Robust Task Representations for Offline Meta-Reinforcement Learning via Contrastive Learning

Haoqi Yuan [1] Zongqing Lu [1]

Abstract

We study offline meta-reinforcement learning, a practical reinforcement learning paradigm that learns from offline data to adapt to new tasks. The distribution of offline data is determined jointly by the behavior policy and the task. Existing offline meta-reinforcement learning algorithms cannot distinguish these factors, making task representations unstable to the change of behavior policies. To address this problem, we propose a contrastive learning framework for task representations that are *robust* to the distribution mismatch of behavior policies in training and test. We design a bi-level encoder structure, use mutual information maximization to formalize task representation learning, derive a contrastive learning objective, and introduce several approaches to approximate the true distribution of negative pairs. Experiments on a variety of offline meta-reinforcement learning benchmarks demonstrate the advantages of our method over prior methods, especially on the generalization to out-of-distribution behavior policies.

1. Introduction

Deep reinforcement learning (RL) has achieved great successes in playing video games (Mnih et al., 2013; Lample & Chaplot, 2017), robotics (Nguyen & La, 2019), recommendation systems (Zheng et al., 2018) and multi-agent systems (OroojlooyJadid & Hajinezhad, 2019). However, deep RL still faces two challenging problems: data efficiency and generalization. To play Atari games, model-free RL takes millions of steps of environment interactions, while model-based RL takes 100k environment steps (Łukasz Kaiser et al., 2020). The tremendous interactions along with safety issues prevent its applications in many real-world scenarios. Offline reinforcement learning (Levine et al., 2020) tackles the sample efficiency problem by learning from pre-collected offline datasets, without any online interaction. On the other hand, as deep RL is supposed to be tested in the same environment to the training environment, when the transition dynamics or the reward function changes, the performance degenerates. To tackle the generalization problem, meta-reinforcement learning (Finn et al., 2017; Duan et al., 2016) introduces learning over task distribution to adapt to new tasks.

Offline Meta-Reinforcement Learning (OMRL), an understudied problem, lies in the intersection of offline RL and meta-RL. Usually, the offline dataset is collected from multiple tasks by different behavior policies. The training agent aims at learning a meta-policy, which is able to efficiently adapt to unseen tasks. Recent studies (Li et al., 2020; Dorfman et al., 2020; Li et al., 2021a;b) extend context-based meta-RL to OMRL. They propose to use a context encoder to learn task representations from the collected trajectories, then use the latent codes as the policy's input.

However, context-based methods are vulnerable to the distribution mismatch of behavior policies in training and test phases. The distribution of collected trajectories depends both on the behavior policy and the task. When behavior policies are highly correlated with tasks in the training dataset, the context encoder is likely to memorize the feature of behavior policies. Thus, in the test phase, the context encoder produces biased task inference due to the change of the behavior policy. Although Li et al. (2021b;a) employed contrastive learning to improve task representations, their learning objectives are simply based on discriminating trajectories from different tasks, and thus cannot eliminate the influence of behavior policies.

To overcome this limitation, we propose a novel framework of **CO**ntrastive **R**obust task **R**epresentation learning for **O**MRL (**CORRO**). We design a bi-level structured task encoder, where the first level extracts task representations from one-step transition tuples instead of trajectories and the second level aggregates the representations. We formalize the learning objective as mutual information maximization between the representation and task, to maximally eliminate the influence of behavior policies from task representations.

[1]School of Computer Science, Peking University. Correspondence to: Zongqing Lu <zongqing.lu@pku.edu.cn>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

We introduce a contrastive learning method to optimize for InfoNCE, a mutual information lower bound. To approximate the negative pairs' distribution, we introduce two approaches for negative pairs generation, including generative modeling and reward randomization. Experiments in Point-Robot environment and multi-task MuJoCo benchmarks demonstrate the substantial performance gain of CORRO over prior context-based OMRL methods, especially when the behavior policy for adaptation is out-of-distribution.

Our main contributions, among others, are:

- We propose a framework for learning robust task representations with *fully* offline datasets, which can distinguish tasks from the distributions of transitions jointly determined by the behavior policy and task.
- We derive a contrastive learning objective to extract shared features in the transitions of the same task while capturing the essential variance of reward functions and transition dynamics across tasks.
- We empirically show on a variety of benchmarks that our method much better generalizes to out-of-distribution behavior policies than prior methods, even better than *supervised task learning* that assumes the ground-truth task descriptions.

2. Related Work

Offline Reinforcement Learning allows policy learning from data collected by arbitrary policies, increasing the sample efficiency of RL. In off-policy RL (Mnih et al., 2013; Wang et al., 2016; Haarnoja et al., 2018; Lillicrap et al., 2016; Peng et al., 2019), the policy reuses the data collected by its past versions, and learns based on Q-learning or importance sampling. Batch RL studies learning from fully offline data. Recent works (Lange et al., 2012; Fujimoto et al., 2019b; Levine et al., 2020; Wu et al., 2019; Fujimoto et al., 2019a; Kumar et al., 2019) propose methods to overcome distributional shift and value overestimation issues in batch RL. Our study follows the batch RL setting, but focuses on learning task representations and meta-policy from offline multi-task data.

Meta-Reinforcement Learning learns to quickly adapt to new tasks via training on a task distribution. Context-based methods (Duan et al., 2016; Zintgraf et al., 2020; Rakelly et al., 2019; Fakoor et al., 2020; Parisotto et al., 2019) formalize meta-RL as POMDP, regard tasks as unobservable parts of states, and encode task information from history trajectories. Optimization-based methods (Finn et al., 2017; Rothfuss et al., 2019; Stadie et al., 2018; Al-Shedivat et al., 2018; Foerster et al., 2018; Houthooft et al., 2018) formalize task adaptation as performing policy gradients over few-shot samples and learn an optimal policy initialization. Our study is based on the framework of context-based meta-RL.

Offline Meta-Reinforcement Learning studies learning to learn from offline data. Because there is a distributional mismatch between offline data and online explored data during test, learning robust task representations is an important issue. Recent works (Li et al., 2021b;a) apply contrastive learning over trajectories for compact representations, but ignore the influence of behavior policy mismatch. To fix the distributional mismatch, Li et al. (2020); Dorfman et al. (2020) assume *known* reward functions of different tasks, and Pong et al. (2021) require additional online exploration. Unlike them, we consider learning robust task representations with fully offline data.

Contrastive Learning (He et al., 2020; Grill et al., 2020; Wang & Isola, 2020) is a popular method for self-supervised representation learning. It constructs positive or negative pairs as noisy versions of samples with the same or different semantics. Via distinguishing the positive pair among a large batch of pairs, it extracts meaningful features. Contrastive learning has been widely applied in computer vision (Chen et al., 2020; He et al., 2020; Patrick et al., 2020), multi-modal learning (Sermanet et al., 2018; Tian et al., 2020; Liu et al., 2020b) and image-based reinforcement learning (Anand et al., 2019; Stooke et al., 2021; Laskin et al., 2020). In our study, we apply contrastive learning in offline task representation learning, for robust OMRL.

3. Preliminaries

3.1. Problem Formulation

A task for reinforcement learning is formalized as a fully observable Markov Decision Process (MDP). MDP is modeled as a tuple $M = (\mathcal{S}, \mathcal{A}, T, \rho, R, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $T(s'|s,a)$ is the transition dynamics of the environment, $\rho(s)$ is the initial state distribution, $R(s,a)$ is the reward function, $\gamma \in [0,1)$ is the factor discounting the future reward. The policy of the agent is a distribution $\pi(a|s)$ over actions. Starting from the initial state, for each time step, the agent performs an action sampled from π, then the environment updates the state with T and returns a reward with R. We denote the marginal state distribution at time t as $\mu_\pi^t(s)$. The objective of the agent is to maximize the expected cumulative rewards $\max_\pi J_M(\pi) = \mathbb{E}_{s_t\sim\mu_\pi^t, a_t\sim\pi}[\sum_{t=0}^\infty \gamma^t R(s_t,a_t)]$. To evaluate and optimize the policy, we define V-function and Q-function as follows:

$$V_\pi(s) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{s_t\sim\mu_\pi^t, a_t\sim\pi}\left[R(s_t,a_t)\right] \tag{1}$$

$$Q_\pi(s,a) = R(s,a) + \gamma \mathbb{E}_{s'\sim T(s'|s,a)}[V_\pi(s')]. \tag{2}$$

Q-learning solves for the optimal policy by iterating the Bellman optimality operator $\mathcal{B}$ over Q-function:

$$\mathcal{B}\hat{Q}(s,a) = R(s,a) + \gamma \mathbb{E}_{s'\sim T(s'|s,a)}\left[\max_{a'} \hat{Q}(s',a')\right]. \tag{3}$$

In offline meta-reinforcement learning (OMRL), we assume that the task follows a distribution $M_i = (\mathcal{S}, \mathcal{A}, T_i, \rho, R_i, \gamma) \sim P(M)$. Tasks share the same state space and action space, but varies in reward functions and transition dynamics. Thus, we can also denote the task distribution as $P(R, T)$. Given N training tasks $\{M_i\}_{i=1}^N$, for each task i, an offline dataset $X_i = \{(s_{i,j}, a_{i,j}, r_{i,j}, s'_{i,j})\}_{j=1}^K$ is collected by arbitrary behavior policy π_β^i. The learning algorithms can only access the offline datasets to train a meta-policy π_{meta}, without any environmental interactions. At test time, given an unseen task $M \sim P(M)$, an arbitrary exploration (behavior) policy collects a context $c = \{(s_j, a_j, r_j, s'_j)\}_{j=1}^k$, then the learned agent performs task adaptation conditioned on c to get a task-specific policy π_M and evaluate in the environment. The objective for OMRL is to learn a meta policy maximize the expected return over the test tasks:

$$J(\pi_{\text{meta}}) = \mathbb{E}_{M \sim P(M)}[J_M(\pi_M)] \,. \tag{4}$$

3.2. Context-Based OMRL and FOCAL

Context-based learning regards OMRL as solving partially observable MDPs. Considering the task M as the unobservable part of the state, the agent gathers task information and makes decisions upon the history trajectory: $a_t \sim \pi(a|\tau_{0:t-1}, s_t)$, where $\tau_{0:t-1} = (s_0, a_0, r_0, \cdots, s_{t-1}, a_{t-1}, r_{t-1})$.

A major assumption in context-based learning is that different tasks share some common structures and the variation over tasks can be described by a compact representation. For example, in a robotic object manipulation scenario, tasks can be described as object mass, friction, and target position, while rules of physical interaction are shared across tasks. Context-based OMRL uses a task encoder to learn a latent space $z_t = E(\tau_{0:t-1})$, to represent task information (Rakelly et al., 2019; Li et al., 2021b) or task uncertainty (Dorfman et al., 2020). The policy $\pi(a|s, z)$ is conditioned on the latent task representation. Given a new task, the exploration policy collects a few trajectories (context), and the task encoder adapts the policy by producing z. In principle, the task encoder and the policy can be trained on offline datasets with off-policy or batch RL methods. *In our setting, we assume that the exploration policy is arbitrary, and the context is a single trajectory.* During training, the context is sampled from the offline dataset.

FOCAL (Li et al., 2021b) improves task representation learning of the encoder via distance metric learning. The encoder minimizes the distance of trajectories from the same task and pushes away trajectories from different tasks in the latent space. The loss function is

$$\begin{aligned}\mathcal{L}_{dml} =& \mathbf{1}\{y_i = y_j\}\|q_i - q_j\|_2^2 + \\ & \mathbf{1}\{y_i \neq y_j\}\beta \cdot \frac{1}{\|q_i - q_j\|_2^n + \epsilon}\end{aligned} \tag{5}$$

where $q_i = E(\tau_i)$ is the representation of a trajectory, y_i is the identifier of the offline dataset. The policy is separately trained, conditioned on the learned encoder, with batch Q-learning methods.

3.3. Task Representation Problems in OMRL

Since the offline datasets are collected by different behavior policies, to answer which dataset a trajectory belongs to, one may infer based on the feature of behavior policy rather than rewards and state transitions. As an example: In a 2D goal reaching environment, tasks differ in goal positions. In each training task, the behavior policy is to go towards the goal position. Training on such datasets, the task encoder in FOCAL can simply ignore the rewards and distinguish the tasks based on the state-action distribution in the context. Thus, it will make mistakes when the context exploration policy changes.

Similar problem is also mentioned in Dorfman et al. (2020); Li et al. (2020). To eliminate the effects of behavior policies in task learning, Dorfman et al. (2020) augment the datasets by collecting trajectories in different tasks with the same policy, and relabeling the transition tuples with different tasks' reward functions. However, in our fully offline setting, collecting additional data and accessing the reward functions are not allowed. Li et al. (2020) use the learned reward functions in different tasks to relabel the trajectories and apply metric learning over trajectories. This approach does not support the settings where tasks differ in transition dynamics, because we cannot perform transition relabeling of s' over the trajectory to mimic the trajectory in other tasks. Also, the reward function learned with limited single-task dataset can be inaccurate to relabel the unseen state-action pairs.

4. Method

To address the offline task representation problem, we propose CORRO, a novel contrastive learning framework for robust task representations with fully offline datasets, which decreases the influence of behavior policies on task representations while supporting tasks that differ in reward function and transition dynamics.

4.1. Bi-Level Task Encoder

We first design a bi-level task encoder where representation learning focuses on transition tuples rather than trajectories. Concretely, our task encoder consists of a transition encoder E_{θ_1} and an aggregator E_{θ_2}, parameterized collectively as θ. Given a context $c = \{(s_i, a_i, r_i, s'_i)\}_{i=1}^k$, the transition encoder extracts latent representations for all the transition tuples $z_i = E_{\theta_1}(s_i, a_i, r_i, s'_i)$, then the aggregator gather all the latent codes into a task representation $z = E_{\theta_2}(\{z_i\}_{i=1}^k)$.

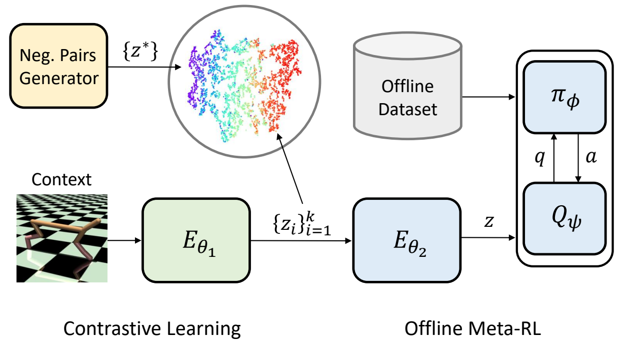

Figure 1. CORRO framework. The transition encoder E_{θ_1} extracts the latent representations of each transition tuple in the context. Contrastive learning is applied to robustify the task encoder. The aggregator E_{θ_2} gathers all the latent codes, to condition the actor and critic.

The Q-function $Q_\psi(s, a, z)$ and the policy $\pi_\phi(a|s, z)$ are conditioned on z, parameterized with ϕ, ψ.

Compared to trajectories, transition tuples leak less behavior policy information and are suitable for both transition relabeling and reward relabeling. Thus, we design robust task representation learning methods to train only the transition encoder in the following sections. The architecture of the transition encoder is simple MLP. The aggregator is trained with Q-function and policy using offline RL algorithms. Intuitively, it should draw attention to the transition tuples with rich task information. Inspired by self-attention (Vaswani et al., 2017), we introduce an aggregator which computes the weighted sum of the input vectors:

$$z = \sum_{j=1}^{k} \text{softmax}\left(\{\text{MLP}(z_i)\}_{i=1}^{k}\right)_j \cdot z_j. \tag{6}$$

Figure 1 gives an overview of our framework.

4.2. Contrastive Task Representation Learning

An ideally robust task representation should be dependent on tasks and invariant across behavior policies. We introduce mutual information to measure the mutual dependency between the task representation and the task. As, intuitively, mutual information measures the uncertainty reduction of one random variable when the other one is observed, we propose to maximize the mutual information between the task representation and the task itself, so that the task encoder learns to maximally reduce task uncertainty while minimally preserving task-irrelevant information.

Mathematically, we formalize the transition encoder E_{θ_1} as a probabilistic encoder $z \sim P(z|x)$, where $x = (s, a, r, s')$ denotes the transition tuple. Task M follows the task distribution $P(M)$ and the distribution of x is determined jointly by M and the behavior policy. The learning objective for the transition encoder is:

$$\max I(z; M) = \mathbb{E}_{z,M}\left[\log \frac{p(M|z)}{p(M)}\right]. \tag{7}$$

Optimizing mutual information is intractable in practice. Inspired by noise contrastive estimation (InfoNCE) in the literature of contrastive learning (Oord et al., 2018), we derive a lower bound of Eq. (7) and have the following theorem.

Theorem 4.1. *Let $\mathcal{M}$ be a set of tasks following the task distribution, $|\mathcal{M}| = N$. $M \in \mathcal{M}$ is the first task with reward and transition R, T. Let $x = (s, a, r, s'), z \sim P(z|x), h(x, z) = \frac{P(z|x)}{P(z)}$, where (s, a) follows an arbitrary distribution, $r = R(s, a), s' \sim T(s'|s, a)$. For any task $M^* \in \mathcal{M}$ with reward and transition R^*, T^*, denote $x^* = (s, a, r^*, s^{*\prime})$ as a transition tuple generated in M^* conditioned on (s, a), where $r^* = R^*(s, a), s^{*\prime} \sim T^*(s'|s, a)$. Then we have*

$$I(z; M) - \log(N) \geq \mathbb{E}_{\mathcal{M},x,z}\left[\log\left(\frac{h(x, z)}{\sum_{M^* \in \mathcal{M}} h(x^*, z)}\right)\right].$$

We give a proof of the theorem in Appendix A. Following InfoNCE, we approximate h with the exponential of a score function $S(z^*, z)$, which is a similarity measure between the latent codes of two samples. We derive a sampling version of the tractable lower bound to be the transition encoder's learning objective

$$\max_{\theta_1} \sum_{\substack{M_i \in \mathcal{M} \\ x,x' \in X_i}} \left[\log\left(\frac{\exp(S(z, z'))}{\sum_{M^* \in \mathcal{M}} \exp(S(z, z^*))}\right)\right], \tag{8}$$

where $\mathcal{M}$ is the set of training tasks, x, x' are two transition tuples sampled from dataset X_i and z, z' are latent representations of x, x'. For tasks $M^* \in \mathcal{M} \setminus \{M_i\}$, z^* is the representation of the transition tuple x^* sampled in task M^* conditioned on the same state-action pair in x. For $M^* = M_i$, we define $z^* = z'$ for consistent notation. We use cosine similarity for the score function in practice.

Following the literature, we name (x, x') a positive pair and name $\{(x, x^*)\}_{M^* \in \mathcal{M} \setminus \{M\}}$ negative pairs. Eq. (8) optimizes for a N-way classification loss to classify the positive pair out of all the pairs. To maximize the score of positive pairs, the transition encoder should extract shared features in the transitions of the same task. To decrease the score of negative pairs, the transition encoder should capture the essential variance of rewards and state transitions since state-action pairs are the same across tasks.

4.3. Negative Pairs Generation

Generating negative pairs in Eq. (8) involves computing (r, s') with tasks' reward functions and transition dynamics, which is impossible in the fully offline setting. Li et al. (2020) introduce a reward relabeling method that fits a reward model for each task using offline datasets. In our setting, we also need to fit transition models with higher dimensions. However, the dataset for each task is usually small, making reward models and transition models overfit. When the overlap of state-action pairs between different tasks is small, relabeling with separately learned models can be inaccurate, making contrastive learning ineffective.

To generate high-quality negative samples, we design methods based on two major principles.

Fidelity: To ensure the task encoder learning on real transition tuples, the distribution of generated negative pairs should approximate the true distribution in the denominator of Eq. (8). Concretely, given (s, a), the true distribution of (r, s') in negative pairs is

$$p(r, s'|s, a) \propto \mathbb{E}_{M\sim P(M)}[T(s'|s, a)\mathbf{1}\{R(s, a) = r\}] \quad (9)$$

Diversity: According to prior works (Patrick et al., 2020; Chen et al., 2020), the performance of contrastive learning depends on the diversity of negative pairs. More diversity increases the difficulty in optimizing InfoNCE and helps the encoder to learn meaningful representations. In image domains, large datasets, large negative batch sizes, and data augmentations are used to increase diversity. We adopt this principle in generating negative pairs of transitions.

We propose two approaches for negative pairs generation.

Generative Modeling: We observe that, though a state-action pair may not appear in all the datasets, it usually appears in a subset of training datasets. Fitting reward and transition models with larger datasets across tasks can reduce the prediction error. Thus, we propose to train a generative model over the union of training datasets $\cup_{i=1}^{N} X_i$, to approximate the offline data distribution $P_{\{X_i\}}(r, s'|s, a)$. When the distributions of (s, a) are similar across tasks, this distribution approximates the distribution in Eq. (9).

We adopt conditional VAE (CVAE) (Sohn et al., 2015) for generative modeling. CVAE consists of a generator $p_\xi(r, s'|s, a, z)$ and a probabilistic encoder $q_\omega(z|s, a, r, s')$. The latent vector z describes the uncertain factors for prediction, follows a prior Gaussian distribution $p(z)$. CVAE is trained to minimize the loss function

$$\begin{aligned}\mathcal{L}_{\text{CVAE}} = -\mathbb{E}_{(s,a,r,s')\in\{X_i\}}\Big[\mathbb{E}_{q_\omega(z|s,a,r,s')} \\ [\log p_\xi(r, s'|s, a, z)] - \text{KL}[q_\omega(z|s, a, r, s')\|p(z)]\Big],\end{aligned} \quad (10)$$

where $\text{KL}(\cdot\|\cdot)$ is KL-divergence, working as an information bottleneck to minimize the information in z.

Reward Randomization: In the cases where the overlap of state-action pairs between tasks is small, CVAE can collapse almost to a deterministic prediction model and the diversity in negative pairs decreases. When tasks only differ in reward functions, similar to data augmentation strategies in image domains, we can generate negative pairs via adding random noise to the reward. We have $r^* = r + \nu$, where the perturbation ν follows a noise distribution $p(\nu)$. Though reward randomization cannot approximate the true distribution of negative pairs, it provides an infinitely large space to generate diverse rewards, making contrastive learning robust.

Algorithm 1. Meta Training

```
Input: Datasets $\{X_i\}_{i=1}^N$; OMRL models $E_{\theta_1}$, $E_{\theta_2}$, $Q_\psi$, $\pi_\phi$
A. If use generative modeling, pre-train CVAE:
Initialize CVAE $q_\omega, p_\xi$
repeat
  Update $\omega, \xi$ to minimize Eq. (10).
until Done
B. Train the transition encoder:
repeat
  Sample a task $M$ and two transition tuples $x, x'$
  $z = E_{\theta_1}(x), z' = E_{\theta_1}(x')$
  for $M^* \in \mathcal{M}$ do
    if use generative modeling then
      Sample $x^*$ from CVAE
    else if use reward randomization then
      Add noise to the reward to get $x^*$
    end if
    $z^* = E_{\theta_1}(x^*)$
  end for
  Compute Eq. (8)
  Update $\theta_1$ to maximize Eq. (8)
until Done
C. Train the policy:
repeat
  Sample a task dataset $X$ and a context $c$
  $z = E_{\theta_2}(E_{\theta_1}(c))$
  Augment the states in $X$ with z
  Update $\theta_2, \psi, \phi$ with offline RL algorithms on $X$
until Done
```

4.4. Algorithm Summary

We summarize our training method in Algorithm 1. The test process is shown in Algorithm 2. The code for our work is available at https://github.com/PKU-AI-Edge/CORRO.

5. Experiments

In experiments, we aim to demonstrate: (1) The performance of CORRO on tasks adaptation in diverse task distributions and offline datasets; (2) The robustness of CORRO

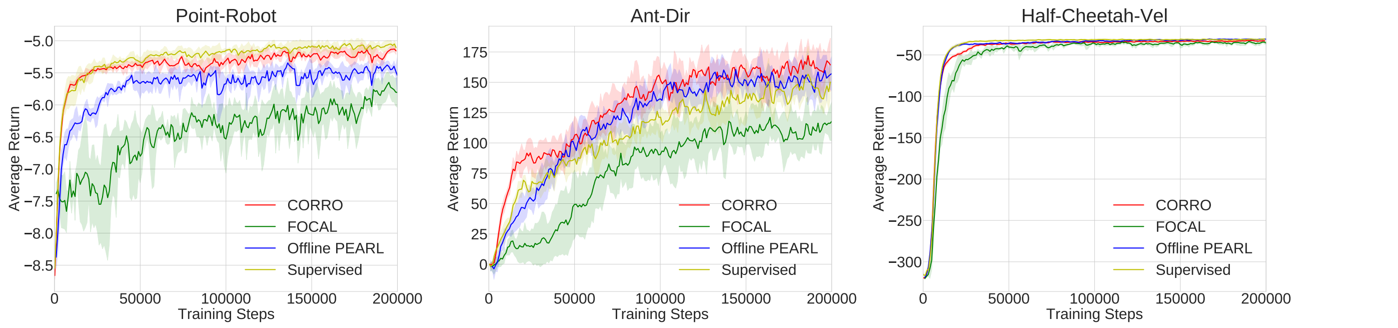

Figure 2. Test returns of CORRO against the baselines in the environments with different reward functions: Point-Robot, Ant-Dir, and Half-Cheetah-Vel. The shaded region shows standard deviation across 5 seeds.

Algorithm 2. Meta Test

```
Input: Trained models $E_{\theta_1}$, $E_{\theta_2}$, $Q_\psi$, $\pi_\phi$
Sample a task $M$
Collect a context trajectory $c$ with an arbitrary policy
$z = E_{\theta_2}(E_{\theta_1}(c))$
repeat
    Observe $s$, execute $a \sim \pi_\phi(a|s,z)$, get $r$
until Environment terminates
```

on task inference; and (3) How to choose the strategy of negative pairs generation.

5.1. Experimental Settings

We adopt a simple 2D environment and multi-task MuJoCo benchmarks to evaluate our method.

Point-Robot is a 2D navigation environment introduced in Rakelly et al. (2019). Starting from the initial point, the agent should navigate to the goal location. Tasks differ in *reward functions*, which describe the goal position. The goal positions are uniformly distributed in a square.

Half-Cheetah-Vel, Ant-Dir are multi-task MuJoCo benchmarks where tasks differ in *reward functions*. In Half-Cheetah-Vel, the task is specified by the target velocity of the agent. The distribution of target velocity is $U[0, v_{\max}]$. In Ant-Dir, the task is specified by the goal direction of the agent's motion. The distribution of goal direction is $U[0, 2\pi]$. These benchmarks are also used in Mitchell et al. (2021); Li et al. (2021b); Zintgraf et al. (2020).

Walker-Param, Hopper-Param are multi-task MuJoCo benchmarks where tasks differ in *transition dynamics*. For each task, the physical parameters of body mass, inertia, damping, and friction are randomized. The agent should adapt to the varying environment dynamics to accomplish the task. Previous works (Mitchell et al., 2021; Rakelly et al., 2019) also adopt these benchmarks.

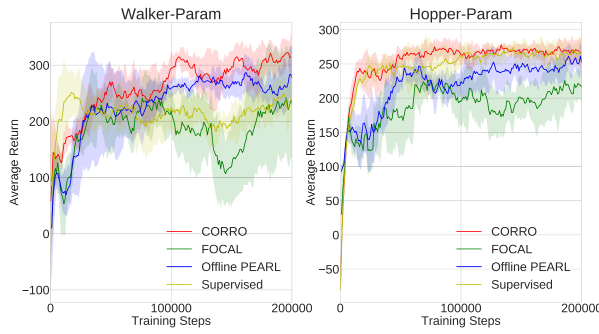

Figure 3. Test returns of CORRO against the baselines in the environments with different transition dynamics: Walker-Param and Hopper-Param. The shaded region shows standard deviation across 5 seeds.

For each environment, 20 training tasks and 20 testing tasks are sampled from the task distribution. On each task, we use SAC (Haarnoja et al., 2018) to train a single-task policy independently. The replay buffers are collected to be the offline datasets. During training, the context is a contiguous section of 200 transition tuples sampled from the offline dataset of the corresponding task. More details about experimental settings are available in Appendix B.

To evaluate the performance of our method, we introduce several representative context-based fully-offline methods to compare with.

Offline PEARL is a direct offline extension of PEARL (Rakelly et al., 2019). The context encoder is a permutation invariant MLP, updated together with policy and Q-function using offline RL algorithms. No additional terms are used for task representation learning. This baseline is introduced in (Li et al., 2020; Anonymous, 2022; Li et al., 2021b).

FOCAL (Li et al., 2021b) uses metric learning to train the context encoder. Positive and negative pairs are trajectories sampled from the same and the different tasks, respectively. Note that while the original FOCAL uses an offline RL

Table 1. Average test returns of CORRO against baselines with different types of context exploration policy. *IID* means the context distribution is the same to the training dataset, while *OOD* means the context is out-of-distribution.

Environment	**Supervised.**		**Offline PEARL**		**FOCAL**		**CORRO**	
	IID	*OOD*	*IID*	*OOD*	*IID*	*OOD*	*IID*	*OOD*
Point-Robot	**-4.89**±0.10	**-5.84**±0.14	-5.4±0.17	-6.74±0.19	-6.06±0.42	-7.34±0.20	**-5.19**±0.05	**-6.39**±0.05
Ant-Dir	136±17.6	131.7±11.4	155.4±24.4	141.5±11.3	109.8±12.8	53.5±16.4	**156.8**±35.2	**154.7**±25.8
Half-Cheetah-Vel	**-31.6**±0.7	**-32.1**±0.9	**-31.2**±0.5	-242.7±6.0	-38.0±4.0	-204.1±9.5	-33.7±1.1	**-89.7**±7.4
Walker-Param	232.7±29.2	221.2±43.4	259.1±48.2	254.7±35.8	225.4±56.4	193.3±151.5	**301.5**±37.9	**284.0**±19.3
Hopper-Param	**269.2**±20.3	251.9±28.8	244.0±18.5	236.6±18.5	195.6±62.3	199.7±51.9	**267.6**±25.6	**268.0**±13.8

method BRAC (Wu et al., 2019), we implement FOCAL with SAC to better study the task representation learning problem.

Supervised Task Learning assumes that *the ground truth task descriptions* can be accessed during training. Based on PEARL, we add L_2 loss to supervise the context encoder. We introduce this supervised learning method as a strong baseline. Note that the ground truth task labels are usually *unavailable* in meta-RL settings.

For fairness, we replace the policy learning methods of baselines and CORRO with SAC. The hyperparameters for policy learning are fixed across different methods. All the experiments are conducted over 5 different random seeds. All hyperparameters are available in Appendix C.

5.2. Tasks Adaptation Performance

To evaluate the performance on tasks adaptation, we use tasks in the test set, sample one trajectory in the pretrained replay buffer of each task to be the context, then test the agent in the environment conditioned on the context. The performance is measured by the average return over all the test tasks. In this experiment, there is *no distributional shift of contexts* between training and test, since they are generated by the single-task policies learned with the same algorithm.

Figure 2 shows the test results in the environments with different reward functions. In Point-Robot, CORRO and the supervised baseline outperform others. In Ant-Dir, CORRO outperforms all the baselines including the supervised learning method. Although the value of return in Point-Robot varies a little, we have to mention that the return in this environment is insensitive to the task performance: the mean return of an optimal policy is around -5, while the mean return of a random policy is around -9. Benefited from the pretrained task encoder, CORRO shows great learning efficiency, and reaches high test returns after 20k steps of offline training. Though in Half-Cheetah-Vel we cannot distinguish the performance of different methods, we will demonstrate their difference when context distribution shifts exist in Section 5.3 and the difference on the learned representations in Section 5.4.

Figure 3 demonstrates the results in the environments with varying transition dynamics. In both Walker-Param and Hopper-Param, CORRO outperforms the baselines on task adaptation returns. Further, CORRO is more stable on policy improvement, while the performance of FOCAL sometimes degenerates during the offline training.

5.3. Robust Task Inference

A robust task encoder should capture a compact representation of rewards and transition dynamics from the context, eliminating the information of the behavior policy. Conditioned on the context collected by an arbitrary behavior policy, the agent performs accurate task adaptation and reaches high returns. To measure the robustness, we design experiments of out-of-distribution (*OOD*) test. Denote the saved checkpoints of data collection policies for all the tasks during different training periods as $\{\pi_\beta^{i,t}\}_{i,t}$, where i is the task and t is the training epoch. For each test task M_j^{test}, we sample $\pi_\beta^{i,t}$ as the behavior policy to collect a context rollout, and test the agent conditioned on the context. Because $\pi_\beta^{i,t}$ is a policy trained on an arbitrary task, the context distribution is never seen during meta training. Then, we sample S behavior policies, measure the *OOD* test performance as $\frac{1}{NS}\sum_{k=1}^{S}\sum_{j=1}^{N} J(M_j^{\text{test}}, \pi_\beta^{i_k,t_k})$, where $J(M,\pi)$ means the average return in task M with context collection policy π. Accordingly, we denote the standard in-distribution test in Section 5.2 as *IID* test.

Table 1 summarizes the *IID* and *OOD* test results of different methods. All the test models are from the last training epoch. While the performance of all the methods degenerates from *IID* to *OOD*, CORRO outperforms Offline PEARL and FOCAL in *OOD* test by a large margin. In Half-Cheetah-Vel, Offline PEARL and FOCAL almost fail in task inference with distributional shift. The slight performance degeneration shows the robustness of CORRO. Note that although the supervised task learning method does not degenerate in

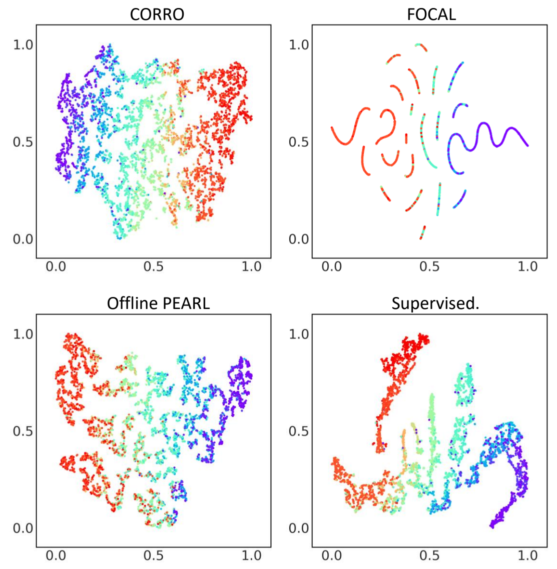

Figure 4. The 2D projection of the learned task representation space in Half-Cheetah-Vel. Points are uniformly sampled from test tasks. Tasks of velocities from 0 to 3 are mapped to rainbow colors, from red to purple.

Half-Cheetah-Vel, it uses the true task labels to learn the relation between contexts and tasks, which is not allowed in common meta-RL settings.

5.4. Latent Space Visualization

To qualitatively analyze the learned task representation space, we visualize the task representations by projecting the embedding vectors into 2D space via t-SNE (Maaten & Hinton, 2008). For each test task, we sample 200 transition tuples in the replay buffer to visualize. As shown in Figure 4, compared to the baselines, CORRO distinguishes transitions from different tasks better. CORRO represents the task space of Half-Cheetah-Vel into a 1D manifold, where the distance between latent vectors is correlated to the difference between their target velocities, as shown in the color of the points.

5.5. Ablation: Negative Pairs Generation

Negative pairs generation is a key component of CORRO. Generating more diverse negative samples can make contrastive learning robust and efficient, while the distribution of negative pairs should also be close to the true distribution as described in Section 4.2. Along with our design of generative modeling and reward randomization, we also introduce **Relabeling** and **None** method. **Relabeling** is to separately learn a reward model and a transition model on each offline training dataset. The negative pair is generated by first sampling a model, then relabeling (r, s') in the transition tuple. **None** is a straightforward method where we simply construct negative pairs with transition tuples from different tasks and (s, a) in the transitions can be different.

Table 2. Ablation study on negative pairs generation methods in Half-Cheetah-Vel environment (the upper four rows) and Point-Robot environment (the lower four rows).

Method	Contrastive Loss	*IID* Return	*OOD* Return
Generative	0.07	**-33.7**$\pm$1.1	-89.7$\pm$7.4
Randomize	0.83	-34.3$\pm$1.5	**-84.5**$\pm$1.3
Relabeling	0.04	-40.8$\pm$1.5	-245.3$\pm$12.9
None	1.20	-34.1$\pm$2.4	-97.6$\pm$3.1
Generative	2.83	-9.41$\pm$0.42	-9.42$\pm$0.42
Randomize	0.54	**-5.19**$\pm$0.05	**-6.39**$\pm$0.05
Relabeling	0.04	-9.22$\pm$0.24	-9.27$\pm$0.22
None	1.46	-5.24$\pm$0.27	-6.52$\pm$0.08

As shown in Table 2, in Half-Cheetah-Vel, our methods using generative modeling and reward randomization outperform None and Relabeling in *OOD* test, due to better approximating the distribution of negative pairs.

In Half-Cheetah-Vel, we notice that the None method achieves the average return of -97.6 in *OOD* test and outperforms FOCAL, of which the average return is -204.1 according to Table 1. They have two main differences: 1. samples for representation learning are transition tuples in 'None', but are trajectories in FOCAL; 2. the contrastive loss. To find which affects the OOD generalization more, we re-implement FOCAL with our InfoNCE loss. It achieves an average return of -41.9 in the IID test and -208.2 in the OOD test, which is close to FOCAL's performance. Thus, we argue that focusing on transition tuples rather than whole trajectories contributes to OOD generalization. Our approach naturally discourages the task encoder to extract the feature of behavior policy from the trajectory sequence.

While generative modeling and reward randomization has similar performance in Half-Cheetah-Vel, the latter solves Point-Robot better. Because in Point-Robot, behavior policies in different tasks move in different directions, making state-action distributions non-overlapping. CVAE collapses into predicting the transition of the specific task where the state-action pair appears, and cannot provide diverse negative pairs to support contrastive learning.

The column of contrastive loss shows that the two proposed methods generate diverse and effective negative pairs in the right circumstances, improving the learning of task encoder. Though the relabeling method has the best sample diversity, it underperforms other methods in task adaptation due to the inaccurate predictions on unseen data.

Table 3. Average test returns of CORRO against baselines with a random exploration policy for context collection.

Environment	Supervised.	Offline PEARL	FOCAL	CORRO
Point-Robot	-5.32±0.20	-7.06±0.99	-8.64±0.26	**-5.59**±0.57
Ant-Dir	149.8±20.5	148.4±35.3	89.8±8.7	**163.0**±35.8
Half-Cheetah-Vel	-37.6±0.8	**-35.4**±1.8	-41.6±3.3	-42.9±0.7
Walker-Param	221.7±91.1	276.6±37.7	245.6±67.8	**300.5**±34.2
Hopper-Param	253.5±21.2	245.9±18.9	203.6±46.6	**273.3**±3.9

5.6. Adaptation with an Exploration Policy

Context exploration is a crucial part of meta-RL. Existing works either explore with the meta policy before adaptation (Zintgraf et al., 2020; Finn et al., 2017), or learn a separate exploration policy (Liu et al., 2020a; Zhang et al., 2020). To test the task adaptation performance of CORRO with an exploration policy, we adopt a uniformly random policy to collect the context. As shown in Table 3, CORRO and the supervised baseline outperform others in Point-Robot. In Half-Cheetah-Vel, all the methods have great adaptation performance compared with the results in Table 1. In the other three environments, CORRO outperforms all the baselines.

6. Conclusion

In this paper, we address the task representation learning problem in the fully offline setting of meta-RL. We propose a contrastive learning framework over transition tuples and several methods for negative pairs generation. In experiments, we demonstrate how our method outperforms prior methods in diverse environments and task distributions. We introduce *OOD* test to quantify the superior robustness of our method and ablate the choice of negative pairs generation. Though, our study has not considered to learn an exploration policy for context collection, and to perform moderate environmental interactions when training data is extremely limited. We leave these directions as future work.

Acknowledgements

We would like to thank the anonymous reviewers for their useful comments to improve our work. This work is supported in part by NSF China under grant 61872009.

References

Al-Shedivat, M., Bansal, T., Burda, Y., Sutskever, I., Mordatch, I., and Abbeel, P. Continuous Adaptation Via Meta-Learning in Nonstationary and Competitive Environments. In *International Conference on Learning Representations*, 2018.

Anand, A., Racah, E., Ozair, S., Bengio, Y., Côté, M.-A., and Hjelm, R. D. Unsupervised State Representation Learning in Atari. In *Advances in neural information processing systems*, 2019.

Anonymous. Model-Based Offline Meta-Reinforcement Learning with Regularization. In *Submitted to The Tenth International Conference on Learning Representations*, 2022. under review.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A Simple Framework for Contrastive Learning of Visual Representations. In *International Conference on Machine Learning*, 2020.

Dorfman, R., Shenfeld, I., and Tamar, A. Offline Meta Learning of Exploration. *arXiv preprint arXiv:2008.02598*, 2020.

Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., and Abbeel, P. RL2: Fast Reinforcement Learning Via Slow Reinforcement Learning. *arXiv preprint arXiv:1611.02779*, 2016.

Fakoor, R., Chaudhari, P., Soatto, S., and Smola, A. J. Meta-Q-Learning. In *International Conference on Learning Representations*, 2020.

Finn, C., Abbeel, P., and Levine, S. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *International Conference on Machine Learning*, 2017.

Foerster, J., Farquhar, G., Al-Shedivat, M., Rocktäschel, T., Xing, E., and Whiteson, S. DiCE: The Infinitely Differentiable Monte Carlo Estimator. In *International Conference on Machine Learning*, 2018.

Fujimoto, S., Conti, E., Ghavamzadeh, M., and Pineau, J. Benchmarking Batch Deep Reinforcement Learning Algorithms. *arXiv preprint arXiv:1910.01708*, 2019a.

Fujimoto, S., Meger, D., and Precup, D. Off-Policy Deep Reinforcement Learning Without Exploration. In *International Conference on Machine Learning*, 2019b.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. A., Guo,

Z. D., Azar, M. G., Piot, B., Kavukcuoglu, K., Munos, R., and Michal, V. Bootstrap Your Own Latent: A New Approach To Self-Supervised Learning. In *Advances in neural information processing systems*, 2020.

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*, 2018.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum Contrast for Unsupervised Visual Representation Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Houthooft, R., Chen, R. Y., Isola, P., Stadie, B. C., Wolski, F., Ho, J., and Abbeel, P. Evolved Policy Gradients. In *Advances in neural information processing systems*, 2018.

Kumar, A., Fu, J., Tucker, G., and Levine, S. Stabilizing Off-Policy Q-Learning Via Bootstrapping Error Reduction. In *Advances in Neural Information Processing Systems*, 2019.

Lample, G. and Chaplot, D. S. Playing FPS Games with Deep Reinforcement Learning. In *AAAI Conference on Artificial Intelligence*, 2017.

Lange, S., Gabel, T., and Riedmiller, M. Batch Reinforcement Learning. In *Reinforcement learning*, pp. 45–73. Springer, 2012.

Laskin, M., Srinivas, A., and Abbeel, P. Curl: Contrastive Unsupervised Representations for Reinforcement Learning. In *International Conference on Machine Learning*, 2020.

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.

Li, J., Vuong, Q., Liu, S., Liu, M., Ciosek, K., Ross, K., Christensen, H. I., and Su, H. Multi-Task Batch Reinforcement Learning with Metric Learning. In *International Conference on Learning Representations*, 2020.

Li, L., Huang, Y., Chen, M., Luo, S., Luo, D., and Huang, J. Provably Improved Context-Based Offline Meta-RL with Attention and Contrastive Learning. *arXiv preprint arXiv:2102.10774*, 2021a.

Li, L., Yang, R., and Luo, D. FOCAL: Efficient Fully-Offline Meta-Reinforcement Learning Via Distance Metric Learning and Behavior Regularization. In *International Conference on Learning Representations*, 2021b.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous Control with Deep Reinforcement Learning. In *International Conference on Learning Representations*, 2016.

Liu, E. Z., Raghunathan, A., Liang, P., and Finn, C. Explore Then Execute: Adapting Without Rewards Via Factorized Meta-Reinforcement Learning. *arXiv preprint arXiv:2008.02790*, 2020a.

Liu, Y., Yi, L., Zhang, S., Fan, Q., Funkhouser, T., and Dong, H. P4Contrast: Contrastive Learning with Pairs of Point-Pixel Pairs for RGB-D Scene Understanding. *arXiv preprint arXiv:2012.13089*, 2020b.

Maaten, L. v. d. and Hinton, G. Visualizing Data Using T-SNE. *Journal of Machine Learning Research*, 9(11): 2579–2605, 2008.

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*, 2021.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing Atari with Deep Reinforcement Learning. *arXiv preprint arXiv:1312.5602*, 2013.

Nguyen, H. and La, H. Review of Deep Reinforcement Learning for Robot Manipulation. In *IEEE International Conference on Robotic Computing*, 2019.

Oord, A. v. d., Li, Y., and Vinyals, O. Representation Learning with Contrastive Predictive Coding. *arXiv preprint arXiv:1807.03748*, 2018.

OroojlooyJadid, A. and Hajinezhad, D. A Review of Cooperative Multi-Agent Deep Reinforcement Learning. *arXiv preprint arXiv:1908.03963*, 2019.

Parisotto, E., Ghosh, S., Yalamanchi, S. B., Chinnaobireddy, V., Wu, Y., and Salakhutdinov, R. Concurrent Meta Reinforcement Learning. *arXiv preprint arXiv:1903.02710*, 2019.

Patrick, M., Asano, Y. M., Kuznetsova, P., Fong, R., Henriques, J. F., Zweig, G., and Vedaldi, A. Multi-Modal Self-Supervision From Generalized Data Transformations. *arXiv preprint arXiv:2003.04298*, 2020.

Peng, X. B., Kumar, A., Zhang, G., and Levine, S. Advantage-Weighted Regression: Simple and Scalable Off-Policy Reinforcement Learning. *arXiv preprint arXiv:1910.00177*, 2019.

Pong, V. H., Nair, A., Smith, L., Huang, C., and Levine, S. Offline Meta-Reinforcement Learning with Online Self-Supervision. *arXiv preprint arXiv:2107.03974*, 2021.

Rakelly, K., Zhou, A., Quillen, D., Finn, C., and Levine, S. Efficient Off-Policy Meta-Reinforcement Learning Via Probabilistic Context Variables. In *International Conference on Machine Learning*, 2019.

Rothfuss, J., Lee, D., Clavera, I., Asfour, T., and Abbeel, P. ProMP: Proximal Meta-Policy Search. In *International Conference on Learning Representations*, 2019.

Sermanet, P., Lynch, C., Chebotar, Y., Hsu, J., Jang, E., Schaal, S., Levine, S., and Brain, G. Time-Contrastive Networks: Self-Supervised Learning From Video. In *IEEE International Conference on Robotics and Automation*, 2018.

Sohn, K., Lee, H., and Yan, X. Learning Structured Output Representation using Deep Conditional Generative Models. In *Advances in neural information processing systems*, 2015.

Stadie, B. C., Yang, G., Houthooft, R., Chen, X., Duan, Y., Wu, Y., Abbeel, P., and Sutskever, I. Some Considerations on Learning To Explore Via Meta-Reinforcement Learning. *arXiv preprint arXiv:1803.01118*, 2018.

Stooke, A., Lee, K., Abbeel, P., and Laskin, M. Decoupling Representation Learning From Reinforcement Learning. In *International Conference on Machine Learning*, 2021.

Tian, Y., Krishnan, D., and Isola, P. Contrastive Multiview Coding. In *European Conference on Computer Vision*, 2020.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is All You Need. In *Advances in neural information processing systems*, 2017.

Wang, T. and Isola, P. Understanding Contrastive Representation Learning Through Alignment and Uniformity on The Hypersphere. In *International Conference on Machine Learning*, 2020.

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., and Freitas, N. Dueling Network Architectures for Deep Reinforcement Learning. In *International Conference on Machine Learning*, 2016.

Wu, Y., Tucker, G., and Nachum, O. Behavior Regularized Offline Reinforcement Learning. *arXiv preprint arXiv:1911.11361*, 2019.

Zhang, J., Wang, J., Hu, H., Chen, Y., Fan, C., and Zhang, C. Learn to effectively explore in context-based meta-rl. *arXiv preprint arXiv:2006.08170*, 2020.

Zheng, G., Zhang, F., Zheng, Z., Xiang, Y., Yuan, N. J., Xie, X., and Li, Z. DRN: A Deep Reinforcement Learning Framework for News Recommendation. In *World Wide Web Conference*, 2018.

Zintgraf, L., Shiarlis, K., Igl, M., Schulze, S., Gal, Y., Hofmann, K., and Whiteson, S. VariBAD: A Very Good Method for Bayes-Adaptive Deep RL Via Meta-Learning. In *International Conference on Learning Representations*, 2020.

Łukasz Kaiser, Babaeizadeh, M., Miłos, P., Osiński, B., Campbell, R. H., Czechowski, K., Erhan, D., Finn, C., Kozakowski, P., Levine, S., Mohiuddin, A., Sepassi, R., Tucker, G., and Michalewski, H. Model-Based Reinforcement Learning for Atari. In *International Conference on Learning Representations*, 2020.

A. Contrastive Learning Objective

In this section, we first introduce a lemma. Then we give a proof of the main theorem in the text.

Lemma A.1. *Given $M \sim P(M)$ with reward R and transition T, (s,a) follows some distribution, $x = (s,a,r,s')$, $r = R(s,a)$, $s' \sim T(s'|s,a)$, $z \sim P(z|x)$, then we have*

$$\frac{P(M|z)}{P(M)} = \mathbb{E}_x\left[\frac{P(z|x)}{P(z)}\right].$$

Proof.

$$\begin{aligned}\frac{P(M|z)}{P(M)} &= \frac{P(z|M)}{P(z)}\\ &= \int_x \frac{P(x|M)P(z|x)}{P(z)}dx\\ &= \mathbb{E}_x\left[\frac{P(z|x)}{P(z)}\right]\end{aligned}$$

□

Theorem A.2. *Let $\mathcal{M}$ be a set of tasks following the task distribution, $|\mathcal{M}| = N$. $M \in \mathcal{M}$ is the first task with reward and transition R, T. Let $x = (s,a,r,s')$, $z \sim P(z|x)$, $h(x,z) = \frac{P(z|x)}{P(z)}$, where (s,a) follows an arbitrary distribution, $r = R(s,a)$, $s' \sim T(s'|s,a)$. For any task $M^* \in \mathcal{M}$ with reward and transition R^*, T^*, denote $x^* = (s,a,r^*,s^{*\prime})$ as a transition tuple generated in M^* conditioned on (s,a), where $r^* = R^*(s,a)$, $s^{*\prime} \sim T^*(s'|s,a)$. Then we have*

$$I(z;M) - \log(N) \geq \mathbb{E}_{\mathcal{M},x,z}\left[\log(\frac{h(x,z)}{\sum_{M^*\in\mathcal{M}} h(x^*,z)})\right].$$

Proof. Using Lemma A.1 and Jensen's inequality, we have

$$\begin{aligned}I(z;M) - \log(N) &= \mathbb{E}_{z,\mathcal{M}}\log\left[\frac{p(M|z)}{p(M)}\right] - \log N\\ &= \mathbb{E}_{z,\mathcal{M}}\left[\log \mathbb{E}_x\left[\frac{p(z|x)}{p(z)}\right]\right] - \log N\\ &\geq \mathbb{E}_{z,\mathcal{M}}\mathbb{E}_x\left[\log\left(\frac{1}{\frac{p(z)}{p(z|x)}N}\right)\right]\\ &\geq \mathbb{E}_{z,\mathcal{M}}\mathbb{E}_x\left[\log\left(\frac{1}{1+\frac{p(z)}{p(z|x)}(N-1)}\right)\right]\\ &= \mathbb{E}_{z,\mathcal{M}}\mathbb{E}_x\left[-\log\left(1+\frac{p(z)}{p(z|x)}(N-1)\right)\right]\\ &= \mathbb{E}_{z,\mathcal{M}}\mathbb{E}_x\left[-\log\left(1+\frac{p(z)}{p(z|x)}(N-1)\mathbb{E}_{M^*\in\mathcal{M}\backslash\{M\}}\left[\frac{p(z|x^*)}{p(z)}\right]\right)\right]\\ &= \mathbb{E}_{z,\mathcal{M}}\mathbb{E}_x\left[-\log\left(1+\frac{p(z)}{p(z|x)}\sum_{M^*\in\mathcal{M}\backslash\{M\}}\frac{p(z|x^*)}{p(z)}\right)\right]\\ &= \mathbb{E}_{\mathcal{M},x,z}\left[\log\left(\frac{\frac{p(z|x)}{p(z)}}{\frac{p(z|x)}{p(z)}+\sum_{M^*\in\mathcal{M}\backslash\{M\}}\frac{p(z|x^*)}{p(z)}}\right)\right]\\ &= \mathbb{E}_{\mathcal{M},x,z}\left[\log(\frac{h(x,z)}{\sum_{M^*\in\mathcal{M}} h(x^*,z)})\right]\end{aligned}$$

□

B. Environment Details

In this section, we present the details of the environments in our experiments.

Point-Robot: The start position is fixed at $(0, 0)$ and the goal location g is sampled from $U[-1, 1] \times U[-1, 1]$. The reward function is defined as $r_t = -\|s_t - g\|_2$, where s_t is the current position. The maximal episode steps is set to 20.

Ant-Dir: The goal direction is sampled from $\theta \sim U[0, 2\pi]$. The reward function is $r_t = v_x \cos\theta + v_y \sin\theta$, where (v_x, v_y) is the horizontal velocity of the ant. The maximal episode steps is set to 200.

Half-Cheetah-Vel: The goal velocity is sampled from $v_g \sim U[0, 3]$. The reward function is $r_t = -|v_t - v_g| - \frac{1}{2}\|a_t\|_2^2$, where v_t is the current forward velocity of the agent and a_t is the action. The maximal episode steps is set to 200.

Walker-Param: Transition dynamics are varied in body mass and frictions, described by 32 parameters. Each parameter is sampled by multiplying the default value with $1.5^\mu, \mu \sim U[-3, 3]$. The reward function is $r_t = v_t - 10^{-3} \cdot \|a_t\|_2^2 + 1$, where v_t is the current forward velocity of the walker and a_t is the action. The episode terminates when the height of the walker is less than 0.5. The maximal episode steps is also set to 200.

Hopper-Param: Transition dynamics are varied in body mass, inertia, damping and frictions, described by 41 parameters. Each parameter is sampled by multiplying the default value with $1.5^\mu, \mu \sim U[-3, 3]$. The reward function is $r_t = v_t - 10^{-3} \cdot \|a_t\|_2^2$, where v_t is the current forward velocity of the hopper and a_t is the action. The maximal episode steps is set to 200.

C. Experimental Details

In Table 4 and 5, we list the important configurations and hyperparameters in the data collection and meta training phases that we used to produce the experimental results.

Table 4. Configurations and hyperparameters used in dataset collection to produce all the experimental results.

Configutations	Point-Robot	Ant-Dir	Half-Cheetah-Vel	Walker-Param	Hopper-Param
Dataset size	2100	2e4	2e5	2e4	6e4
Training steps	1e3	4e4	1e5	6e5	2e5
Batch size	256	256	256	256	256
Network width	32	128	128	128	128
Network depth	3	3	3	3	3
Learning rate	3e-4	3e-4	3e-4	3e-4	3e-4

Table 5. Configurations and hyperparameters used in offline meta training to produce all the experimental results.

Configutations	Point-Robot	Ant-Dir	Half-Cheetah-Vel	Walker-Param	Hopper-Param
Negative pairs	Randomize	Randomize	Generative	Generative	None
$p(\nu)$	$\mathcal{N}(0, 0.5)$	$\mathcal{N}(0, 0.5)$	–	–	–
Latent space dim	5	5	5	32	40
Task batch size	16	16	16	16	16
Training steps	2e5	2e5	2e5	2e5	2e5
RL batch size	256	256	256	256	256
Contrastive batch size	64	64	64	64	64
Negative pairs number	16	16	16	16	16
RL network width	64	256	256	256	256
RL network depth	3	3	3	3	3
Encoder width	64	64	64	128	128
Learning rate	3e-4	3e-4	3e-4	3e-4	3e-4