s',a',z)]. \tag{11}$$

Based on Equation 11, the optimization objective of the reward critic and policy can be expressed as follows:

$$\min_{Q_r} \mathbb{E}_{s,a,s' \sim \mathcal{D}, z \sim q_\phi(z|e)}[(Q_r(s,a,z) - \mathcal{B}^\pi_p Q_r(s,a,z))^2]. \tag{12}$$

$$\max_{\pi \in \Delta_{|S|}} \mathbb{E}_{s \sim \mathcal{D}, a \sim \pi(\cdot|s), z \sim q_\phi(z|e)}[\mathbb{1}(Q_c(s,a,z) \leq b) Q_r(s,a,z)]. \tag{13}$$

After learning the CPQ policy and the context encoder in an offline manner, we can apply the online adaptation procedure to safely deploy COSTA in the online tasks. The pseudo codes of the overall training process and adaptation process are provided in detail in Appendix A.

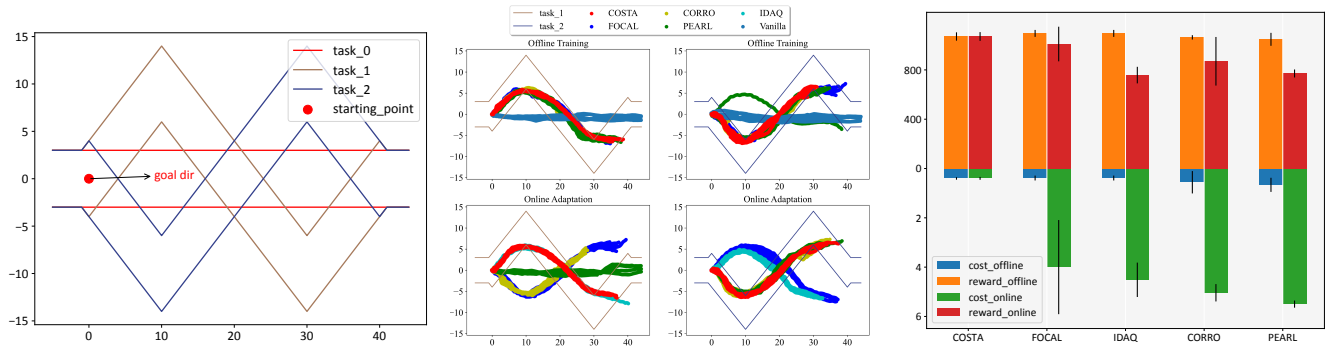
5 EXPERIMENTS

In this section, we conduct experiments in six different MuJoCo environments, aiming to address the following questions: (1) What happens when a single-task offline safe RL algorithm simultaneously learns safety constraints for multiple tasks, and can COSTA accurately identify different safe tasks in both offline training and online adaptation (Section 5.2) ? (2) Can COSTA surpass other baseline algorithms in terms of safety performance across various meta environments (Section 5.3) ? (3) Does the policy learned by COSTA demonstrate generalizability and transferability to unseen tasks (Section 5.4) ? (4) What contributions do the different components of COSTA make to its performance (Section 5.5) ?

For the evaluation, we compare COSTA with multiple baselines. All experimental results are averaged over five random seeds and are accompanied by standard deviation information. Details regarding experimental environments and network architecture parameters will be provided in Appendix C and D, respectively.

5.1 Baselines and Environments

To completely evaluate the performance of COSTA, we introduce the following offline safe meta RL methods. First of all, we consider **Vanilla**, which simply applies CPQ to learn a single non-context-based policy from all tasks to investigate the impact of COSTA's



(a) AntDir (b) Trajectory Projection (c) Performance Comparisons

Figure 2: Motivated example in AntDir. (a) An agent starts from the left part and has different safety constraints (zones) in different tasks. (b) The projections of trajectories of different methods during both offline training and online adaptation in 2 tasks. (c) A comparative analysis of the overall performance of meta-policies obtained through various algorithms.

overall framework. Then, to validate the impact of cost-aware context learning, we compare COSTA with **FOCAL** [21], which uses only distance metric learning loss between offline data to train the context encoder. While **CORRO** [44] additionally designs a bi-level context encoder and generates samples to obtain robust task representations. Besides, **PEARL** [28] learns a probabilistic fully-connected context encoder to capture the task identifications for meta RL. Next, we study the online adaptation process by comparing COSTA with **IDAQ** [14], which introduces in-distribution trajectory filtering and uses only in-distribution trajectories generated by Thompson sampling as context, and we re-implement its return-based trajectory filtering as SIDS-based trajectory ranking just like COSTA. We additionally implement **FOCAL_oracle**, which has access to the test task index to study if COSTA can achieve comparable performance with the baseline having extra task information and data. It's worth noting that, to ensure the fairness of the experiments, all of the above methods are re-implemented based on the offline safe RL method CPQ.

We consider multiple benchmarks based on MuJoCo robots and tasks from the safety benchmark Safety-Gymnasium [13] and other works [10]. In **AntDir**, the Ant robot receives rewards while moving in a specified direction, and different winding routes serve as safety zones for different tasks, simulating various road configurations found in real-world environments. **CheetahVel** encourages the Half-Cheetah robot to move as fast as possible, but exceeding specific speed limits for different tasks incurs costs. In **AntWalk** or **CheetahWalk**, the Ant and Half-Cheetah robots are no longer required to move in a specific direction. They receive rewards for moving in any direction away from the origin, and different tasks have roadways in various quadrants as safety zones. In **AntCircle**, the Ant robot is directed to traverse a circular trajectory. However, under different tasks, safety zones are defined by areas enclosed by two parallel lines characterized by differing angular orientations. Similar to AntDir, **AntGoal** incentivizes the Ant robot to navigate towards a predefined target point, which may involve encountering various obstacles along the path. Importantly, the Ant robot will not be obstructed when entering obstacle areas; however, such moves incur associated costs.

5.2 Motivated example: Learning to identify tasks with efficient online adaptation

At first glance, we design a three-task safe RL environment AntDir (Figure 2(a)), where an agent starts from the left staring-point and receives rewards when moving towards the right, under different safety constraints for different tasks. Specifically, for task_0, it is considered safe when the agent's y-coordinate is within the range of -3 to 3. However, for task_1 and task_2, their safe regions are defined by the complex and winding boundaries formed by the brown and blue lines in the diagram. Such constraints pose significant challenges to the agent's safety performance learning. Furthermore, in this environment, the shared safe regions among different tasks are small and not continuous. Therefore, to enable the agent to learn safe and high-performance policies for all three tasks simultaneously, accurate task recognition is essential.

As depicted in Figure 2(b), the results of offline training reveal that Vanilla struggled to acquire a policy capable of simultaneously meeting multiple task constraints. This underscores the challenge of directly applying purely offline safe RL algorithms in scenarios involving multiple tasks. As for offline meta RL algorithms (i.e., FOCAL and CORRO) or off-policy meta RL algorithm PEARL modified with the offline safe algorithm CPQ, they demonstrate the ability to accurately identify tasks during offline training. However, they all fall short of correctly identifying all tasks during online adaptation, whether utilizing algorithms based on Thompson sampling or further enhancing them with in-distribution trajectory filtering techniques such as IDAQ. This highlights the need for special designs aimed at enhancing the robustness of the context encoder in task recognition for safe scenarios. The successful and unbiased task inference achieved by COSTA provides compelling evidence of the effectiveness of the combination of the cost-aware encoder and in-distribution online adaptation.

Figure 2(c) presents a comparison of the average final performance of context-based algorithms across all tasks, with cost normalized according to the constraint limit. COSTA does not only achieve impressive results during offline training but also stands out as the only algorithm capable of attaining online adaptation results comparable to those from offline training. In contrast, the baselines all exhibit a substantial decrease in reward and an increase in cost during online adaptation. These findings strongly underscore the effectiveness of COSTA.

5.3 Overall Performance Comparison

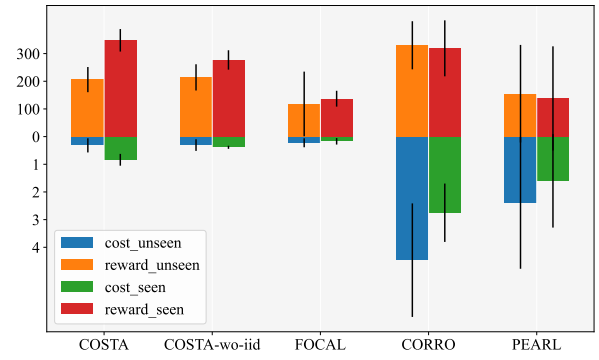
To validate the generality of COSTA's outstanding performance, we conduct extensive experiments in six distinct environments.

From the offline training results shown in Table 1, it is evident that Vanilla, which does not incorporate task embeddings, fails to obtain a policy that ensures safety across all tasks in all environments. This further emphasizes the inadequacy of traditional offline safe algorithms in addressing the meta offline safe problem. Meanwhile, none of the context-based baselines, including FOCAL, CORRO, and PEARL, manage to learn safe policies in all environments. Among them, FOCAL performs relatively well, ensuring safety in 5 out of 6 tasks, with the failed task slightly exceeding safety limits, which underscores the significance of distance metric learning in ensuring that task representations are as distinct as possible. Only our method, COSTA, stands out by achieving success in all environments. Upon observing the final averaged results, COSTA not only emerges as the safest approach but also outperforms another safe method, FOCAL, in terms of rewards.

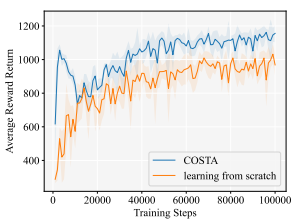
During the online adaptation phase, COSTA consistently maintains safety across all environments. In contrast, all other Thompson sampling-based methods fail to do so. Interestingly, we find that most baselines can ensure safety across all tasks in CheetaVel. This is because there exist some trajectories that will not violate constraints on any task. However, the reward returns they achieved are significantly lower than that of COSTA. Meanwhile, COSTA outperforms FOCAL_oracle in a broader range of environments, even without prior knowledge of the test task and the need to store the entire offline training buffer. COSTA only requires the preservation of a small set of context embeddings, which is more aligned with practical application scenarios. Additionally, the experimental results show that in some environments, IDAQ fails to achieve satisfying results, even inferior to FOCAL. This phenomenon emanates from IDAQ's inherent limitation in ensuring the availability of in-distribution trajectories.

5.4 Generalization and Transfer Studies

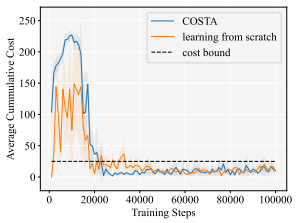
To comprehensively evaluate COSTA's generalization capability, we select CheetahVel as the benchmark since it exhibits continuity between different tasks. In CheetahVel, we randomly sample three tasks for training and leave out three tasks for generalization testing. The experimental results are depicted in Figure 3(a). It can be observed that the rewards obtained by FOCAL, CORRO, and PEARL in unseen tasks are similar to seen tasks despite the more stringent safety constraints associated with unseen tasks. Meanwhile, CORRO and PEARL also exhibit significant violations of safety. This suggests that these methods fail to generalize to unseen tasks when



(a) Task Generalization



(b) Task Transfer Reward



(c) Task Transfer Cost

Figure 3: (a) Comparison of generalization ability in CheetahVel, the left column represents the results for policies in unseen tasks, while the right column represents the results for policies in seen tasks. (b), (c) Training results of being transferred to unseen tasks in AntDir.

the number of training tasks is limited. In contrast, COSTA maintains safety and achieves a significantly different average reward return in unseen tasks compared to seen tasks. Additionally, to demonstrate that the reward difference in COSTA is not solely due to selecting the most conservative task's in-distribution trajectories using SIDS, we include the experimental results for COSTA-wo-iid, which utilizes the average embedding of all generated trajectories. Overall, these results confirm that COSTA possesses stronger generalization capabilities compared to other baseline algorithms.

For tasks that do not inherently possess generalizability, such as AntDir, we conduct extra experiments to assess COSTA's task transferability. Specifically, we utilize the previously learned context encoder and policy to train on contexts sampled from an offline dataset of unseen tasks. This is compared to training CPQ from scratch, as depicted in Figure 3(b) and Figure 3(c). It can be observed that during the initial stages of training, COSTA does not exhibit a significant advantage in training efficiency due to changes in the safety state space. However, as it progresses, COSTA consistently maintains a relatively high reward while ensuring safety. This demonstrates COSTA's capability for task transferability.

5.5 Ablation Studies and Visualization Analysis

In the first part of this part, we aim to investigate the influence of different components of COSTA on its performance, considering multiple variants. First, we study the impact of different modules in cost-aware context learning. Specifically, **COSTA-wo-dml** uses

Table 1: Final average reward and normalized cost return ± standard error in all test environments. The cost threshold is 1. The ↑ symbol denotes that the higher the reward, the better. The ↓ symbol denotes that the lower the cost (up to threshold 1), the better. Each value is averaged over all tasks, 10 evaluation episodes, and 5 random seeds. Black: Safe agents. Gray: Unsafe agents. Blue: Safe agent with the highest reward.

Task		COSTA		FOCAL		FOCAL_oracle		IDAQ		CORRO		PEARL		Vanilla	
		reward↑	cost↓	reward↑	cost↓	reward↑	cost↓	reward↑	cost↓	reward↑	cost↓	reward↑	cost↓	reward↑	cost↓
AntDir	Training	1071±33.75	0.40±0.05	1097±28.30	0.39±0.10	/	/	/	/	1064±17.65	0.56±0.45	1048±52.57	0.66±0.29	1529±83.85	2.59±0.15
	Adaptation	1071±35.70	0.40±0.06	1010±140.38	4.00±1.90	1103±28.08	0.44±0.05	757±67.16	4.51±0.70	870±197.22	5.04±0.35	771±32.35	5.50±0.14	/	/
CheetahVel	Training	354±31.70	0.89±0.12	254±189.34	0.65±0.37	/	/	/	/	329±56.61	1.74±0.89	169±177.03	1.70±1.41	136±195.84	1.73±1.35
	Adaptation	348±40.69	0.84±0.21	137±28.64	0.17±0.12	245±170.51	0.40±0.18	190±27.18	0.14±0.06	319±101.15	2.75±1.05	138±188.25	1.60±1.69	/	/
AntWalk	Training	495±17.98	0.12±0.17	506±5.02	0.06±0.04	/	/	/	/	496±8.32	0.10±0.11	528±12.03	7.07±0.73	523±16.61	7.93±0.26
	Adaptation	490±15.62	0.09±0.12	537±17.07	5.42±0.11	502±7.48	0.11±0.07	415±50.91	3.64±3.28	511±34.32	6.54±2.30	525±9.89	5.46±2.95	/	/
CheetahWalk	Training	260±72.97	0.00±0.00	234±103.93	1.19±2.38	/	/	/	/	263±175.02	6.64±6.43	216±126.89	0.40±0.80	502±108.14	14.20±0.02
	Adaptation	248±66.97	0.00±0.00	296±120.04	8.51±6.95	225±117.70	0.72±1.44	298±105.74	2.84±5.68	244±159.75	13.06±2.30	162±42.27	11.37±5.68	/	/
AntCircle	Training	1181±219.65	0.55±0.36	1019±351.95	1.01±0.66	/	/	/	/	1232±418.41	1.05±0.70	982±287.94	0.67±0.24	1869±812.58	3.21±0.53
	Adaptation	1147±118.63	0.58±0.35	727±278.45	2.78±1.11	923±396.39	0.92±0.90	730±107.51	2.02±1.1	1001±518.32	2.24±1.39	1217±484.25	2.44±0.86	/	/
AntGoal	Training	819±14.80	0.64±0.10	826±15.54	0.57±0.17	/	/	/	/	824±8.55	0.69±0.14	810±6.48	0.58±0.23	892±12.01	1.14±0.13
	Adaptation	827±20.86	0.75±0.21	851±18.92	2.28±1.02	824±19.33	0.60±0.17	852±13.26	2.33±1.15	810±40.78	2.9±0.67	853±24.28	1.57±0.74	/	/
Average	Training	697	0.43	656	0.65	/	/	/	/	701	1.80	626	1.85	909	5.13
	Adaptation	689	0.45	593	3.86	637	0.53	583	2.58	599	5.42	611	4.66	/	/

Table 2: Final average reward and normalized cost return ± standard error of ablations in two test environments.

Task		COSTA		COSTA-ts		COSTA-wo-iid		COSTA-wo-dml		COSTA-wo-cc	
		reward↑	cost↓	reward↑	cost↓	reward↑	cost↓	reward↑	cost↓	reward↑	cost↓
AntDir	Training	1071±33.75	0.40±0.05	/	/	/	/	1092±38.85	1.89±0.60	1088±36.56	0.45±0.06
	Adaptation	1071±35.70	0.40±0.06	1043±160.16	1.43±1.04	547±61.39	4.50±0.21	1161±84.74	1.36±0.65	1055±55.53	0.38±0.08
CheetahWalk	Training	260±72.97	0.00±0.00	/	/	/	/	273±86.59	0.00±0.00	197±136.41	2.83±3.84
	Adaptation	248±66.97	0.00±0.00	220±77.46	5.60±6.86	138±72.97	2.52±5.05	225±56.70	0.00±0.00	205±112.25	5.21±6.42

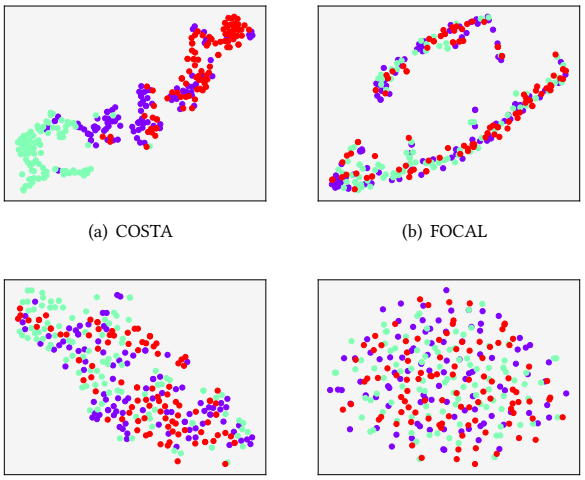
only $\mathcal{L}_{\text{cc}}$ and $\mathcal{L}_{\text{dec}}$ to train the encoder, while **COSTA-wo-cc** relies solely on $\mathcal{L}_{\text{dml}}$ and $\mathcal{L}_{\text{dec}}$. Next, during online adaptation, we introduce **COSTA-ts**, which employs Thompson sampling to generate contexts, whereas **COSTA-wo-iid** utilizes the average embedding of all rollout trajectories without the SIDS filtering mechanism. **COSTA-wo-trunc** employs the original form of SIDS without truncating unsafe trajectories. These variants can verify the necessity of our proposed safe in-distribution online adaptation process. Furthermore, we combine the two kinds of variants and derive **COSTA-ts-wo-cc**, in which only $\mathcal{L}_{\text{dml}}$ and $\mathcal{L}_{\text{dec}}$ are employed for training the context encoder, and contexts for testing are generated through Thompson sampling.

The experimental results in two environments are presented in Table 2. To begin with, we aim to understand the impact of $\mathcal{L}_{\text{dml}}$ and $\mathcal{L}_{\text{cc}}$ in context encoder learning on the results of offline training. From the results, we can find that both of these losses are indispensable during the learning process of our context encoder. The absence of either one could potentially affect the robustness of the context encoder in recognizing tasks. Furthermore, it can be observed that these two objectives have varying effects on the algorithm in different environments. In AntDir, the absence of $\mathcal{L}_{\text{dml}}$ leads to errors in task recognition during training, resulting in more severe safety violations. On the other hand, the absence of $\mathcal{L}_{\text{cc}}$ results in unsafe actions in CheetahWalk.

Moving forward, upon observing the results of online adaptation, it becomes evident that the absence of $\mathcal{L}_{\text{dml}}$ or $\mathcal{L}_{\text{cc}}$ can lead to a significant decline in performance and violations in safety constraints, even though the offline training results surpass those of COSTA, e.g., COSTA-wo-cc in CheetahWalk. To explain the phenomenon, we believe that, in the absence of $\mathcal{L}_{\text{cc}}$, the distribution shift caused by offline RL and trajectory truncation may have more pronounced and adverse effects on context encoder, thus resulting in a higher probability of incorrect task identification. This further demonstrates the effectiveness of the combination of $\mathcal{L}_{\text{dml}}$ and $\mathcal{L}_{\text{cc}}$.

In the study of components within online adaptation, it turns out that COSTA often struggles with correctly identifying all tasks when utilizing traditional Thompson sampling-based methods for task recognition, which is reflected in the results of COSTA-ts. Consequently, there tends to be a certain degree of safety constraint violations. Moreover, in the absence of in-distribution trajectory ranking based on SIDS, a considerable decline in performance is observed, as COSTA-wo-iid implies. The phenomenon validates the necessity of our robust online task adaptation in solving offline safe meta RL problems. More comprehensive results about ablation studies in other environments can be seen in Appendix E.

Finally, to better assess the ability of the learned context encoder to differentiate tasks based on cost functions rather than state-action



(a) COSTA (b) FOCAL

(c) CORRO (d) PEARL

Figure 4: The 2D projection of the learned task representation space in AntDir using t-SNE. Here, all contexts from different tasks are collected using a same behavioral policy.

distributions for a robust and safe policy, we visualize the task representations by projecting context embeddings into a 2D space using t-SNE [37]. Specifically, we utilize a same behavioral policy for each task to collect 100 trajectories in AntDir. Then, we treat each trajectory as a context and produce its low-dimensional embedding via the context encoder. As shown in Figure 4, in comparison to other baselines, only COSTA is capable of correctly distinguishing tasks to some extent when the only difference lies in the cost. Although there is still some overlap among the learned task representations, unlike the other baselines that learn representations for task identification like FOCAL, CORRO, and PERAL, which thoroughly mix tasks together and cannot distinguish them at all. This clearly demonstrates a certain degree of success in decoupling the context encoder from the behavioral policies in COSTA's context encoder learning, indicating that the design of COSTA could indeed obtain a highly differentiated task identification. More experimental results, such as how $\text{SIDS}_{\text{trunc}}$ influences the cumulative cost during online adaptation and how parameter α_{cc} influences COSTA, could be found in Appendix E.

6 FINAL REMARKS

This paper introduces a novel offline safe meta RL algorithm called COSTA to enable safer and more widely applicable RL in real-world applications. To learn a generalized safe policy from offline datasets, COSTA learns robust task representations by incorporating distance metric learning and cost contrastive loss into the training of the context encoder. During the online adaptation process, we generate truncated in-distribution trajectories with safety guarantee and introduce two measurements, SIDS and $\text{SIDS}_{\text{trunc}}$, to filter out trajectories. Extensive experimental results demonstrate that our method outperforms other approaches by providing better safety guarantees, higher rewards, and fewer safety constraint violations during the exploration process. It also exhibits a degree of generalization and transferability. However, there is also some future research that could be conducted. Firstly, we discuss the offline safe meta RL problem under the small-scale scenarios, the extension to complex environments with more tasks and multiple constraints is not thoroughly studied. Secondly, COSTA requires extra information preserved during offline training to perform robust online adaptation, more accurate and fast recognition is subject to further research. Finally, investigating the offline meta-safe problem in the multi-agent reinforcement learning setting is a promising and valuable direction for future work.

ACKNOWLEDGMENTS

This work is supported by National Key Research and Development Program of China (2020AAA0107200) and the National Science Foundation of China (61921006, 62022039, 62276124). We would like to express our gratitude to the anonymous reviewers for their kind reviews and constructive feedback.

REFERENCES

[1] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. 2017. Constrained policy optimization. In *Proceedings of the International Conference on Machine Learning*. 22–31.
[2] Eitan Altman. 2021. *Constrained Markov Decision Processes*. Routledge.
[3] Yarden As, Ilnura Usmanova, Sebastian Curi, and Andreas Krause. 2022. Constrained policy optimization via bayesian world models. In *International Conference on Learning Representations*.
[4] Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shimon Whiteson. 2023. A survey of meta-reinforcement learning. *arXiv preprint arXiv:2301.08028* (2023).
[5] Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. 2017. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research* 18, 1 (2017), 6070–6120.
[6] Ron Dorfman, Idan Shenfeld, and Aviv Tamar. 2020. Offline meta learning of exploration. *arXiv preprint arXiv:2008.02598* (2020).
[7] Ron Dorfman, Idan Shenfeld, and Aviv Tamar. 2021. Offline Meta Reinforcement Learning–Identifiability Challenges and Effective Data Collection Strategies. In *Advances in neural information processing systems*. 4607–4618.
[8] Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. 2016. Rl ^{2}: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779* (2016).
[9] Javier Garcıa and Fernando Fernández. 2015. A comprehensive survey on safe reinforcement learning. *The Journal of Machine Learning Research* 16, 1 (2015), 1437–1480.
[10] Shangding Gu, Jakub Grudzien Kuba, Yuanpei Chen, Yali Du, Long Yang, Alois Knoll, and Yaodong Yang. 2023. Safe multi-agent reinforcement learning for multi-robot control. *Artificial Intelligence* 319 (2023), 103905.
[11] Shangding Gu, Long Yang, Yali Du, Guang Chen, Florian Walter, Jun Wang, Yaodong Yang, and Alois Knoll. 2022. A review of safe reinforcement learning: Methods, theory and applications. *arXiv preprint arXiv:2205.10330* (2022).
[12] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. 2018. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905* (2018).
[13] Jiaming Ji, Borong Zhang, Xuehai Pan, Jiayi Zhou, Juntao Dai, and Yaodong Yang. 2023. Safety-Gymnasium. https://github.com/PKU-Alignment/safety-gymnasium. *GitHub repository* (2023).
[14] Jianhao Wang, Jin Zhang 016, Haozhe Jiang, Junyu Zhang, Liwei Wang, and Chongjie Zhang. 2023. Offline Meta Reinforcement Learning with In-Distribution Online Adaptation. In *Proceedings of the International Conference on Machine Learning*. 36626–36669.
[15] Jens Kober, J Andrew Bagnell, and Jan Peters. 2013. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research* 32, 11 (2013), 1238–1274.
[16] Hanna Krasowski, Xiao Wang, and Matthias Althoff. 2020. Safe reinforcement learning for autonomous lane changing using set-based prediction. In *IEEE International Conference on Intelligent Transportation Systems*. 1–7.
[17] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. 2020. Conservative q-learning for offline reinforcement learning. In *Advances in neural information processing systems*. 1179–1191.

[18] Hoang Le, Cameron Voloshin, and Yisong Yue. 2019. Batch policy learning under constraints. In *Proceedings of the International Conference on Machine Learning*. 3703–3712.

[19] Jongmin Lee, Cosmin Paduraru, Daniel J Mankowitz, Nicolas Heess, Doina Precup, Kee-Eung Kim, and Arthur Guez. 2022. Coptidice: Offline constrained reinforcement learning via stationary distribution correction estimation. In *International Conference on Learning Representations*.

[20] Jiachen Li, Quan Vuong, Shuang Liu, Minghua Liu, Kamil Ciosek, Henrik Christensen, and Hao Su. 2020. Multi-task batch reinforcement learning with metric learning. In *Advances in neural information processing systems*. 6197–6210.

[21] Lanqing Li, Rui Yang, and Dijun Luo. 2021. FOCAL: Efficient Fully-Offline Meta-Reinforcement Learning via Distance Metric Learning and Behavior Regularization. In *International Conference on Learning Representations*.

[22] Zuxin Liu, Zijian Guo, Haohong Lin, Yihang Yao, Jiacheng Zhu, Zhepeng Cen, Hanjiang Hu, Wenhao Yu, Tingnan Zhang, Jie Tan, et al. 2023. Datasets and Benchmarks for Offline Safe Reinforcement Learning. *arXiv preprint arXiv:2306.09303* (2023).

[23] Zuxin Liu, Zijian Guo, Yihang Yao, Zhepeng Cen, Wenhao Yu, Tingnan Zhang, and Ding Zhao. 2023. Constrained Decision Transformer for Offline Safe Reinforcement Learning. In *Proceedings of the International Conference on Machine Learning*. 21611–21630.

[24] Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101* (2017).

[25] Eric Mitchell, Rafael Rafailov, Xue Bin Peng, Sergey Levine, and Chelsea Finn. 2021. Offline meta-reinforcement learning with advantage weighting. In *Proceedings of the International Conference on Machine Learning*. 7780–7791.

[26] Jan Peters and Stefan Schaal. 2008. Reinforcement learning of motor skills with policy gradients. *Neural networks* 21, 4 (2008), 682–697.

[27] Nicholas Polosky, Bruno C Da Silva, Madalina Fiterau, and Jithin Jagannath. 2022. Constrained offline policy optimization. In *Proceedings of the International Conference on Machine Learning*. 17801–17810.

[28] Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. 2019. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *Proceedings of the International Conference on Machine Learning*. 5331–5340.

[29] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *Proceedings of the International Conference on Machine Learning*. 1889–1897.

[30] Wenjie Shang, Yang Yu, Qingyang Li, Zhiwei Qin, Yiping Meng, and Jieping Ye. 2019. Environment reconstruction with hidden confounders for reinforcement learning based recommendation. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 566–576.

[31] Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*. 1849–1857.

[32] Adam Stooke, Joshua Achiam, and Pieter Abbeel. 2020. Responsive safety in reinforcement learning by pid lagrangian methods. In *Proceedings of the International Conference on Machine Learning*. 9133–9143.

[33] Malcolm Strens. 2000. A Bayesian framework for reinforcement learning. In *Proceedings of the International Conference on Machine Learning*. 943–950.

[34] Yihao Sun. 2023. OfflineRL-Kit: An Elegant PyTorch Offline Reinforcement Learning Library. https://github.com/yihaosun1124/OfflineRL-Kit.

[35] Chen Tessler, Daniel J Mankowitz, and Shie Mannor. 2018. Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074* (2018).

[36] Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. Mujoco: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 5026–5033.

[37] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *The Journal of Machine Learning Research* 9, 11 (2008).

[38] Hao-nan Wang, Ning Liu, Yi-yun Zhang, Da-wei Feng, Feng Huang, Dong-sheng Li, and Yi-ming Zhang. 2020. Deep reinforcement learning: a survey. *Frontiers of Information Technology & Electronic Engineering* 21, 12 (2020), 1726–1744.

[39] Xiao Wang. 2022. Ensuring safety of learning-based motion planners using control barrier functions. *IEEE Robotics and Automation Letters* 7, 2 (2022), 4773–4780.

[40] Xu Wang, Sen Wang, Xingxing Liang, Dawei Zhao, Jincai Huang, Xin Xu, Bin Dai, and Qiguang Miao. 2022. Deep reinforcement learning: a survey. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[41] Haoran Xu, Xianyuan Zhan, and Xiangyu Zhu. 2022. Constraints penalized q-learning for safe offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*. 8753–8760.

[42] Tsung-Yen Yang, Justinian Rosca, Karthik Narasimhan, and Peter J Ramadge. 2020. Projection-based constrained policy optimization. In *Advances in neural information processing systems*.

[43] Yang Yu. 2018. Towards Sample Efficient Reinforcement Learning. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 5739–5743.

[44] Haoqi Yuan and Zongqing Lu. 2022. Robust task representations for offline meta-reinforcement learning via contrastive learning. In *Proceedings of the International Conference on Machine Learning*. 25747–25759.

[45] Qin Zhang, Linrui Zhang, Haoran Xu, Li Shen, Bowen Wang, Yongzhe Chang, Xueqian Wang, Bo Yuan, and Dacheng Tao. 2023. Saformer: A conditional sequence modeling approach to offline safe reinforcement learning. In *Advances in neural information processing systems*.

[46] Xiangyu Zhao, Liang Zhang, Zhuoye Ding, Long Xia, Jiliang Tang, and Dawei Yin. 2018. Recommendations with Negative Feedback via Pairwise Deep Reinforcement Learning. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1040–1048.

Appendix of submission 533

A ALGORITHMS

As described in the main manuscript, the workflow of our algorithm primarily includes two parts: offline training and online adaptation. Detailed pseudo codes are provided in Algorithm 1 and Algorithm 2, respectively.

During the offline training process, our algorithm is primarily divided into three parts. Lines 2-8 of Algorithm 1 represent the first part: the training of task-specific cost models and conditional variational autoencoders (CVAE). The cost models are trained using traditional supervised learning, where (s, a, s') serves as the input, and its corresponding cost is the output. The CVAEs are trained using standard KL loss and reconstruction loss, with a as the reconstruction target and s as the conditioning variable. Lines 9-25 illustrate the second part: the learning of COSTA's context encoder. In practice, as shown in line 15, $\mathcal{L}_{\text{cc}}$ is calculated only when $\mathcal{T}_i$ and $\mathcal{T}_j$ are different to avoid unnecessary updates. Lines 26-37 demonstrate the third part of the algorithm: the policy update. The estimation of the OOD action distribution is performed using the previously trained CVAEs, as indicated in lines 31-32.

Algorithm 2 illustrates the process of online adaptation in our algorithm. In lines 1-4, we utilize previously stored context embeddings to perform rollouts in an unknown environment and truncate each trajectory based on its cumulative cost. Lines 5-7 of the algorithm demonstrate how COSTA selects the optimal context based on $\text{SIDS}_{\text{trunc}}$ and conducts policy evaluation accordingly.

B DEFINITIONS

In this section, we first provide a detailed definition of the task-augmented CMDP mentioned in the main manuscript, as shown in Definition B.1. Following that, we present the definition of the Bellman optimality operator based on the task-augmented CMDP, as shown in Definition B.2.

*Definition B.1 (**TA-CMDP**).* A task-augmented Constrained Markov Decision Process (TA-CMDP) can be modeled as $\mathcal{M} = (S, Z, A, r, c, P, \gamma, b)$:

- S: state space
- Z: contextual latent space
- A: action space
- r: reward function $r(s, z, a) = r_z(s, a)$
- c: cost function $c(s, z, a) = c_z(s, a)$
- P: transition probability function $P(s', z'|s, z, a) = P_z(s'|s, a)$ if there is no intra-task transition
- γ: discount factor
- b: safety constraint limit

Definition B.2. The Bellman optimality operator $\mathcal{B}_z$ on TA-CMDP is defined as

$$(\mathcal{B}_z\hat{Q}_r)(s, z, a) := r(s, z, a) + \gamma \mathbb{E}_{P(s',z'|s,z,a)}[\max_{a'} \hat{Q}_r(s', z', a')], \tag{14}$$

$$(\mathcal{B}_z\hat{Q}_c)(s, z, a) := r(s, z, a) + \gamma \mathbb{E}_{P(s',z'|s,z,a)}[\min_{a'} \hat{Q}_c(s', z', a')]. \tag{15}$$

We here also present the detailed expression of $\text{SIDS}_{\text{trunc}}$:

$$\text{SIDS}_{\text{trunc}}(\tau) = \begin{cases} R(\tau_{1:t_{\text{tc}}}) & C(\tau_{1:t_{\text{tc}}}) \le b + \epsilon_1, R(\tau_{1:t_{\text{tc}}}) \ge \delta \\ -C(\tau_{1:t_{\text{tc}}}) & C(\tau_{1:t_{\text{tc}}}) > b + \epsilon_1, R(\tau_{1:t_{\text{tc}}}) \ge \delta \\ -K & R(\tau_{1:t_{\text{tc}}}) < \delta \end{cases}. \tag{16}$$

If $\epsilon_2 = \epsilon_1$, then $\text{SIDS}_{\text{trunc}}$ can be simplified to:

$$\text{SIDS}_{\text{trunc}}(\tau) = R(\tau_{1:t_{\text{tc}}}). \tag{17}$$

C DETAILED DESCRIPTION OF THE ENVIRONMENTS AND BASELINES

We introduce the details of the relevant environments and baselines in this part.

C.1 Environments

The environments in this article are all based on the Ant or Half-Cheetah robots. The reward function for an Ant robot in all environments consists of four components: reward = healthy reward - ctrl cost - contact cost + forward reward. In different environments, the first three components of the reward remain the same, with the healthy reward being 1 as long as the running is not terminated. Ctrl cost refers to the cost incurred by controlling the agent and measures the magnitude of the agent's actions, while contact cost quantifies the external contact forces experienced by the agent. For more details, please refer to the MuJoCo documentation. On the other hand, a Half-Cheetah robot does not have a termination condition, so its reward function consists of two components: reward = forward reward - ctrl cost, where ctrl cost is defined similarly to that in the Ant case. The forward reward for both robots is environment-specific, and we will provide a more detailed explanation of this below. Moreover, the maximum episode length of all environments is set to 300.

Algorithm 1 COSTA Offline Training

Input: Pre-collected buffer $\mathcal{D}_i = \{(s_j, a_j, r_j, c_j, s'_j)\}_{j=1}^{|\mathcal{D}_i|}$ of training tasks $\{\mathcal{T}_i\}_{i=1}^{T}$ from $p(\mathcal{T})$, KL threshold d, safety constraint limit b, α_{cc}, α_{dec}, λ, p, ϵ, β_{KL}, α_{cpq}, discount factor γ

Initialize: randomly initialize inference network q_ϕ, cost models $\{f_i\}_{i=1}^{T}$, CVAEs $\{q_{\omega_1^i}, p_{\omega_2^i}\}_{i=1}^{T}$, meta-policy π_θ , reward critic Q_{φ_r}, cost critic Q_{φ_c}

while not done **do**
  for step in training steps **do**
    for each $\mathcal{T}_i$ **do**
      Sample a mini-batch of state-action-cost transitions $(s, a, s', c) \sim \mathcal{D}_i$
      Update f_i using supervised learning
      Update $q_{\omega_1^i}$ and $p_{\omega_2^i}$ using standard CVAE loss
    end for
  end for
  for step in context training steps **do**
    for each $\mathcal{T}_i$ **do**
      Sample a mini-batch $e_i \sim \mathcal{D}_i$ for context encoder training
      for each $\mathcal{T}_j$ **do**
        Sample a mini-batch e_j from $\mathcal{D}_j$
        Compute $\mathcal{L}_{\text{dml}}^{ij}$ with Equation (4)
        if $\mathcal{T}_i \neq \mathcal{T}_j$ **then**
          Create e_{pos} with e_j and f_i
          Create e_{neg} with e_i and f_j
          Compute $\mathcal{L}_{\text{cc}}^{ij}$ with Equation (5)
        end if
      end for
      Compute $\mathcal{L}_{\text{dec}}^{i}$ with Equation (6)
    end for
    $\mathcal{L}_{\text{dml}} = \sum_{ij} \mathcal{L}_{\text{dml}}^{ij}, \quad \mathcal{L}_{\text{cc}} = \sum_{ij} \mathcal{L}_{\text{cc}}^{ij}, \quad \mathcal{L}_{\text{dec}} = \sum_{i} \mathcal{L}_{\text{dec}}^{i}$
    Update q_ϕ to minimize Equation (7)
  end for
  for step in RL training steps **do**
    for each $\mathcal{T}_i$ **do**
      Sample mini-batches e_i and $b_i \sim \mathcal{D}_i$ for policy training
      Compute context embedding z_i with e_i and q_ϕ
      for each state s in b_i **do**
        Sample n actions $\{a_v \sim \pi_\theta(\cdot|s)\}_{v=1}^{n}$, get latent mean and std $\{\mu_v, \sigma_v = q_{\omega_1^i}(s, a_v)\}_{v=1}^{n}$ and extract $m (m \geq 0)$ actions $\{a_u | D_{\text{KL}}(\mathcal{N}(\mu_u, \sigma_u) || \mathcal{N}(0, 1)) \geq d\}_{u=1}^{m}$ from them
        Let $Q_c(s, v(s)) = \frac{1}{m} \sum_u Q_c(s, a_u)$ if $m > 0$ else 0
      end for
      Update cost critic Q_{φ_c} by Equation (10) and reward critic Q_{φ_r} by Equation (12)
      Update policy π_θ by Equation (13)
    end for
  end for
end while

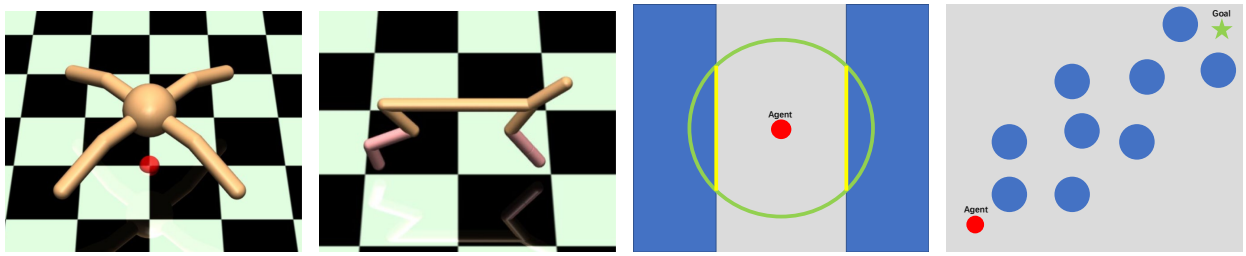
- **AntDir**: In this environment, the Ant robot receives a reward when it moves in the positive x-axis direction, consistent with MuJoCo's reward. Specifically, the forward reward is calculated as the change in the agent's mass center along the positive x-axis divided by the time between actions. To provide a clearer explanation of the cost function components, let's denote the x-coordinate of the agent's mass center as x and the y-coordinate as y. For task_0, if $-3 \leq y \leq 3$, the cost is 0; otherwise, the cost is 1. For task_1, the cost function

Algorithm 2 COSTA Online Adaptation

Input: A test task $\mathcal{T}_{test} \sim p(\mathcal{T})$, a set of previously computed context embeddings of training tasks $\{z_s\}_{s=1}^{T\times m}$, meta-trained policy π_θ, context encoder q_ϕ, safety constraint limit b, $\text{SIDS}_{\text{trunc}}$ parameters $\epsilon_1, \epsilon_2, \delta, K$, final trajectory num l

Initialize: Final context $e^* = \{\}$, rollout trajectory set $\tau = \{\}$

1: **for** z_s in context embedding set **do**
2: Using π_θ to rollout a trajectory τ_s based on z_s until terminal or truncated for safety
3: Adding τ_s to τ
4: **end for**
5: Select l trajectories in τ with highest $\text{SIDS}_{\text{trunc}}$ values as the final context e^*
6: Compute context embedding $z^* = q_\phi(e^*)$
7: Rollout policy $\pi_\theta(\cdot|\cdot, z^*)$ for evaluation



(a) Ant (b) Half-Cheetah (c) Circle (d) Goal

Figure 5: Visualization of robots and 2 environments.

is denoted as

$$\text{cost}_1^{\text{AntDir}}(x,y) = \begin{cases} 1 - \mathbb{1}(x-4 \le y \le x+4) & 0 \le x \le 10 \\ 1 - \mathbb{1}(16-x \le y \le 24-x) & 10 < x \le 30 \\ 1 - \mathbb{1}(x-36 \le y \le x-44) & 30 < x \le 40 \\ 1 - \mathbb{1}(-3 \le y \le 3) & \text{otherwise} \end{cases}, \tag{18}$$

while the cost function of task_2 is just the symmetry of task_1's cost function with respect to x-axis.

- **CheetahVel**: The Half-Cheetah robot receives its forward reward based on its movement speed along the x-axis, with higher rewards for faster speeds. In each task, a number is randomly sampled from a uniform distribution ranging from 1.5 to 3, and this number serves as the speed threshold. If the speed of the Cheetah robot exceeds this threshold, the cost is set to 1; otherwise, the cost is 0. The design of this environment is intended to investigate scenarios where cost functions between tasks are solely related to actions.
- **AntWalk**: In AntWalk, the forward reward is determined by the change in the Manhattan distance between the Ant robot's center of mass and the origin, divided by the time between actions. The task_0's cost function is denoted as:

$$\text{cost}_0^{\text{AntWalk}}(x,y) = \begin{cases} 1 & x \le -1 \\ 1 - \mathbb{1}(0.87x - 3 \le y \le 0.87x + 3) & otherwise \end{cases}, \tag{19}$$

while the cost function of task_1 is just the symmetry of task_0's cost function with respect to x-axis. The task_2's cost function is:

$$\text{cost}_2^{\text{AntWalk}}(x,y) = \begin{cases} 1 & x \ge 1 \\ 1 - \mathbb{1}(-4 \le y \le 4) & otherwise \end{cases}. \tag{20}$$

This environment is designed to study scenarios where there is minimal overlap in the state-action distributions between different tasks' behavioral policies.

- **CheetahWalk**: In CheetahWalk, the forward reward is the same as in AntWalk. However, due to the unique characteristics of the Half-Cheetah robot, the cost function is only related to the x-coordinate. For task_0, the cost is 1 if $x \le -1$, otherwise, it's 0, while for task_1, the cost is 1 if $x \ge 1$.
- **AntCircle**: In AntCircle, the agent is rewarded when it moves along a circular path centered at the origin. Specifically, assuming the radius of the circle is denoted as r_{circle}, the distance of the agent from the origin is r_{agent}, and the agent's velocity components along the x and y axes are u and v. Then

$$\text{forward\_reward} = \frac{1}{1 + |r_{\text{agent}} - r_{\text{circle}}|} * \frac{-uy + vx}{r_{\text{agent}}}, \tag{21}$$

where $r_{\text{circle}} = 10$ in this paper. However, only part of the circle is safe. For task_0, the cost function is

$$\text{cost}_0^{\text{AntCircle}}(x, y) = 1 - \mathbb{1}(-6 \leq x \leq 6). \tag{22}$$

Just as Figure 5(c) shows, the circular path is considered safe only when it lies between two yellow parallel lines, while task_1 and task_2 involve rotating the safe area of task_0 around the origin by 90 degrees and 45 degrees respectively.

- **AntGoal**: In this environment, all tasks share a common target point located at (25, 25). The forward reward for the agent is calculated as the negative change in the distance between the agent's center of mass and the target position, divided by the time between actions. On the path from the agent to the target point, there are non-safe regions. For both task_0 and task_1, the non-safe region is a large contiguous area, and they are symmetric with respect to the line y = x:

$$\text{cost}_0^{\text{AntGoal}}(x, y) = \mathbb{1}(10 \leq x \leq 20, 2 \leq y \leq 20). \tag{23}$$

In contrast, task_2's non-safe region consists of multiple smaller areas. Specifically, it is composed of circles with radius of 1 and centered at points from the following set: $\{(5, 5), (5, 10), (10, 5), (12.5, 12.5), (10, 17.5), (17.5, 10), (20, 20), (22.5, 25), (25, 22.5)\}$, similar to what is shown in Figure 5(d).

C.2 Baselines

We provide a more detailed introduction to the comparison baselines of the experiment in this section.

FOCAL [21] is designed for fully offline meta-reinforcement learning problems. It employs a deterministic context encoder and utilizes distance metric learning during training to ensure that the context encoder can robustly identify tasks. Due to its fully offline nature, it doesn't involve an online adaptation process.

IDAQ [14] is an online adaptation method based on FOCAL. It uses uncertainty quantification to assess the in-distribution degree of trajectories and proposes three specific uncertainty quantification methods: Prediction Error, Prediction Value, and Return-based. In the end, it selects only the trajectories with in-distribution degrees exceeding a certain threshold as the final context.

CORRO [44] designs a bi-level encoder structure containing a transition encoder and an aggregator. The transition encoder is learned through contrastive learning based on transition pairs, while the aggregator is learned through the loss of value functions and policies. It introduces two methods: Reward Randomization and Generative Modeling to generate negative samples for contrastive learning. Reward Randomization generates negative samples by adding random noise to the reward information of tasks. On the other hand, Generative Modeling trains a CVAE using (s, a) as conditions and (r, s') as reconstruction targets to obtain negative samples. Due to the poor performance of the two negative pairs generation methods CORRO employs when dealing with costs, we also use the method of constructing negative pairs with transition tuples from different tasks.

PEARL [28] learns a probabilistic fully-connected context encoder through the value function loss during policy training and the KL divergence loss with a given prior distribution for off-policy meta reinforcement learning. It first introduces the Thompson sampling method for meta-testing in context-based meta reinforcement learning.

D EXPERIMENTAL DETAILS

D.1 Offline Data Collection

The offline datasets used in this paper are all collected using CPO [1]. Due to the increased difficulty of learning with safety considerations compared to scenarios with only rewards, a larger number of training steps are required for data collection in this study. Furthermore, all datasets consist of mixed data, which includes both unsafe and safe trajectories. We quantify this using the Cost Rate metric, defined as the number of non-safe transitions in the dataset divided by the total number of transitions. For example, if the max episode length is 300 and the safety constraint limit is 25, an expert dataset should have a Cost Rate less than or equal to 0.083. The generation of mixed datasets involves periodically saving checkpoints during the training process of the online algorithm, starting from a certain step, and collecting trajectories based on these checkpoints. Detailed data is presented in Table 3. The reason for the additional one-dimensional observation dimension in CheetahWalk compared to CheetahVel is that the reward calculation in CheetahWalk is related to the robot's x-coordinate. Therefore, the x-coordinate information is included as an extra component in the observation.

D.2 Practical Implementation and Hyperparameters

This paper implements COSTA based on CQL [17] within the OfflineRL-Kit [34] framework. Default parameters in the framework are used for those hyperparameters not mentioned in the following sections.

In the stages of training for the cost models and CVAEs, all environment parameters are kept consistent. To enhance the reliability of the cost models, we employ a model ensemble approach, training a total of "num ensemble" models and selecting "num elites" models for integration based on their accuracy on the validation set. Both the cost models and CVAEs utilize MLP networks and employ the Adam optimizer for parameter optimization. Detailed hyperparameters are provided in Table 4.

In the subsequent training stages, there are cases where hyperparameters vary between different environments. Therefore, we categorize hyperparameters into two groups: environment-specific and environment-independent. Environment-independent hyperparameters remain constant across our experiments. During the learning of the context encoder, the encoder is implemented as a neural network with an added

Table 3: Details of fixed datasets used in all our experiments. The three numbers in the "Checkpoints" column stand for starting epoch: ending epoch: checkpoint spacing. The "Trajectory num" column means the number of trajectories collected by each checkpoint. The "Cost Rate" column is the mean cost rate of all tasks.

Meta Env	Task num	Checkpoints	Trajectory num	Trajectory Steps	Cost Rate	Safety Constraint Limit	Obs Dim	Act Dim
AntDir	3	6e5:3e7:6e4	10	300	0.106	25	29	8
CheetahVel	3	6e5:3e7:6e4	10	300	0.096	25	20	6
AntWalk	3	6e5:3e7:6e4	10	300	0.098	25	29	8
CheetahWalk	2	0:6e5:6e3	1	300	0.04	10	21	6
AntCircle	3	6e5:1.8e7:6e4	10	300	0.116	25	29	8
AntGoal	3	6e5:3e7:6e4	10	300	0.09	25	29	8

Table 4: Hyperparameters for CVAE training and cost model training.

	Hyperparameters	Value
CVAE	latent dim	16
	β_{KL}	0.01
	lr	1e-3
	encoder hidden layers	[128,128]
	decoder hidden layers	[128,128]
	batch size	512
cost model	num ensemble	5
	num elites	5
	validation ratio	0.2
	lr	5e-4
	hidden layers	[128,64,32,16]
	batch size	512

Attention mechanism, while the decoder is a regular MLP network. Both are trained using the AdamW [24] optimizer, and we introduce a stepwise learning rate decay to facilitate better learning. In the offline policy training process, the actor, reward critics, and cost critics are all composed of MLP networks and optimized using the Adam optimizer. Additionally, we adopt the double Q-network mechanism from SAC [12], which helps alleviate the issue of overestimating rewards by selecting the Q-network with the lower estimated reward value among the two. However, this mechanism is not applied to the cost critic. The environment-independent hyperparameters are listed in Table 5, while the environment-specific hyperparameters are detailed in Table 6.

E ADDITIONAL EXPERIMENTAL RESULTS

E.1 Ablation Studies

To enhance the generality of the ablation study results, we conduct experiments in two additional environments, and the outcomes are presented in Table 7. The conclusion drawn from this experiment is consistent with the previous ablation study, further substantiating the effectiveness of each main component in COSTA.

To better understand the impact of $\mathcal{L}_{\text{cc}}$ on the context encoder's ability to correctly identify tasks in COSTA, we conduct a comparison between COSTA and COSTA-wo-cc using traditional Thompson sampling-based online adaptation. The results are depicted in Figure 6(a). The comparative results reveal that, even though COSTA-ts may not achieve perfect task recognition in three of the environments, its performance exhibits significant improvements in both reward and cost aspects compared to COSTA-ts-wo-cc. Additionally, in environments where both methods meet safety constraints, COSTA-ts demonstrates a notable advantage in terms of reward.

Finally, in regard to the impact of trajectory truncation using SIDS_trunc on the accumulation of costs during the online adaptation process, we conduct a comparison between COSTA, FOCAL, and COSTA-wo-trunc, as illustrated in Figure 6(b). It can be observed that while both COSTA and COSTA-wo-trunc correctly identify tasks, COSTA-wo-trunc incurs nearly five times the total number of costs during the recognition process as compared to COSTA. On the other hand, FOCAL, despite incurring approximately four times the total number of costs as COSTA, still fails to correctly identify tasks.

E.2 Sensitivity Studies

The primary components of COSTA's loss function consist of $\mathcal{L}_{\text{dml}}$ and $\mathcal{L}_{\text{cc}}$. Therefore, we conduct sensitivity experiments in AntCircle by manipulating the parameter α_{cc}, which combines these two components. The results are depicted in the graph. Observing the outcomes of offline training, it is evident that gradually increasing α_{cc} from 0 to 1 consistently enhances COSTA's rewards while maintaining safety.

Table 5: Environment-independent hyperparameters for context encoder training, meta-policy training and online adaptation.

	Hyperparameters	Value
context encoder	λ_1	1
	λ_2	1
	p	2
	ϵ	0.1
	α_{dec}	1
	encoder latent dim	16
	encoder hidden layers	[128,64,32,16]
	cost decoder hidden layers	[128,64,32,16]
	training epochs	100
	training steps per epoch	100
	context batch size n	300
	meta batch size	8
	lr	1e-4
	lr decay rate	0.9
	lr decay step	2000
meta-policy	α_{cpq}	1
	policy hidden layers	[256,256,256]
	training steps per epoch	1000
	batch size	1024
	actor lr	1e-4
	critic lr	3e-4
	discount factor γ	0.99
	random action num	10
online adaptation	K	300
	ϵ_1	5
	l	1
	m	5

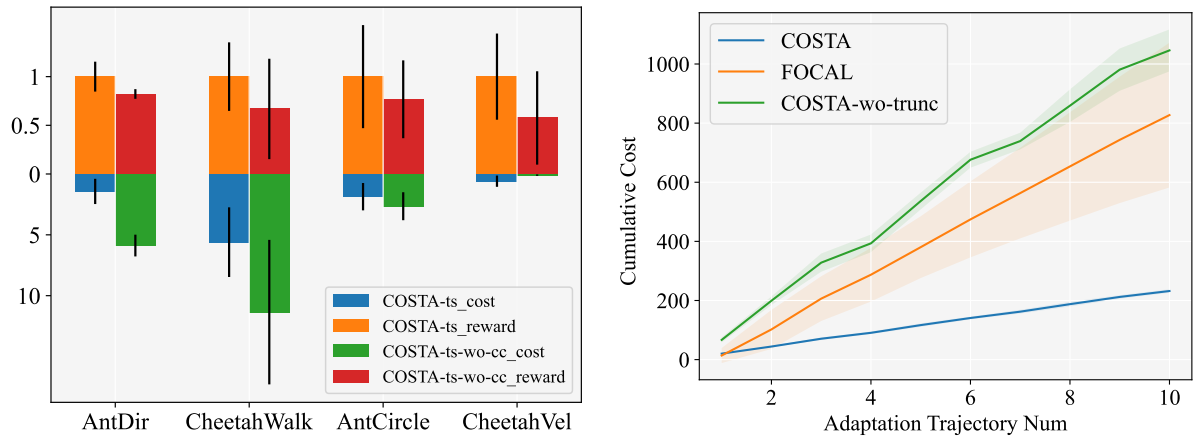
Table 6: Environment-specific hyperparameters for context encoder training, meta-policy training and online adaptation.

Hyperparameters	AntDir	CheetahVel	AntWalk	CheetahWalk	AntCircle	AntGoal
α_{cc}	0.1	0.1	0.01	0.1	0.1	0.01
KL threshold d	26	14	12	12	18	14
safety constraint limit b	25	25	25	10	25	25
ϵ_2	5	5	5	5	10	10
δ	600	-50	300	-100	100	600

Table 7: Additional ablation study.

Task		COSTA		COSTA-ts		COSTA-wo-iid		COSTA-wo-dml		COSTA-wo-cc	
		reward↑	cost↓	reward↑	cost↓	reward↑	cost↓	reward↑	cost↓	reward↑	cost↓
AntCircle	Training	1182±219.65	0.55±0.36	/	/	/	/	1103±334.99	0.81±0.65	967±353.69	0.73±0.57
	Adaptation	1147±118.64	0.58±0.35	812±429.85	1.87±1.12	420±170.04	0.56±0.97	1067±108.38	1.61±0.61	1102±412.73	1.07±0.82
CheetahVel	Training	354±31.70	0.89±0.12	/	/	/	/	402±52.07	1.53±0.27	354±106.21	1.10±0.28
	Adaptation	348±40.69	0.84±0.21	238±105.33	0.59±0.46	277±35.06	0.39±0.06	220±188.19	1.11±0.89	280±151.40	0.52±0.32

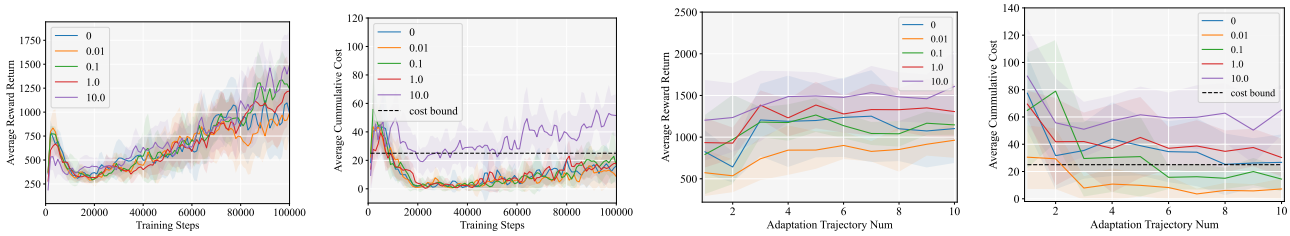
However, further increasing it to 10 leads to errors in task recognition during offline scenarios, resulting in a severe breach of safety constraints. In the context of online adaptation, when α_{cc} is set to 0, even though offline training ensures safety, it violates safety constraints during online adaptation. To a certain extent, increasing α_{cc} to 0.01 or 0.1 ensures policy safety. However, pushing α_{cc} further to 1 or higher again results in safety constraint violations. This underscores the significance of carefully selecting this parameter, as it is essential to strike a balance between $\mathcal{L}_{\text{dml}}$ and $\mathcal{L}_{\text{cc}}$.



(a) Online Adaptation based on Thompson Sampling

(b) Cumulative Cost

Figure 6: (a) Ablation study on the performance of COSTA with or without cost contrastive loss during online adaptation based on Thompson sampling. The rewards are normalized by COSTA-ts's reward. (b) Ablation study on cumulative cost during the whole process of online adaptation in AntDir.



(a) Offline Training Reward

(b) Offline Training Cost

(c) Online Adaptation Reward

(d) Online Adaptation Cost

Figure 7: Sensitivity of parameter α_{cc} in AntCircle.

F MORE DETAILS ABOUT RELATED WORK

Safe RL. Safety has been one of the major roadblocks in the way of deploying RL policies to the real world [9, 11]. To solve the problem, safe RL mostly models the environment as a CMDP [2] and learns a policy that maximizes cumulative reward while satisfying safety constraints. Lagrangian methods [5, 35] are classic approaches to solving constrained optimization problems. They introduce the Lagrange multiplier as a dual variable and apply gradient-based algorithms iteratively to derive an optimal and constraint-satisfying policy. However, they often take illegal actions and violate the constraints during training. PID-Lagrangian methods [32] fix this deficiency by interpreting the learning process as a dynamic system and updating the Lagrange multiplier via derivatives of the constraint functions. Constrained Policy Optimization (CPO) [1] combines local policy search [26] and trust region methods [29] to guarantee the monotonic improvement in reward while ensuring the safety constraints. Nevertheless, the computation of the Fisher information matrix makes the computation expensive, and approximation errors might have a detrimental effect. PCPO [42] introduces a cost projection into CPO and optimizes the policy in a two-step process. As et al. [3] utilize Bayesian world models to maximize optimistic upper bounds of reward and pessimistic upper bounds on safety constraints as well. Gu et al. [10] have extended CPO and Lagrangian methods to multi-agent reinforcement learning (MARL) and developed safe MARL benchmark suites.