Conditional Meta-Reinforcement Learning with State Representation

Yuxuan Sun Laura Toni Yiannis Andreopoulos
Department of Electronic and Electrical Engineering
University College London
{yuxuan.sun.22,l.toni,i.andreopoulos}@ucl.ac.uk

Abstract

Reinforcement Learning (RL) has achieved remarkable success in diverse areas, yet its sample inefficiency—requiring extensive interactions to develop optimal policies—remains a challenge. Meta-Reinforcement Learning (Meta-RL) addresses this by leveraging previously acquired knowledge, often integrating contextual information into learning. This study delves into conditional Meta-RL, investigating how context influences learning efficiency. We introduce a novel theoretical framework for both unconditional and conditional Meta-RL scenarios, with a focus on approximating the value function using state representations in environments where the transition kernel is known. This framework lays the groundwork for understanding the advantages of conditional Meta-RL over unconditional Meta-RL approaches. Furthermore, we present a conditional Meta-RL algorithm that shown to offer more than 50 percent increase in the average return than unconditional setting in MiniGrid environments.

1 Introduction

Reinforcement learning has achieved unprecedented success in numerous fields [Mnih et al., 2015, Schulman et al., 2017]. However, it faces a major challenge in its typically low sample efficiency, necessitating extensive datasets to reach optimal performance. To address this challenge, various strategies, including Meta-Reinforcement Learning (Meta-RL) [Rakelly et al., 2019] and state representation learning, have been explored. The latter improves sample efficiency by combining the learning of a low-dimensional state representation from high-dimensional observations with the learning of the optimal policy [Wang et al., 2022].

Meta-RL leverages the insight that certain parameters are universally applicable across tasks, while others require task-specific tuning. This approach, foundational to the Model-Agnostic Meta-Learning (MAML) framework [Finn et al., 2017], divides learning into meta-training, where parameters are learned from a variety of tasks, and meta-testing, where these parameters are fine-tuned for individual tasks. Specifically in reinforcement learning, Meta-RL applies this concept to efficiently learn the value function, enabling quick adaptation to new tasks with minimal additional training. When the transition kernel is known, one method to address the problem is to learn a value function and subsequently selecting the action which chooses the state that maximizes the estimated value function. However, if the transition kernel is unknown, agents typically have two options: learning the transition kernel itself or an action-value function. Since the theory underlying the two cases is similar, we primarily focus on the first scenario where the transitional kernel is known [Konidaris et al., 2011].

Typical deep RL neural networks including DQN [Mnih et al., 2015] learn value functions through nonlinear functions in the hidden layer and the final layer is linearly mapping to a value function prediction. Therefore, the final layer can be seen as a state representation (Figure 1) [Le Lan et al., 2022]. As neural networks are commonly used in meta-RL [Liu et al., 2021, Zintgraf et al., 2021],

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

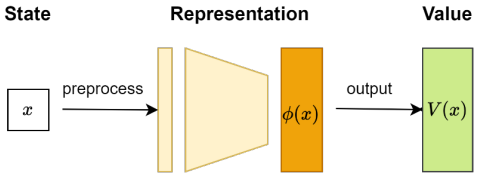


Figure 1: A deep RL architecture seen as a state representation $\phi(x)$ and a value prediction $V(x)$.

we can assume that state representations are learned for tasks in meta-RL, and all the tasks share the same state representations [Cheng et al., 2022, Chung et al., 2018].

Utilizing identical state representations across all tasks proves beneficial primarily in scenarios where tasks share a strong correlation [Cheng et al., 2022]. However, in real-world applications, tasks often span a wide spectrum of diversity with merely superficial similarities. For example, consider the development of a robotic vacuum cleaner: initial pretraining at the manufacturing site aims to equip the device with the versatility to adapt seamlessly to various home environments. This phase might involve classifying rooms by function—such as living rooms versus bedrooms—to tailor cleaning strategies accordingly. A singular, shared state representation might fall short in capturing the nuanced differences between these task categories, underscoring the challenge when tasks diverge into distinct clusters each demanding a specialized low-dimensional representation. Consequently, it becomes imperative to adopt a context-specific approach to representation learning. In our example, this would mean developing different representations for different room types, thereby optimizing the robotic cleaner's performance across varying domestic environments.

Context-based methods are used in meta-RL to improve sample efficiency and reach higher asymptotic performance [Rakelly et al., 2019, Zhang et al., 2021, Yuan and Lu, 2022]. In particular, it can solve the aforementioned problem, if we assume each task comes with a context indicating which cluster it comes from. However, the literature lacks a detailed theoretical analysis of how context influences the generalization of state representations within Meta-RL frameworks. Our study addresses this gap by rigorously examining the role of context in Meta-RL, providing theoretical evidence to support the efficacy of context utilization. We demonstrate that context-aware learning significantly bolsters the adaptability and efficiency of Meta-RL models, enabling them to navigate complex environments with nuanced differences more effectively.

In this paper, we focus on understanding the power of contextual information in meta-RL, introducing the concept of 'conditional meta-RL.' This approach is characterized by its use of contextual information to inform the learning process, which we refer to as 'conditional' due to the conditioning on specific task-related information. Conversely, approaches that do not leverage such contextual information are termed 'unconditional meta-RL.' Several works provided a general framework for conditional meta-supervised learning and provided a bound for the transfer risk [Denevi et al., 2020, 2022]. In this work, we extend this concept to the field of reinforcement learning.

2 Related work

Meta-reinforcement learning: Meta-RL integrates the principles of reinforcement learning with the concepts of meta-learning. Meta-RL algorithms leverage the common structure shared among tasks in the meta-training phase, effectively spreading the learning cost across these tasks. This allows for quick adaptation to novel tasks in the meta-testing phase, requiring only minimal samples [Finn et al., 2017, Xu et al., 2018].

These approaches typically necessitate on-policy meta-training and have been demonstrated to be sample inefficient in practice. To address these limitations, several algorithms have been introduced, focusing on meta-learning a policy with off-policy data and contextual information [Rakelly et al., 2019, Fu et al., 2021, Zhang et al., 2021]. Here, the contextual information is learned through a few interactions with the environment in the beginning. The contextual information should contain some latent information of the distribution of the tasks.Therefore, context-based meta-RL is expected to perform better on the generalization of the tasks.

Representation learning in RL: In reinforcement learning, state representations are utilized to manage large problem spaces effectively. These representations not only can be used in the approximation of the value function, but also aid in generalizing to states that are newly encountered [Le Lan et al., 2022]. These representations can be learned explicitly in linear approximations of the value functions [Konidaris et al., 2011, Mahadevan and Maggioni, 2007, Chung et al., 2018, Lyle et al., 2021, Parr et al., 2008], as well as implicitly in deep RL network architectures such as DQN [Mnih et al., 2015], where the final layer can be viewed as a state representation.

Learning a shared representation among a class of reinforcement learning tasks has been proven to be a powerful approach in multi-task setting [Cheng et al., 2022]. The shared state representation should acquire universal and expressive features across the environments, without being tailored to any specific policy [Jianye et al., 2022]. This approach has been demonstrated to be advantageous compared to solving each task independently. We want to take advantage of the linear representation model and apply it to meta-RL setting.

3 Background

We consider a Markov decision process $M = (S, A, R, P, \gamma)$ with finite state space S, action space A, state transition kernel P, reward function $R : S \quad A \; ! \; [\quad R_{\max}, R_{\max}]$, and the discount factor γ [Sutton and Barto, 2018]. For simplicity, we assume that $S = f1, \ldots, Sg$.

A stationary policy π is a mapping from states to distributions over actions. For any policy π, $V \; (s)$ is the value function. Let P be the $S \quad S$ transition matrix with $P \; (s, s^0)$ the transition probability from state s to s^0 under policy π. Then following the vector notation [Mahadevan and Maggioni, 2007], we have

$$V \quad = \sum_{t=0}^{1} (\gamma P \;)^t r \quad = (I \quad \gamma P \;) \quad {}^1 r \quad . \tag{1}$$

Here $r \quad 2 \; \mathsf{R}^S$ is the expected reward. We consider the following approximation of the value function with a linear combination of features [Parr et al., 2008]:

$$V \;_{;w}(s) = \phi(s)^T w \tag{2}$$

where $\phi : S \; ! \; \mathsf{R}^d$ is a function from states to d-dimensional representations and $w \; 2 \; \mathsf{R}^d$ are weights. Without loss of generality, we assume that $k\phi k \quad 1$ and $kwk \quad W_{\max}$. This can be expressed in a matrix form

$$V \;_{;w} = \quad w \tag{3}$$

where $\quad 2 \; \mathsf{R}^{S \quad d}$ is the feature matrix.

We define the Meta-RL setting as involving a family of tasks $M_i = (S, A, R_i, P_i, \gamma)$ drawn from a distribution $p(M)$. Assume that all the tasks share the same state space and action space, but have different reward functions and state transition probabilities [Dorfman et al., 2021, Liu et al., 2021, Mitchell et al., 2021, Zintgraf et al., 2019]. We are interested in the problem of value function approximation in (3) for each task, assuming that the transition probability matrix is known for each task but the reward needs to be learned. Namely, we are interested in learning a common feature matrix for all tasks, while w is the weight learnt for each single task. In the following sections, we will formalize the mathematical framework employed to define the problem of conditional Meta-RL.

3.1 Within-task objective

We consider the batch Monte Carlo policy evaluation setting, where we are provided with a group of training samples $Z = f(s_1, y_1), \ldots, (s_n, y_n)g \; 2 \; (S \quad \mathsf{R})^n$. Here s_i is a random initial state and y_i is a realization of the random return [Bellemare et al., 2017] defined by

$$G \;(s) = \sum_{t=0}^{1} \gamma^t R^{a_t}_{s_t}, \quad s_0 = s, a_t \quad \pi(\, j s_t). \tag{4}$$

Suppose the initial state follows a distribution ν. Then for a single task we want to minimize the expected squared error of a linear approximation $V \;_{;w}$

$$R_M(w) = R(V \;_{;w}) = \mathsf{E}_{y \;\; s \;\; G \;\; (s)} (V \;_{;w}(s) \quad y)^2 = \mathsf{E}_{y \;\; s \;\; G \;\; (s)} (\phi(s)^T w \quad y)^2. \tag{5}$$

In real-world applications, given samples $Z = \{(s_1, y_1), \ldots, (s_n, y_n)\} \in (S \times \mathbb{R})^n$, we minimize the regularized empirical risk function [Kolter and Ng, 2009, Petrik et al., 2010]

$$R_{Z;\lambda}(w) = \frac{1}{n}\sum_{i=1}^{n}(\phi(s_i)^T w - y_i)^2 + \frac{\lambda}{2}\|w\|^2, \tag{6}$$

where we have also added a regularizing term parameterized by λ, with λ assuming values either 0 or 1. When $\lambda = 1$, $R_{Z;1}(w)$ represents the regularized empirical risk function. Since we can change the bound $W_{\max}$ for w, we can actually control the amount of regularization applied to the model. Therefore, we assume $\lambda = 1$ for simplicity. When $\lambda = 0$, we get the empirical risk function $R_{Z;0}(w)$. Equipped with the above notation, the learning algorithm $A(\phi, \cdot) : \bigcup_{n=1}^{\infty}(S \times \mathbb{R})^n \to \mathbb{R}^d$ is defined as follows

$$A(\phi, Z) = \operatorname*{argmin}_{w \in \mathbb{R}^d} R_{Z;1}(w). \tag{7}$$

3.2 Unconditional meta-RL

The goal of unconditional meta-RL is to find the optimal policy for each inner task M_i. This object can be expressed as minimizing the transfer risk

$$\min_{\phi \in F} E_p(\phi), \quad E_p(\phi) = \mathbb{E}_{M_i \sim p(M)} \mathbb{E}_{Z \sim M_i} R_{M_i}(A(\phi, Z)) \tag{8}$$

where F is the function space of ϕ.

Standard meta-RL methods usually aim at finding the policy that maximizes the expected returns [Beck et al., 2023]. In this work, since we assume that the transition kernel is known and random returns are generated, we use a different objective. As we will show in Section 5, in real-world scenarios where the transitional kernel is unknown, we can set a target $v_{\text{old};c}(s_t)$ for value function and update it iteratively. Essentially, this means that instead of using y_i in equation (6), we will use $\phi_{\text{old}}(s_{t;i}, c)^T w$ as an approximation of y_i and update ϕ and w accordingly.

3.3 Conditional meta-RL

We assume that for all the tasks M_i, there is an additional side information $c \in C$. In this case, we consider a joint distribution of the environment and the side information $p(M, c)$. The concept of side information is broad, including descriptive context about the mission of a task [Chevalier-Boisvert et al., 2023] or the previous collected experiences [Rakelly et al., 2019]. Utilize the idea of contextual MDP [Amani et al., 2022], the feature function can be written as $\phi(s, c)$ where c is the context. We call this setting *conditional meta-RL* since the learning process for each task is conditioned on the contextual information. In conditional meta-RL, the aim is to solve the problem

$$\min_{(s;c) \in F^{\emptyset}} E_p(\phi, C), \quad E_p(\phi, C) = \mathbb{E}_{(M_i;c_i) \sim p(M;C)} \mathbb{E}_{Z \sim M_i} R_{M_i}(A(\phi(s, c_i), Z)) \tag{9}$$

where $F^{\emptyset}$ is the space of feature functions whose input contains the side information. Note that the unconditional meta-RL setting above can be viewed as restricting the side information to be a constant across all the clusters. In the following, we will show that the conditional setting achieves a lower transfer risk than unconditional setting.

4 The advantage of conditional meta-RL

To measure how well the conditional feature function setting approximates the true value function and assess the advantage of conditional meta-RL, we analyze the generalization properties of $\phi(s, c)$. To achieve this, we compare the conditional transfer risk $E_p(\phi, C)$ with the optimal minimum risk

$$E_p^* = \mathbb{E}_{M \sim p(M)} R_M(w_M^*) \quad w_M^* = \operatorname*{argmin}_{w \in \mathbb{R}^d} R_M(w). \tag{10}$$

In this Section, we will first prove a bound between the conditional transfer risk and the optimal minimum risk. Then, we compare the conditional transfer risk with its unconditional counterpart.

Now we want to measure the effectiveness of our conditional feature function $\phi(s, c)$ approximation. This is done by proving a bound on the difference between the conditional transfer risk $E_p(\phi, C)$ and the optimal minimum risk E_p. The proof is provided in Appendix.

Theorem 1. Let $(e_i)_{i=1}^{S}$ $\mathbb{R}^S$ be the standard basis and N , $\mathbb{E}_i$ $[e_i e_i^T]$. We use $_c$ to represent the feature matrix when context information exists. We assume further that $_c$ $^{\gamma}_c$ is diagonal, then

$$E_p(\phi, C) \quad E_p \quad \mathbb{E}_c \; {}_{p(C)}[\frac{R_{\max}^2}{2(1 \quad \gamma)^2} Tr((\; _c \; {}^T_c)^{\gamma} M(c)) + \frac{8}{n}(W_{\max} + \frac{R_{\max}}{1 \quad \gamma})^2 Tr(\; _c \; {}^T_c N(c))] \tag{11}$$

where $N(c) = \mathbb{E}_{M \;\; p(M|c)} N$ and $M(c) = \mathbb{E}_{M \;\; p(M|c)} N^{\gamma}$.

In conditional meta-RL, the aim is to minimize $E_p(\phi, C)$ in (9). From Theorem 1, we know that the feature function $\phi(s, c)$ that minimizes the upper bound in (11) is a potential optimal function for the conditional meta-RL problem. The upper bound in (11) contains the traces of both $^T_c N \;\; _c$ and its inverse. In the next proposition, we will find a feature function matrix $_c$ minimize the upper bound and provide a bound for the according excess risk.

Proposition 1. (Best conditioning function in hindsight) The minimizer of the bound in (11) in set F^0 is the feature function $\phi_p(s, c)$ that satisfies the condition

$$_c \; {}^T_c = \frac{p\overline{n} R_{\max} N(c)^{\gamma=2} (N(c)^{\frac{1}{2}} M(c) N(c)^{\frac{1}{2}})^{\frac{1}{2}} N(c)^{\gamma=2}}{4(R_{\max} + W_{\max}(1 \quad \gamma))}. \tag{12}$$

The according excess risk satisfies

$$E_p(\phi_p, C) \quad E_p \quad p\frac{4R_{\max}}{\overline{n}(1 \quad \gamma)}(W_{\max} + \frac{R_{\max}}{1 \quad \gamma})\mathbb{E}_c \; {}_{p(C)} Tr(M(c)N(c))^{\frac{1}{2}}. \tag{13}$$

Note that here we do not assume full ranks for $N(c)$ and $M(c)$, therefore, there can be many solutions $_c \; 2 \; \mathbb{R}^{S \;\; d}$ with different d. However, if we assume full ranks for both $N(c)$ and $M(c)$, we get $_c = \frac{p}{2}\frac{\overline{n}^p \overline{R_{\max}}}{(R_{\max} + W_{\max}(1 \quad))}((N(c)^{\gamma})^{\frac{3}{2}} M(c) N(c)^{\frac{1}{2}})^{\frac{1}{4}}$.

From Proposition 1, we know that the states with a smaller conditional expectation in the initial distribution, will have a feature vector with a larger norm. Besides, since $M(c)$, $N(c)$ and $N(c)^{\gamma=2}$ are diagonal, and all of them have ranks equal to the number of states have non-zero expectations in the initial state distribution ν, $_c \; {}^T_c$ has the same rank according to (12). Because $rank(\; _c) = rank(\; _c \; {}^T_c) = d \quad S$, we know that the dimension of the feature space d should be equal to the number of states that has non-zero expectations in the initial state distribution ν. In other words, for the best conditioning feature matrix $_c$, the dimension of the feature space d equals the number of states that actually appears in the initial distribution ν.

4.1 Conditional vs unconditional meta-RL

When applying Proposition 1 within the unconditional set F, by assuming the context c remains constant across clusters, we obtain

$$E_p(\phi_p) \quad E_p \quad p\frac{4R_{\max}}{\overline{n}(1 \quad \gamma)} \quad (W_{\max} + \frac{R_{\max}}{1 \quad \gamma}) Tr(\mathbb{E}_{M \;\; p(M)} N \quad \mathbb{E}_{M \;\; p(M)} N^{\gamma})^{1=2}. \tag{14}$$

Intuitively, the unconditional bound in (14) is always greater than conditional bound in (13), since the latter selects minimum over a larger set. If we look at bounds we get in (13) and (14), we can see that the difference lies between $\mathbb{E}_c \; {}_{p(C)} \left[\mathbb{E}_{M \;\; p(M|c)} N \quad \mathbb{E}_{M \;\; p(M|c)} N^{\gamma}\right]$ and $\mathbb{E}_{M \;\; p(M)} N \quad \mathbb{E}_{M \;\; p(M)} N^{\gamma}$. Basically, this means that in unconditional settings, the initial state distribution ν is assumed to be constant across the tasks, while in conditional settings, a context c indicates different initial state distributions for different tasks. This conditional expectation (conditional on the context) of the matrices N and N^{γ} cause the difference. Thus, when the initial distribution of states varies significantly across clusters, the disparity between the conditional and unconditional bounds will be substantial. Below we use an illustrative example to demonstrate the difference between conditional and unconditional setting.

Algorithm 1: Meta Training

Input: Learning rate > 0.
Output: Value function $v_{;c}(s)$ and policy $_{;c}$.
Init: Initialize policies $_{;c}$ and value functions $v_{;c}(s)$.
while $t < T$ **do**
  Sample tasks and side information $(M_t; c_t)$ $p(M; C)$.
  Sample dataset $Z_t = f(s_{t;i}; y_{t;i})_{i=1}^{n}g$ M_t^n with an equal probability for each initial state and current policy $_{;c_t}$, and $y_{t;i} = v_{\text{old};c}(s_{t;i}) = {}_{\text{old}}(s_{t;i}; c)^T w$ with ${}_{\text{old}}$ the feature function computed from the previous iteration.
  Update $(s; c)$, w and according to the PPO object loss functions $L_{\text{value}}^{PPO}(\ )$ and $L_{\text{policy}}^{PPO}(\ )$.
  t = t+1
end

Example Suppose there are two clusters of environments, each containing two states, s_1 and s_2. In the first cluster, the initial state distribution is $s_1 = 0.9, s_2 = 0.1$, whereas in the second cluster, the initial state distribution is $s_1 = 0.1, s_2 = 0.9$. Suppose the environments from the first cluster are selected with a probability of 0.9.

In this scenario, under the unconditional setting, the expected environment matrix, denoted as $E_{M \ p(M)}N$, is calculated as $diag(0.82, 0.18)$, and the pseudoinverse of this matrix, $E_{M \ p(M)}N^y$, is $diag(2, 9.11)$. This computation leads to an unconditional excess risk bound that is 1.64 times greater than the conditional excess risk bound.

Besides, the maximum of the true value function $\frac{R_{\max}}{1}$ and the maximum of the norm of the weights $W_{\max}$ can also affect the bound. The larger they are, the higher the bound. Moreover, the bound is inversely proportional to the square root of the number of samples n. Therefore, the more samples we have, the tighter the bound becomes.

5 Conditional meta-RL algorithm

In practice, the objective is to identify the optimal policy for each task within a given task family. Consequently, the random returns in (6) are for these optimal policies. However, direct access to such returns is typically unavailable. To circumvent this limitation, we utilize the value function estimated from the previous iteration as a surrogate for the actual returns. This estimated value function is then iteratively updated to refine our approximation of the optimal policy's returns. Essentially, this means that instead of using y_i in (6), we will use $\phi_{\text{old}}(s_{t;i}, c)^T w$ as an approximation of y_i and update ϕ and w accordingly.

In this section, we provide an actor-critic style algorithm based on PPO algorithm [Schulman et al., 2017], detailed in Algorithms 1 and 2 for the meta-training and meta-testing phases, respectively. This approach is adaptable to any deep Reinforcement Learning (RL) network architecture, where the final layer is conceptualized as a dynamic state representation from which the output value function is linearly derived [Le Lan et al., 2022]. Within this architecture, actor neural networks, denoted as $\pi_{;c}$, are employed to approximate the policy. Simultaneously, critic neural networks, expressed as $v_{;c}(s) = \phi(s, c)^T w$, serve to approximate the value function. Here, $\phi(s, c)$ represents the learned state representation at the final layer.

In a special case, if the critic neural network is structured to include only one hidden layer with a linear activation function, the state representation can be formulated as $\phi(s, c) = M_c\psi(s)$, where $\psi(s)$ signifies the initial features (or pre-processed features) of the state. For example, in the MiniGrid environments, the state s is the observation image of dimension $7 \quad 7 \quad 3$ and ψ is a convolutional neural network outputs the embedding of the image.

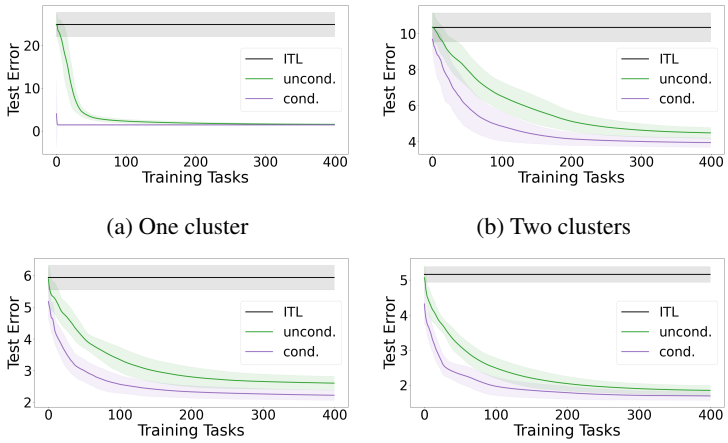
Note that the value function loss and the policy loss of the PPO algorithm are

$$L_{\text{value}}^{PPO}(\phi) = E_t\left[(v_{;c}(s_t) \quad v_{\text{old};c}(s_t))^2\right] + \eta k w k^2,$$

$$L_{\text{policy}}^{PPO}(\theta) = \quad E_t\left[\min\left(\frac{\pi_{;c}(a_t j s_t)}{\pi_{\text{old};c}(a_t j s_t)}\hat{A}_t, \text{clip}(\rho, 1 \quad \epsilon, 1 + \epsilon)\hat{A}_t\right)\right].$$

Algorithm 2: Meta Test

Input: Trained value function $v_{;c}(s)$ and policy $_{;c}$.
Sample task and side information $(M_{test}; c_{test})$ from $p(M; C)$.
Collect episodes by interacting with M_{test} with an equal probability for each initial state and the policy $_{;c}$.
while $n < N$ **do**
  Sample an initial state from the learned task M_{test}.
  Collect an episode in M_{test} with the learned policy $_{;c}$.
  Update $(s; c)$, w and according to the PPO object loss function.
  n = n+1
end
Evaluate the policy $_{;c}$.



(a) One cluster (b) Two clusters

(c) Uneven clusters (d) Extremely uneven clusters

Figure 2: Synthetic experiment results: test errors of independent task learning, unconditional meta-RL and conditional meta-RL with regard to the number of tasks in different environments. The result is averaged over 10 seeds.

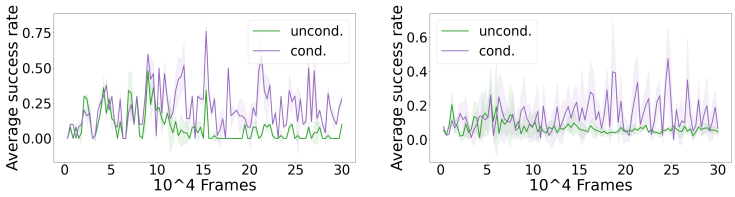
Here $v_{;c}(s_t)$ is the predicted value of state s_t under the value function $v_{;c}(s) = \phi(s,c)^T w$. We modify the value function loss by adding penalty on the weight to satisfy our theory setting. $\pi_{;c}(a_t|s_t)$ is the probability of taking action a_t in state s_t under policy $\pi_{;c}$. The policy parameters are denoted as θ, with θ_{old} representing the old policy parameters used for comparison in the update. $\hat{A}_t$ stands for the advantage estimate at time t, $\hat{A}_t = \delta_t + (\gamma\lambda)\delta_{t+1} + \ldots + (\gamma\lambda)^{T-t+1}\delta_{T-1}$, where δ_t is the temporal difference (TD) error at time t, calculated as: $\delta_t = r_t + \gamma v_{;c}(s_{t+1}) - v_{;c}(s_t)$. Additionally, ρ is the importance sampling ratio, while ϵ is a small constant introduced to clip the ratio, ensuring stability in the optimization process. The clip function $\text{clip}(\rho, 1-\epsilon, 1+\epsilon) = \max(\min(\rho, 1+\epsilon), 1-\epsilon)$ to make sure the ratio is clipped in the interval $[1-\epsilon, 1+\epsilon]$.

6 Experiments

In this section, we compare our conditional meta-RL algorithm Algorithm 1 to the unconditional setting. We address the following question: Does the conditional meta-RL algorithm improves predictive performance comparing to the unconditional counterpart? We will begin by employing a synthetic example to demonstrate our idea. Following that, we will utilize a well-established benchmark in the field of reinforcement learning to further illustrate our approach.

6.1 Synthetic example

In this example, we assume that the tasks come from different clusters, tasks in the same cluster share the same transition kernel. We consider $S = 100$ states and use three types of transition structures: a star graph,a 2D torus and a chain graph. The topology of the graph structures can be



(a) Four Rooms Success Rate (b) Go To Door Success Rate

Figure 3: MiniGrid experiment results: Figures 3a, 3b show success rates of the environments. The results are averaged over 5 seeds.

found in Appendix. For each experiment, we take 560 tasks. For each task, we assume the initial state distribution ν to be uniform and the reward vector r has one element equals 10 and all other elements 0, plus a standard Gaussian error. Then we sample a dataset of n_{tot} pairs of (s_i, y_i) where a Monte Carlo rollout is performed to generate the returns $(y_i)_{i=1}^{n_{tot}}$ and n_{tot} is the number of the samples. We set $\gamma = 0.9$ and use a random policy to select the actions with equal probabilities. The context is a scalar indicating which cluster the task comes from. In other words, context c is given in the following experiments and different feature functions $\phi(s, c)$ are trained for for different clusters. In the training, we use $\phi(s, c) = M_c\psi(s)$ for simplicity and the radial basis for the initial feature.

In the first experiment, we just take one cluster of tasks with a star graph transition structure. We set the number of features to be 1 and $n_{tot} = 20$. The result is shown in Figure 2a. We can see that conditional meta-RL learns faster than unconditional counterpart and they both performs better than independent task learning.

In the second experiment, we take two clusters of MDPs with their transition structures to be 2D torus and chain graphs. We set the number of features to be 10 and $n_{tot} = 200$. We assume that the MDPs come evenly from the two clusters. The result is shown in Figure 2b. The conditional setting outperforms unconditional setting as we expected.

In the third experiment, we want to explore the impact of the distribution of the tasks on the result. We use the same setting as the second experiment, except that the MDPs come from the chain graph cluster with probability 0.9 and the 2D torus with probability 0.1. The result is shown in Figure 2c. The result pattern of this experiment does not change a lot compare to the second experiment, but the test error values decrease significantly.

In the fourth experiment, we want to test further the relationship between the distribution of the tasks and the test error. We use the same setting as the second experiment, and the MDPs come from the chain graph cluster with probability 0.99 and the 2d torus with probability 0.01. The result is shown in Figure 2d.

In our experimental analysis of the last three experiments, we focused on evaluating the performance of conditional and unconditional Meta-RL approaches across three distinct distributions of tasks within two clusters of environments: even, uneven, and extremely uneven distributions. A consistent observation across these experiments was the reduction in test error for the conditional approach as compared to the unconditional approach, with a notable decrease of approximately 14 percent in all three scenarios. This finding suggests that the variability in task distribution—ranging from even to extremely uneven—exerts a minimal influence on the efficacy of the conditional Meta-RL framework in enhancing test performance.

Notably, the magnitude of test error exhibited a declining trend in relation to the unevenness of task distribution, with error rates diminishing from 4 in the even distribution scenario to 2.25 and 1.73 in the uneven and extremely uneven scenarios, respectively. This trend can likely be attributed to the inherent complexity of learning within 2D torus environments. As these environments present significant learning challenges, reducing their prevalence in the task distribution appears to directly contribute to lower overall test errors.

6.2 MiniGrid

MiniGrid is commonly used as benchmark for contextual MDPs. It includes a wide range of grid-based environments with varying levels of complexity, all of which are designed to be simple to understand and work with. Within these environments, the agent is represented by a triangular figure, with a set of discrete actions. The objectives range from navigating through various maze maps to engaging with a variety of items, including doors, keys, and containers [Chevalier-Boisvert et al., 2023]. Here we test our theory on two MiniGrid environments: Four rooms and Go to door. More details are shown in Appendix.

Four rooms. In this environment, there are four interconnected rooms, linked by openings in the walls. The agent is placed randomly and the goal is a green square placed in one of the rooms. To receive a reward, the agent needs to reach the designated green goal square. We assume two clusters of tasks, one with the goal square on the top left corner $(1, 1)$, another with the goal square on the bottom right corner $(17, 17)$. The context here is the position of the goal square.

Go to door. In this setting, there is a room featuring four doors, one on each wall. The four doors are in four different colors, and one of them is the goal door. Upon arriving at and performing the 'done' action next to the goal door, the agent receives a positive reward. We assume four clusters of the tasks depending on the color of the goal doors: red, green, blue and purple. The context here is the color of the goal door.

Table 1: MiniGrid experimental results

Experiment	Unconditional Mean	Conditional Mean
Four Rooms	0.038±0.059	0.105±0.114
Go To Door	0.124±0.122	0.196±0.126

The mean of the return of the conditional and unconditional meta-RL algorithms are shown in Table 1.The results demonstrate a substantial superiority of the conditional mean reward over the unconditional mean reward, providing empirical support for our theoretical framework in MiniGrid environments.

The environments of Four Rooms and Go To Door are notably challenging due to their sparse reward structure, where rewards are only given upon achieving the goal, without any intermediate rewards. Meta-RL with sparse rewards is extremely difficult as it amounts to solving many sparse reward tasks from scratch [Rakelly et al., 2019].The variation in mean rewards across these environments, as observed in our experimental results, highlights this challenge. Following previous meta-RL work on environments with sparse rewards [Zhang et al., 2021, Zintgraf et al., 2021], we also include figures showing the success rate of the two environments in Figuers 3a, 3b. The conditional approach not only demonstrates a higher average success rate but also provides a practical demonstration of its capacity to navigate the complexities inherent in sparse reward environments more effectively than the unconditional approach.

7 Conclusions

This study emphasizes the significant advantages of incorporating contextual information into meta-reinforcement learning (meta-RL). While contextual information is commonly used in meta-RL, there's been a notable lack of statistical analysis on its impact on state representation learning. Our research fills this gap by proposing a theoretical framework to assess the effectiveness of learning state representations in conditional meta-RL. We found that conditional meta-RL, which utilizes contextual information, outperforms traditional methods without such context. We've developed an optimal state representation function to minimize transfer risk in meta-RL settings and introduced a novel algorithm inspired by the PPO algorithm, demonstrating our theoretical insights through synthetic and real-world tests. These experiments validate the importance of context in meta-RL and show that conditional meta-RL yields better performance and efficiency. However, our algorithm only investigates explicitly defined contexts. Future research could explore the potential of employing encoders to generate context.

References

Sanae Amani, Lin Yang, and Ching-An Cheng. Provably efficient lifelong reinforcement learning with linear representation. In *The Eleventh International Conference on Learning Representations*, 2022.

Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shimon Whiteson. A survey of meta-reinforcement learning. *arXiv preprint arXiv:2301.08028*, 2023.

Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International conference on machine learning*, pages 449–458. PMLR, 2017.

Olivier Bousquet and André Elisseeff. Stability and generalization. *The Journal of Machine Learning Research*, 2:499–526, 2002.

Yuan Cheng, Songtao Feng, Jing Yang, Hong Zhang, and Yingbin Liang. Provable benefit of multitask representation learning in reinforcement learning. *arXiv preprint arXiv:2206.05900*, 2022.

Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo de Lazcano, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. *CoRR*, abs/2306.13831, 2023.

Wesley Chung, Somjit Nath, Ajin Joseph, and Martha White. Two-timescale networks for nonlinear value function approximation. In *International conference on learning representations*, 2018.

Giulia Denevi, Massimiliano Pontil, and Carlo Ciliberto. The advantage of conditional meta-learning for biased regularization and fine tuning. *Advances in Neural Information Processing Systems*, 33: 964–974, 2020.

Giulia Denevi, Massimiliano Pontil, and Carlo Ciliberto. Conditional meta-learning of linear representations. *Advances in Neural Information Processing Systems*, 35:253–266, 2022.

Ron Dorfman, Idan Shenfeld, and Aviv Tamar. Offline meta reinforcement learning–identifiability challenges and effective data collection strategies. *Advances in Neural Information Processing Systems*, 34:4607–4618, 2021.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

Haotian Fu, Hongyao Tang, Jianye Hao, Chen Chen, Xidong Feng, Dong Li, and Wulong Liu. Towards effective context for meta-reinforcement learning: an approach based on contrastive learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 7457–7465, 2021.

HAO Jianye, Pengyi Li, Hongyao Tang, Yan Zheng, Xian Fu, and Zhaopeng Meng. Erl-re 2: Efficient evolutionary reinforcement learning with shared state representation and individual policy representation. In *The Eleventh International Conference on Learning Representations*, 2022.

J Zico Kolter and Andrew Y Ng. Regularization and feature selection in least-squares temporal difference learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 521–528, 2009.

George Konidaris, Sarah Osentoski, and Philip Thomas. Value function approximation in reinforcement learning using the fourier basis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25, pages 380–385, 2011.

Charline Le Lan, Stephen Tu, Adam Oberman, Rishabh Agarwal, and Marc G Bellemare. On the generalization of representations in reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pages 4132–4157. PMLR, 2022.

Evan Z Liu, Aditi Raghunathan, Percy Liang, and Chelsea Finn. Decoupling exploration and exploitation for meta-reinforcement learning without sacrifices. In *International conference on machine learning*, pages 6925–6935. PMLR, 2021.

Clare Lyle, Mark Rowland, Georg Ostrovski, and Will Dabney. On the effect of auxiliary tasks on representation dynamics. In *International Conference on Artificial Intelligence and Statistics*, pages 1–9. PMLR, 2021.

Sridhar Mahadevan and Mauro Maggioni. Proto-value functions: A laplacian framework for learning representation and control in markov decision processes. *Journal of Machine Learning Research*, 8(10), 2007.

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

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Ronald Parr, Lihong Li, Gavin Taylor, Christopher Painter-Wakefield, and Michael L Littman. An analysis of linear models, linear value-function approximation, and feature selection for reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, pages 752–759, 2008.

Marek Petrik, Gavin Taylor, Ron Parr, and Shlomo Zilberstein. Feature selection using regularization in approximate linear programs for markov decision processes. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pages 871–878, 2010.

Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pages 5331–5340. PMLR, 2019.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

Lingxiao Wang, Qi Cai, Zhuoran Yang, and Zhaoran Wang. Represent to control partially observed systems: Representation learning with provable sample efficiency. In *The Eleventh International Conference on Learning Representations*, 2022.

Zhongwen Xu, Hado P van Hasselt, and David Silver. Meta-gradient reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

Haoqi Yuan and Zongqing Lu. Robust task representations for offline meta-reinforcement learning via contrastive learning. In *International Conference on Machine Learning*, pages 25747–25759. PMLR, 2022.

Jin Zhang, Jianhao Wang, Hao Hu, Tong Chen, Yingfeng Chen, Changjie Fan, and Chongjie Zhang. Metacure: Meta reinforcement learning with empowerment-driven exploration. In *International Conference on Machine Learning*, pages 12600–12610. PMLR, 2021.

Xingyu Zhou. Strong convexity, 2017. URL `https://xingyuzhou.org/blog/notes/strong-convexity`.

Xingyu Zhou. On the fenchel duality between strong convexity and lipschitz continuous gradient. *arXiv preprint arXiv:1803.06573*, 2018.

Luisa Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. *arXiv preprint arXiv:1910.08348*, 2019.

Luisa M Zintgraf, Leo Feng, Cong Lu, Maximilian Igl, Kristian Hartikainen, Katja Hofmann, and Shimon Whiteson. Exploration in approximate hyper-state space for meta reinforcement learning. In *International Conference on Machine Learning*, pages 12991–13001. PMLR, 2021.

A Generalization bound of the within-task algorithm

The following proposition provides a bound for the average difference between the expected risk and the empirical risk. This proposition shows that the generalization error can be bounded by the square of the sum of the maximum of the approximated value function and the maximum of the true value function $(W_{\max} + \frac{R_{\max}}{1-\gamma})^2$ and the expected square of the feature function of the initial state $\mathbb{E}_s \|\phi(s)\|^2$. We will later use it in Theorem 1.

Proposition 2. For a task $M \sim p(M)$, fix a dataset $Z = (s_i, y_i)_{i=1}^n \sim M^n$. For any $\phi \in F$, let $w(Z)$ be the output in (7) over Z. Then the following generalization bound holds for $w(Z)$:

$$\mathbb{E}_{Z \sim M^n}[R_M(w(Z)) - R_{Z;0}(w(Z))] \leq \frac{8}{n}(W_{\max} + \frac{R_{\max}}{1-\gamma})^2 \mathbb{E}_s \|\phi(s)\|^2$$

Proof. For any $i \in \{1, \ldots, n\}$,consider the dataset $Z^{(i)}$, a copy of the original dataset Z except the i-th input (s_i, y_i) replaced by (s_i', y_i'). now we will show the discrepancy between $w(Z)$ and $w(Z^{(i)})$.

Using Lemma 2 in [Zhou, 2018] (see also [Zhou, 2017]), we know that $R_{Z;1}(w)$ is 1-strongly convex, therefore, we have

$$\frac{1}{2}\|w(Z^{(i)}) - w(Z)\|^2 \leq R_{Z;1}(w(Z^{(i)})) - R_{Z;1}(w(Z))$$

$$\frac{1}{2}\|w(Z^{(i)}) - w(Z)\|^2 \leq R_{Z^{(i)};1}(w(Z)) - R_{Z^{(i)};1}(w(Z^{(i)}))$$

Sum up the two inequalities, we get

$$\begin{aligned} \|w(Z^{(i)}) - w(Z)\|^2 &\leq R_{Z;1}(w(Z^{(i)})) - R_{Z^{(i)};1}(w(Z^{(i)})) \qquad (15) \\ &\quad + R_{Z^{(i)};1}(w(Z)) - R_{Z;1}(w(Z)) \\ &= \frac{A+B}{n} \qquad (16) \end{aligned}$$

where

$$\begin{aligned} A &= l(\langle\phi(s_i'), w(Z)\rangle, y_i') - l(\langle\phi(s_i'), w(Z^{(i)})\rangle, y_i') \\ B &= l(\langle\phi(s_i), w(Z^{(i)})\rangle, y_i) - l(\langle\phi(s_i), w(Z)\rangle, y_i) \\ l(\langle\phi(s), w\rangle, y) &= (\langle\phi(s), w\rangle - y)^2. \end{aligned}$$

Note that the partial derivative

$$\frac{\partial l(\langle\phi(s), w\rangle, y)}{\partial\langle\phi(s), w\rangle} = 2(\langle\phi(s), w\rangle - y) \leq 2(W_{\max} + \frac{R_{\max}}{1-\gamma})$$

This leads to the following

$$\begin{aligned} A &\leq 2(W_{\max} + \frac{R_{\max}}{1-\gamma})|\langle\phi(s_i'), w(Z) - w(Z^{(i)})\rangle| \\ &\leq 2(W_{\max} + \frac{R_{\max}}{1-\gamma})\|\phi(s_i')\|\|w(Z) - w(Z^{(i)})\| \qquad (17) \\ B &\leq 2(W_{\max} + \frac{R_{\max}}{1-\gamma})|\langle\phi(s_i)), w(Z^{(i)}) - w(Z)\rangle| \\ &\leq 2(W_{\max} + \frac{R_{\max}}{1-\gamma})\|\phi(s_i)\|\|w(Z) - w(Z^{(i)})\| \end{aligned}$$

Combine these with (16), we get

$$\|w(Z^{(i)}) - w(Z)\| \leq \frac{2}{n}(W_{\max} + \frac{R_{\max}}{1-\gamma})(\|\phi(s_i')\| + \|\phi(s_i)\|)\|$$

Then bring this into (17), we get

$$l(h\phi(s_i^0), w\ (Z)i, y_i^0)\ \ l(h\phi(s_i^0), w\ (Z^{(i)})i, y_i^0) \quad \frac{4}{n}(W_{\max} + \frac{R_{\max}}{1\quad\gamma})^2 k\phi(s_i^0)k(k\phi(s_i^0)k + k\phi(s_i))k)$$

Now, take expectation with regard to Z and (s_i^0, y_i^0) and apply lemma 7 in [Bousquet and Elisseeff, 2002], we get

$$\mathsf{E}_{Z\quad M^n}\mathsf{E}_{y_i^\theta\ \underset{s_i^\theta}{G}\ (s_i^\theta)}[l(h\phi(s_i^0), w\ (Z)i, y_i^0) \quad l(h\phi(s_i^0), w\ (Z^{(i)})i, y_i^0)] = \mathsf{E}_{Z\quad M^n}[R_M(w\ (Z)) \quad R_{Z;0}(w\ (Z))]$$

Therefore, we get

$$\begin{aligned}\mathsf{E}_{Z\quad M^n}[R_M(w\ (Z)) \quad R_{Z;0}(w\ (Z))] &\quad \frac{4}{n}(W_{\max} + \frac{R_{\max}}{1\quad\gamma})^2 \mathsf{E}_{Z\quad M^n}\mathsf{E}_{y_i^\theta\ \underset{s_i;s_i^\theta}{G}\ (s_i^\theta)}k\phi(s_i^0)k(k\phi(s_i^0)k + k\phi(s_i))k) \\ &\quad \frac{4}{n}(W_{\max} + \frac{R_{\max}}{1\quad\gamma})^2 \mathsf{E}_{Z\quad M^n}\mathsf{E}_{s_i;s_i^\theta}\quad 2k\phi(s_i^0)k^2 \\ &\quad \frac{8}{n}(W_{\max} + \frac{R_{\max}}{1\quad\gamma})^2 \mathsf{E}_{s_i^\theta}\quad k\phi(s_i^0)k^2\end{aligned}$$

□

Lemma. (lemma 7 in [Bousquet and Elisseeff, 2002]) For any symmetric learning algorithm A, we have $8i\ 2\ f1, \ldots, mg$:

$$\mathsf{E}_Z[R(A, Z) \quad R_{\text{emp}}(A, Z)] = \mathsf{E}_{Z;z_i^\theta}[l(A_Z, z_i^0) \quad l(A_{Z^{(i)}}, z_i^0)]$$

.

B Excess risk with conditional feature function

Theorem 1. Let $(e_i)_{i=1}^S \quad \mathsf{R}^S$ be the standard basis and $N\ ,\ \mathsf{E}_i \quad [e_i e_i^T]$. We use $\ _c$ to represent the feature matrix when context information exists. We assume further that $\ _c\ \ _c^y$ is diagonal, then we have

$$\begin{aligned}E_p(\phi, C) \quad E_p \quad \mathsf{E}_{c\ \ p(C)}[&\frac{R_{\max}^2}{2(1\quad\gamma)^2}Tr((\ _c\ \ _c^T)^y M(c)) \\ &+ \frac{8}{n}(W_{\max} + \frac{R_{\max}}{1\quad\gamma})^2 Tr(\ _c\ \ _c^T N(c))],\end{aligned} \tag{18}$$

Proof. For any $(M, c) \quad p(M, C)$, we do the following decomposition:

$$E_p(\phi, C) \quad E_p = \mathsf{E}_{(M;c)\ \ p(M;C)}[B_{M;c} + C_{M;c}].$$

Here,

$$\begin{aligned}B_{M;c} &= \mathsf{E}_{Z\quad M^n}[R_M(A(\phi(s, c), Z)) \quad R_{Z;0}(A(\phi(s, c), Z))] \\ C_{M;c} &= \mathsf{E}_{Z\quad M^n}[R_{Z;0}(A(\phi(s, c), Z)) \quad R_M(w_M)]\end{aligned}$$

From Proposition 2, we know that $B_{M;c} \qquad \frac{8}{n}(W_{\max} + \frac{R_{\max}}{1})^2\mathsf{E}_s \quad k\phi(s, c)k^2 \quad = \frac{8}{n}(W_{\max} + \frac{R_{\max}}{1})^2 Tr(\ _c^T N\ \ _c)$.

As for $C_{M;c}$, we know that

$$
\begin{aligned}
C_{M;c} &= \mathsf{E}_{Z \sim M^n}[\min_{w \in \mathbb{R}^d} R_{Z;1}(w) - R_M(w_M)] \\
&\geq \mathsf{E}_{Z \sim M^n}[R_{Z;1}(w_M) - R_M(w_M)] \\
&= \frac{1}{2}\lambda \|w_M\|^2
\end{aligned}
$$

By the methods of least squares and the property of pseudoinverse, we have $w_M = (\Phi_c^T N \Phi_c)^\dagger \Phi_c^T N V$. Therefore,

$$
\begin{aligned}
C_{M;c} &\geq \frac{1}{2}\lambda \|w_M\|^2 \\
&= \frac{1}{2}\lambda \|(\Phi_c^T N \Phi_c)^\dagger \Phi_c^T N V\|^2 \\
&= \frac{1}{2} Tr((\Phi_c^T N \Phi_c)^\dagger \Phi_c^T N V V^T N \Phi_c (\Phi_c^T N \Phi_c)^\dagger) \\
&= \frac{1}{2} Tr(N \Phi_c (\Phi_c^T N \Phi_c)^\dagger (\Phi_c^T N \Phi_c)^\dagger \Phi_c^T N V V^T)
\end{aligned}
$$

Since both NVV^T and $N \Phi_c (\Phi_c^T N \Phi_c)^\dagger (\Phi_c^T N \Phi_c)^\dagger \Phi_c^T$ are positive semidefinite, we have

$$
\begin{aligned}
C_{M;c} &\geq \frac{1}{2} Tr(N \Phi_c (\Phi_c^T N \Phi_c)^\dagger (\Phi_c^T N \Phi_c)^\dagger \Phi_c^T N V V^T) \\
&\geq \frac{1}{2} Tr(N \Phi_c (\Phi_c^T N \Phi_c)^\dagger (\Phi_c^T N \Phi_c)^\dagger \Phi_c^T) Tr(N V V^T) \\
&= \frac{1}{2} Tr(N \Phi_c (\Phi_c^T N \Phi_c)^\dagger (\Phi_c^T N \Phi_c)^\dagger \Phi_c^T) Tr(V^T N V) \\
&\geq \frac{R_{\max}^2}{2(1-\gamma)^2} Tr((\Phi_c^T N \Phi_c)^\dagger)
\end{aligned}
$$

Assume $A = N^{\frac{1}{2}} \Phi_c$, since $\Phi_c^T \Phi_c$ and N are diagonal, one can check that $A^\dagger = \Phi_c^\dagger (N^\dagger)^{\frac{1}{2}}$. Then because $(A^T A)^\dagger = A^\dagger (A^T)^\dagger$, we get $(\Phi_c^T N \Phi_c)^\dagger = \Phi_c^\dagger N^\dagger (\Phi_c^\dagger)^T$. Therefore,

$$
\begin{aligned}
C_{M;c} &\geq \frac{R_{\max}^2}{2(1-\gamma)^2} Tr(\Phi_c^\dagger N^\dagger (\Phi_c^\dagger)^T) \\
&= \frac{R_{\max}^2}{2(1-\gamma)^2} Tr((\Phi_c^\dagger)^T \Phi_c^\dagger N^\dagger) \\
&= \frac{R_{\max}^2}{2(1-\gamma)^2} Tr((\Phi_c \Phi_c^T)^\dagger N^\dagger).
\end{aligned}
$$

Then we have

$$
\begin{aligned}
E_p(\phi, C) &\geq E_p \geq \mathsf{E}_{(M;c) \sim p(M;C)}[\frac{R_{\max}^2}{2(1-\gamma)^2} Tr((\Phi_c \Phi_c^T)^\dagger N^\dagger) + \frac{8}{n}(W_{\max} + \frac{R_{\max}}{1-\gamma})^2 Tr(\Phi_c \Phi_c^T N)] \\
&= \mathsf{E}_{c \sim p(C)}[\frac{R_{\max}^2}{2(1-\gamma)^2} Tr((\Phi_c \Phi_c^T)^\dagger \mathsf{E}_{M \sim p(M|c)} N^\dagger) + \frac{8}{n}(W_{\max} + \frac{R_{\max}}{1-\gamma})^2 Tr(\Phi_c \Phi_c^T \mathsf{E}_{M \sim p(M|c)} N)]
\end{aligned}
$$

□

Note We want to justify the assumption that $\Phi_c^T \Phi_c$ is diagonal. The feature matrix is $\Phi_c \in \mathbb{R}^{S \times d}$ where S is the number of states and d is the dimension of the representations. Ideally, we want to

find d as small as possible to reduce the computational cost. A simple way of achieving this is to choose linearly independent representations for the d states that appear in the initial distribution, with other representations being vectors with all elements equal to 0 (zero vector). Therefore, we have $_c\ _c^T \in \mathbb{R}^{S\ S}$ as a diagonal matrix with only the first d elements not equal to 0. Therefore, $_c\ _c^y = _c\ _c^T(_c\ _c^T)^y$ is a diagonal matrix.

Proposition 1. (Best conditioning function in hindsight)

The minimizer of the bound in (11) in set F^0 is the feature function $\phi_p(s,c)$ that satisfies the condition

$$_c\ _c^T = \frac{\sqrt{n}R_{\max}N(c)^{y=2}(N(c)^{\frac{1}{2}}M(c)N(c)^{\frac{1}{2}})^{\frac{1}{2}}N(c)^{y=2}}{4(R_{\max} + W_{\max}(1 \quad \gamma))} \tag{19}$$

The according excess risk

$$E_p(\phi_p, C) \quad E_p \quad \sqrt{p}\frac{4R_{\max}}{\sqrt{n}(1 \quad \gamma)}(W_{\max} + \frac{R_{\max}}{1 \quad \gamma})\mathbb{E}_{c \ p(C)}Tr(M(c)N(c))^{\frac{1}{2}}. \tag{20}$$

Proof. The proof follows directly from Proposition 10 in [Denevi et al., 2022] and the fact that $_c\ _c^T$ is positive semidefinite and $N(c)$ and $M(c)$ are diagonal matrices. □

C Graphical structures in synthetic example

In this section, we introduce the topological structures of the three graph structures we mentioned in the synthetic experiment. We first give a detailed textual description of the graphs, followed by showing the topological structures in Figure 4.

A **star graph** is a type of graph in which all vertices are connected to a single central vertex, forming a shape reminiscent of a star. This central vertex is known as the *center* of the star, and the edges connecting the center to the other vertices are known as the *spokes* of the star. The central state is directly connected to $S \quad 1$ other states, making star graphs a special case of a complete bipartite graph $K_{1;S\ 1}$.

The **2D Torus** topology is characterized by a two-dimensional rectangular lattice, arranged in $\sqrt{S}$ rows and $\sqrt{S}$ columns, culminating in a total of S states. This layout ensures that each node is directly connected to its four immediate neighbors, allowing for communication in four principal directions: $+x$, $\quad x$, $+y$, and $\quad y$. A distinctive feature of this topology is the connection of corresponding nodes on opposite edges, significantly enhancing the network's communication capabilities.

A **chain graph** is a type of graph that consists of a sequence of vertices connected by edges, where each state is directly connected to two other states, except for the two pendant states at endpoints. Chain graphs represent linear structures, where the vertices can be thought of as links in a chain.

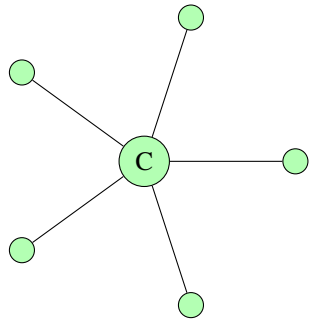
D MiniGrid environment

In this work, we use two environments in MiniGrid: Four Rooms and Go To Door. Specifically, we use 'MiniGrid-FourRooms-v0' and 'MiniGrid-GoToDoor-8x8-v0'.

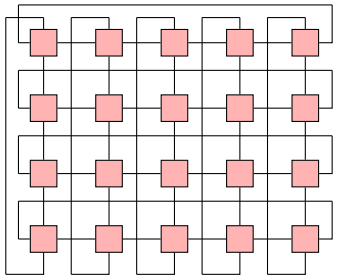
In all tasks, an upper limit of $t_{\max}$ steps is set to motivate the agent to complete the task efficiently. If the agent successfully completes the task in t steps, it receives a reward $r = 1 \quad 0.9\frac{t}{t_{\max}}$ in both environments. The episode is terminated either when the agent obtains the final reward or when $t_{\max}$ is surpassed. Observations are presented as a $7 \quad 7 \quad 3$ grid encoding. The first two dimensions $(7 \quad 7)$ represent the spatial layout, while the third dimension encodes attributes like object type (e.g., wall, door), color (e.g., red, green), and status (e.g., door open, closed, locked). The agent can take seven actions: turn left, turn right, move forward, pick up an object, drop an object, toggle and done. Not all actions are applicable in every task.

E Hyperparameters

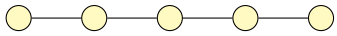
In this section, we list all the hyperparameters we used in learning the MiniGird environments.



(a) Star graph

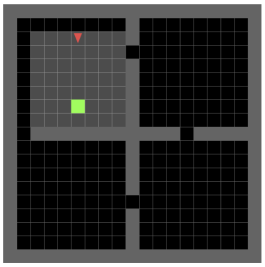


(b) 2D torus

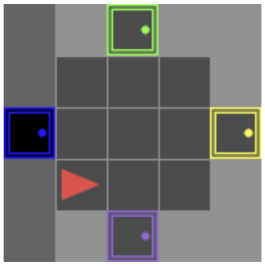


(c) Chain graph

Figure 4: Three types of graphs used in the synthetic experiment.



(a) Four Rooms



(b) Go To Door

Figure 5: MiniGrid environments

Table 2: List of hyperparameters

Parameter	**Value**
Number of frames per iteration	3000
Number of meta-training iteration	100
Number of parallel actors	12
Batch size	256
Discount γ	0.99
Learning rate	0.001
GAE λ	0.95
Value loss coefficient	0.5
Clipping factor PPO	0.2
Maximum norm of gradient	0.5
Penalty parameter η	1e-10