Learning Action Translator for Meta Reinforcement Learning on Sparse-Reward Tasks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Meta reinforcement learning (meta-RL) aims to learn a policy solving a set of training tasks simultaneously and quickly adapting to new tasks. It requires massive amounts of data drawn from training tasks to infer the common structure shared among tasks. Without heavy reward engineering, the sparse rewards in long-horizon tasks exacerbate the problem of sample efficiency in meta-RL. Another challenge in meta-RL is the discrepancy of difficulty level among tasks, which might cause one easy task dominating learning of the shared policy and thus preclude policy adaptation to new tasks. In this work, we introduce a novel objective function to learn an action translator among training tasks. We theoretically verify that value of the transferred policy with the action translator can be close to the value of the source policy. We propose to combine the action translator with context-based meta-RL algorithms for better data collection and more efficient exploration during meta-training. Our approach of policy transfer empirically improves the sample efficiency and performance of meta-RL algorithms on sparse-reward tasks.

1 Introduction

Deep reinforcement learning (DRL) methods achieved remarkable success in solving complex tasks[15, 26, 24]. While conventional DRL methods learn an individual policy for each task, meta reinforcement learning (meta-RL) algorithms [7, 4, 14] learn the shared structure across a distribution of tasks so that the agent can quickly adapt to unseen related tasks in the test phase. Unlike most of the existing meta-RL approaches working on tasks with dense rewards, we instead focus on the sparse-reward training tasks which are more common in real-world scenarios without access to carefully designed reward functions in the environments. Recent works in meta-RL propose off-policy algorithms [21, 5] and model-based algorithms [17, 16, 12, 25] to improve the sample efficiency in meta-training procedures. However, it still remains challenging to efficiently solve multiple tasks that require reasoning over long horizons with sparse rewards. In these tasks, the scarcity of positive rewards exacerbates the issue of sample efficiency which plagues meta-RL algorithms and makes exploration difficult due to lack of guidance signals.

Intuitively, we hope that solving one task facilitates learning of other related tasks since the training tasks share a common structure. However, it is often not the case in practice [23, 19]. Previous works [30, 36] point out that detrimental gradient interference might cause an imbalance in policy learning on multiple tasks. Policy distillation[30] and gradient projection[36] are developed in meta-RL algorithms to alleviate this issue. However, in our sparse-reward setting, this issue might become more severe because it is hard to explore each task to obtain meaningful gradient signals for policy updates. Good performance in one task does not automatically help exploration on the other tasks since the agent lacks positive rewards on the other tasks to learn from.

Submitted to NeurIPS 2021 Deep Reinforcement Learning workshop. Do not distribute.

In this work, we aim to fully exploit the highly-rewarding transitions occasionally discovered by the agent in the exploration. The good experiences in one task should not only improve the policy on this task but also benefit the policy on other tasks to drive deeper exploration.

Specifically, once the agent learns from the successful trajectories in one training task, we transfer the good policy in this task to other tasks to get more positive rewards on other training tasks. In Fig. 1, if the learned policy π performs better on task $\mathcal{T}^{(2)}$ than other tasks, then our goal is to transfer the good policy $\pi(\cdot, \mathcal{T}^{(2)})$ to other tasks $\mathcal{T}^{(1)}$ and $\mathcal{T}^{(3)}$. To enable such transfer, we propose to learn an action translator among multiple training tasks. The objective function forces the translated action to behave on the target task similarly to the source action on the source task. We theoretically show that the transferred policy with this action translator can achieve a value on the target task close to the value of the source policy on the source task. We consider the policy transfer for any pair of source and target tasks in the training task distribution (see the colored arrows in Fig. 1). The agent executes actions following the transferred policy if the transferred policy attains higher rewards than the learned policy on the target task in recent episodes. This approach enables the agent to leverage relevant data from multiple training tasks, encourages the learned policy to perform similarly well on multiple training tasks, and thus leads to better performance when applying the well-trained policy to test tasks.

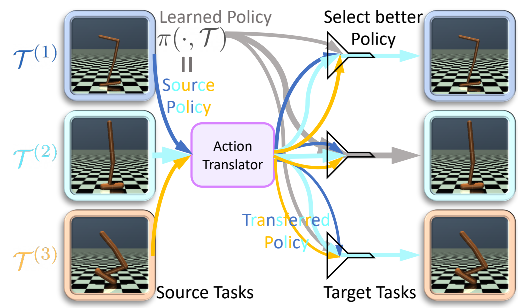


Figure 1: Illustration of our policy transfer. Size of arrows represents avg. episode reward of learned or transferred policy on target tasks. Different colors indicate different tasks.

We summarize our contributions: (1) We introduce a novel objective function to transfer any policy from a source Markov Decision Process (MDP) to a target MDP. We prove a theoretical guarantee that the transferred policy can achieve expected return on the target MDP close to the source policy on the source MDP, where the difference in expected return is (approximately) upper bounded by our loss function with a constant multiplicative factor. (2) We develop an off-policy RL algorithm called **M**eta-RL with **C**ontext-conditioned **A**ction **T**ranslator (MCAT), applying a policy transfer mechanism in meta-RL to help exploration across multiple sparse-rewards tasks. (3) We empirically demonstrate the effectiveness of MCAT on a variety of simulated control tasks with MuJoCo physics engine[31], showing that policy transfer improves the performance of context-based meta-RL algorithms.

2 Method

In this section, we first describe our approach to learn a context encoder capturing the task features and learn a forward dynamics model predicting next state distribution given the task context (Sec. 2.2). Then we introduce an objective function to train an action translator so that the translated action on the target task behaves equivalently to the source action on the source task. The action translator can be conditioned on the task contexts and thus it can transfer a good policy from any arbitrary source task to any other target task in the training set (Sec. 2.3). Finally, we propose to combine the action translator with a context-based meta-RL algorithm to transfer the good policy from any one task to the others. During meta-training, this policy transfer approach helps exploit the good experiences encountered on any one task and benefits the data collection and further policy optimization on other sparse-reward tasks (Sec. 2.4). Fig. 2 provides an overview of our approach MCAT.

2.1 Problem Formulation

Following meta-RL formulation in previous work [4, 14, 21], we assume a distribution of tasks $p(\mathcal{T})$ and each task is a Markov decision process (MDP) defined as a tuple $(\mathcal{S}, \mathcal{A}, p, r, \gamma, \rho_0)$ with state space $\mathcal{S}$, action space $\mathcal{A}$, transition function $p(s'|s, a)$, reward function $r(s, a, s')$, discounting factor γ, and initial state distribution ρ_0. We can alternatively define the reward function as $r(s, a) = \sum_{s' \in \mathcal{S}} p(s'|s, a) r(s, a, s')$. In context-based meta-RL algorithms, we learn a policy $\pi(\cdot|s_t^{(i)}, z_t^{(i)})$ shared for any task $\mathcal{T}^{(i)} \sim p(\mathcal{T})$, where t denotes the timestep in an episode, i denotes the index of a task, the context variable $z_t^{(i)} \in \mathcal{Z}$ captures contextual information on the task MDP

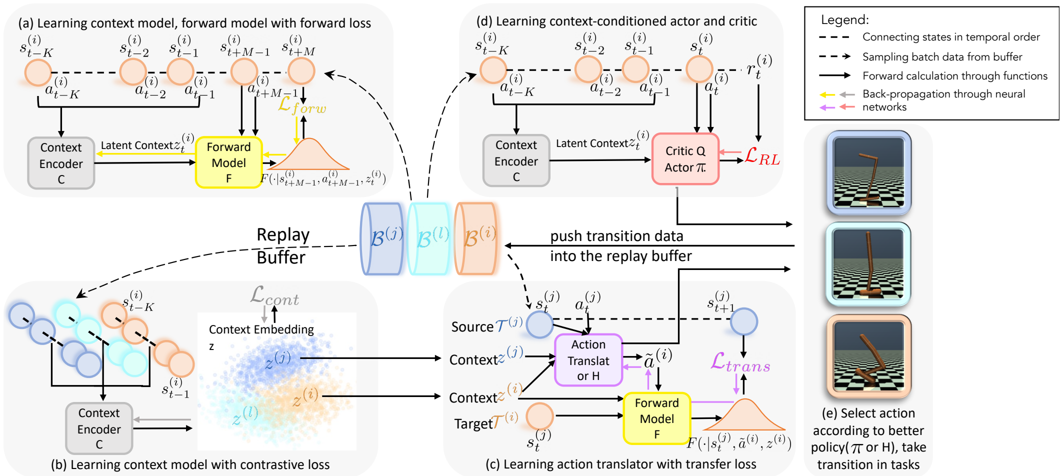


Figure 2: Overview of MCAT. (a) We use forward dynamics prediction loss to train the context encoder C and forward model F. (b) We regularize the context encoder C with the contrastive loss, so context vectors of transition segments from the same task cluster together. (c) With fixed C and F, we learn the action translator H for any pair of source task $\mathcal{T}^{(j)}$ and target task $\mathcal{T}^{(i)}$. The action translator aims to generate action $\tilde{a}^{(i)}$ on the target task leading to the same next state $s_{t+1}^{(j)}$ as the source action $a_t^{(j)}$ on the source task. (d) With fixed C, we learn the critic Q and actor π conditioning on the context feature. We remark that these components C, F, H, Q, π are trained alternatively not jointly and this fact facilitates the learning process. (e) If the agent is interacting with the environment on task $\mathcal{T}^{(i)}$, we compare learned policy $\pi(s, z^{(i)})$ and transferred policy $H(s, \pi(s, z^{(j)}), z^{(j)}, z^{(i)})$, which transfers a good policy $\pi(s, z^{(j)})$ on source task $\mathcal{T}^{(j)}$ to target task $\mathcal{T}^{(i)}$. We select actions according to the policy with higher average episode rewards in the recent episodes. Transition data are pushed into the buffer.

and $\mathcal{Z}$ is the space of context vectors. The context variable is inferred from the history transitions on task $\mathcal{T}^{(i)}$. The shared policy is optimized to maximize its value $V^\pi(\mathcal{T}^{(i)}) = \mathbb{E}_{\rho_0^{(i)},\pi,p^{(i)}}[\sum_{t=0}^{\infty} \gamma^t r_t^{(i)}]$ on each training task $\mathcal{T}^{(i)}$. Following prior works in meta-RL [38, 16, 17, 42, 12], we study tasks with the same state space, action space, reward function but varying dynamics functions. Importantly, we focus on more challenging setting of sparse rewards. Our goal is to learn a shared policy robust to the dynamic changes and generalizable to unseen tasks.

2.2 Learning Context & Forward Model

In order to capture the knowledge about any task $\mathcal{T}^{(i)}$, we leverage a context encoder $C : \mathcal{S}^K \times \mathcal{A}^K \to \mathcal{Z}$, where K is the number of past steps used to infer the context. Related ideas have been explored by [21, 42, 12]. In Fig. 2a, given K past transitions $(s_{t-K}^{(i)}, a_{t-K}^{(i)}, \cdots, s_{t-1}^{(i)}, a_{t-1}^{(i)})$, context encoder C produces the latent context $z_t^{(i)} = C(s_{t-K}^{(i)}, a_{t-K}^{(i)}, \cdots, s_{t-2}^{(i)}, a_{t-2}^{(i)}, s_{t-1}^{(i)}, a_{t-1}^{(i)})$. We train the context encoder C and forward dynamics F with an objective function to predict the forward dynamics in future transitions $s_{t+m}^{(i)}$ $(1 \leq m \leq M)$ within M future steps. The state prediction in multiple future steps drives latent context embeddings $z_t^{(i)}$ to be temporally consistent. The learned context encoder tends to capture dynamics-specific, contextual information (e.g. environment physics parameters). Formally, we minimize the negative log-likelihood of observing the future states under dynamics prediction.

$$\mathcal{L}_{forw} = -\sum_{m=1}^{M} \log F(s_{t+m}^{(i)} | s_{t+m-1}^{(i)}, a_{t+m-1}^{(i)}, z_t^{(i)}). \tag{1}$$

Additionally, given trajectory segments from the same task, we require their context embeddings to be similar, whereas the contexts of history transitions from different tasks should be distinct (Fig. 2b). We propose a contrastive loss [10] to constrain embeddings within a small distance for positive pairs (i.e. samples from the same task) and push embeddings apart with a distance greater than a margin value m for negative pairs (i.e. samples from different tasks). $z_{t_1}^{(i)}$, $z_{t_2}^{(j)}$ denote context embeddings of two trajectory samples from $\mathcal{T}^{(i)}$, $\mathcal{T}^{(j)}$. The contrastive loss function is defined as:

$$\mathcal{L}_{cont} = \mathbb{1}_{i=j}\|z_{t_1}^{(i)} - z_{t_2}^{(j)}\|^2 + \mathbb{1}_{i\neq j}\max(0, m - \|z_{t_1}^{(i)} - z_{t_2}^{(j)}\|) \tag{2}$$

where $\mathbb{1}$ is indicator function. During meta-training, recent transitions on each task $\mathcal{T}^{(i)}$ are stored in a buffer $\mathcal{B}^{(i)}$ for off-policy learning. We randomly sample a fairly large batch of trajectory segments from $\mathcal{B}^{(i)}$, and average their context embeddings to output task feature $z^{(i)}$. $z^{(i)}$ is representative for embeddings on task $\mathcal{T}^{(i)}$ and distinctive from features $z^{(l)}$ and $z^{(j)}$ for other tasks. We note the learned embedding maintains the similarity across tasks. $z^{(i)}$ is closer to $z^{(l)}$ than to $z^{(j)}$ if task $\mathcal{T}^{(i)}$ is more akin to $\mathcal{T}^{(l)}$. We utilize task features for action translation across multiple tasks. Appendix D.5 presents the effect of this auxiliary loss $\mathcal{L}_{cont}$.

2.3 Learning Action Translator

Suppose that transition data $s_t^{(j)}, a_t^{(j)}, s_{t+1}^{(j)}$ behave well on task $\mathcal{T}^{(j)}$. We aim to learn an action translator $H : \mathcal{S} \times \mathcal{A} \times \mathcal{Z} \times \mathcal{Z} \to \mathcal{A}$. $\tilde{a}^{(i)} = H(s_t^{(j)}, a_t^{(j)}, z^{(j)}, z^{(i)})$ translates the proper action $a_t^{(j)}$ from source task $\mathcal{T}^{(j)}$ to target task $\mathcal{T}^{(i)}$. In Fig. 2c, if we start from the same state $s_t^{(j)}$ on both source and target tasks, the translated action $\tilde{a}^{(i)}$ on target task should behave equivalently to the source action $a_t^{(j)}$ on the source task. Thus, the next state $s_{t+1}^{(i)} \sim p^{(i)}(s_t^{(j)}, \tilde{a}^{(i)})$ produced from the transferred action $\tilde{a}^{(i)}$ on the target task should be close to the real next state $s_{t+1}^{(j)}$ gathered on the source task. The objective function of training the action translator H is to maximize the probability of getting next state $s_{t+1}^{(j)}$ under the next state distribution $s_{t+1}^{(i)} \sim p^{(i)}(s_t^{(j)}, \tilde{a}^{(i)})$ on the target task. Because the transition function $p^{(i)}(s_t^{(j)}, \tilde{a}^{(i)})$ is unavailable and might be not differentiable, we use the forward dynamics model $F(\cdot|s_t^{(j)}, \tilde{a}^{(i)}, z^{(i)})$ to approximate the transition function. We formulate objective function for action translator H as:

$$\mathcal{L}_{trans} = -\log F(s_{t+1}^{(j)}|s_t^{(j)}, \tilde{a}^{(i)}, z^{(i)}) \tag{3}$$

where $\tilde{a}^{(i)} = H(s_t^{(j)}, a_t^{(j)}, z^{(j)}, z^{(i)})$. We assume to start from the same initial state, the action translator is to find the action on the target task so as to reach the same next state as the source action on the source task. This intuition to learn the action translator is analogous to learn inverse dynamic model across two tasks.

With a well-trained action translator conditioning on task features $z^{(j)}$ and $z^{(i)}$, we transfer the good deterministic policy $\pi(s, z^{(j)})$ from any source task $\mathcal{T}^{(j)}$ to any target task $\mathcal{T}^{(i)}$. When encountering a state $s^{(i)}$ on $\mathcal{T}^{(i)}$, we query a good action $a^{(j)} = \pi(s^{(i)}, z^{(j)})$ which will lead to a satisfactory next state with high return on the source task. Then H translates this good action $a^{(j)}$ on the source task to action $\tilde{a}^{(i)} = H(s^{(i)}, a^{(j)}, z^{(j)}, z^{(i)})$ on the target task. Executing the translated action $\tilde{a}^{(i)}$ moves the agent to a next state on the target task similarly to the good action on the source task. Therefore, transferred policy $H(s^{(i)}, \pi(s^{(i)}, z^{(j)}), z^{(i)}, z^{(j)})$ can behave similarly to source policy $\pi(s, z^{(j)})$. Sec. 5.1 demonstrates the performance of transferred policy in a variety of environments. Our policy transfer mechanism is related to the action correspondence discussed in [41]. We extend their policy transfer approach across two domains to multiple domains(tasks) and theoretically validate learning of action translator in Sec. 3.

2.4 Combining with Context-based Meta-RL

MCAT follows standard off-policy meta-RL algorithms to learn a deterministic policy $\pi(s_t, z_t^{(i)})$ and a value function $Q(s_t, a_t, z_t^{(i)})$, conditioning on the latent task context variable $z_t^{(i)}$. In the meta-training process, using data sampled from $\mathcal{B}$, we train the context model C and dynamics model F with $\mathcal{L}_{forw}$ and $\mathcal{L}_{cont}$ to accurately predict the next state (Fig. 2a 2b). With the fixed context encoder C and dynamics model F, the action translator H is optimized to minimize $\mathcal{L}_{trans}$ (Fig. 2c). Then, with the fixed C, we train the context-conditioned policy π and value function Q according to $\mathcal{L}_{RL}$ (Fig. 2d). In experiments, we use the objective function $\mathcal{L}_{RL}$ from TD3 algorithm [8]. On sparse-reward tasks where exploration is challenging, the agent might luckily find transitions with high rewards on one task $\mathcal{T}^{(j)}$, and hence the policy learning on this task might be easier than other tasks. If the learned policy π performs better on one task $\mathcal{T}^{(j)}$ than another task $\mathcal{T}^{(i)}$, we consider the policy transferred from $\mathcal{T}^{(j)}$ to $\mathcal{T}^{(i)}$. At a state $s^{(i)}$, we employ the action translator to get a potentially good action $H(s^{(i)}, \pi(s^{(i)}, z^{(j)}), z^{(j)}, z^{(i)})$ on target task $\mathcal{T}^{(i)}$. As illustrated in Fig. 2e and Fig. 1, in the recent episodes, if the transferred policy earns higher scores than the learned policy $\pi(s^{(i)}, z^{(i)})$ on the target task, we follow the translated actions on target task $\mathcal{T}^{(i)}$ to gather transition data in the current episode. These data with better returns are pushed into the replay buffer $\mathcal{B}^{(i)}$ and produce more positive signals for policy learning in the sparse-reward setting. These transition samples help enhance the quality of π on $\mathcal{T}^{(i)}$ after policy update with off-policy RL algorithms. As described in Sec. 2.3, our action translator H allows policy transfer across any pair of tasks. Therefore, with the policy transfer mechanism, the learned policy on each task might benefit from good experiences and policies on any other tasks. See pseudo-code of MCAT in Appendix B.

3 Theoretical Analysis

In this section, we theoretically support our objective function (Equation 3) to learn the action translator. Given s on two MDPs with the same state and action space, we define that action $a^{(i)}$ on $\mathcal{T}^{(i)}$ is equivalent to action $a^{(j)}$ on $\mathcal{T}^{(j)}$ if the actions yielding exactly the same next state distribution and reward, i.e. $p^{(i)}(\cdot|s, a^{(i)}) = p^{(j)}(\cdot|s, a^{(j)})$ and $r^{(i)}(s, a^{(i)}) = r^{(j)}(s, a^{(j)})$. Ideally, the equivalent action always exists on the target MDP $\mathcal{T}^{(i)}$ for any state-action pair on the source MDP $\mathcal{T}^{(j)}$ and there exists an action translator function $H : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{A}$ to identify the exact equivalent action. Starting from state s, the translated action $\tilde{a} = H(s, a)$ on the task $\mathcal{T}^{(i)}$ generates reward and next state distribution the same as action a on the task $\mathcal{T}^{(j)}$ (i.e. $\tilde{a}B_s a$). Then any deterministic policy $\pi^{(j)}$ on the source task $\mathcal{T}^{(j)}$ can be perfectly transferred to the target task $\mathcal{T}^{(i)}$ with $\pi^{(i)}(s) = H(s, \pi^{(j)}(s))$. The value of the policy $\pi^{(j)}$ on the source task $\mathcal{T}^{(j)}$ is equal to the value of transferred policy $\pi^{(i)}$ on the target task $\mathcal{T}^{(i)}$.

Without the assumption of existence of a perfect correspondence for each action, given two deterministic policies $\pi^{(j)}$ on $\mathcal{T}^{(j)}$ and $\pi^{(i)}$ on $\mathcal{T}^{(i)}$, we prove that the difference in policy value is upper bounded by a scalar $\frac{d}{1-\gamma}$ depending on L1-distance between reward functions $|r^{(i)}(s, \pi^{(i)}(s)) - r^{(j)}(s, \pi^{(j)}(s))|$ and total-variation distance between next state distributions $D_{TV}(p^{(i)}(\cdot|s, \pi^{(i)}(s)), p^{(j)}(\cdot|s, \pi^{(j)}(s)))$. Theory (Theorem 1) and proof are in Appendix A.

For a special case where reward function $r(s, a, s')$ only depends on the current state s and next state s', the upper bound of policy value difference is only related to the distance in next state distributions.

Proposition 1. *Let $\mathcal{T}^{(i)} = \{\mathcal{S}, \mathcal{A}, p^{(i)}, r^{(i)}, \gamma, \rho_0\}$ and $\mathcal{T}^{(j)} = \{\mathcal{S}, \mathcal{A}, p^{(j)}, r^{(j)}, \gamma, \rho_0\}$ be two MDPs sampled from the distribution of tasks $p(\mathcal{T})$. $\pi^{(i)}$ is a deterministic policy on $\mathcal{T}^{(i)}$ and $\pi^{(j)}$ is a deterministic policy on $\mathcal{T}^{(j)}$. Assume the reward function only depends on the state and next state $r^{(i)}(s, a^{(i)}, s') = r^{(j)}(s, a^{(j)}, s') = r(s, s')$. Let $M = \sup_{s\in\mathcal{S}, s'\in\mathcal{S}} |r(s, s') + \gamma V^{\pi^{(i)}}(s', \mathcal{T}^{(i)})|$ and $d = \sup_{s\in\mathcal{S}} 2M D_{TV}(p^{(i)}(\cdot|s, \pi^{(i)}(s)), p^{(j)}(\cdot|s, \pi^{(j)}(s)))$. $\forall s \in \mathcal{S}$, we have*

$$\left|V^{\pi^{(i)}}(s, \mathcal{T}^{(i)}) - V^{\pi^{(j)}}(s, \mathcal{T}^{(j)})\right| \leq \frac{d}{1-\gamma} \quad (4)$$

According to Proposition 1, if we can optimize action translator H to minimize d for policy $\pi^{(j)}$ and $\pi^{(i)}(s) = H(s, \pi^{(j)}(s))$, the value of the transferred policy $\pi^{(i)}$ on the target task can be close to the value of source policy $\pi^{(j)}$. In many real-world scenarios, especially sparse-reward tasks, the reward heavily depends on the state and next state instead of action. For example, robots running forward receive rewards according to their velocity (i.e. the location difference between the current and next state within one step); robot arms manipulating various objects earn positive rewards only when they are in the target positions. Thus, our approach focuses on the cases with reward functions approximately as $r(s, s')$ under the assumption of Proposition 1. For any state $s \in \mathcal{S}$, we minimize the total-variation distance between two next state distributions $D_{TV}(p^{(i)}(\cdot|s_t, \pi^{(i)}(s_t)), p^{(j)}(\cdot|s_t, \pi^{(j)}(s_t)))$. Besides, we discuss the policy transfer for tasks with a general reward function in Appendix C.3.

In practice, we approximate the unknown transition function $p^{(i)}(\cdot|s_t, \pi^{(i)}(s_t))$ on target MDP with a forward model $F(\cdot|s_t, \pi^{(i)}(s_t))$, which assumes a Gaussian distribution for next state prediction. There is no closed-form solution of D_{TV} between two Gaussian distributions and D_{TV} is related with Kullback–Leibler (KL) divergence D_{KL} by the inequality $D_{TV}(p\|q)^2 \leq D_{KL}(p\|q)$ [20]. Thus, we instead consider minimizing D_{KL} between two next state distributions. With real data of next states $s^{(j)}_{t+1}$ drawn from $p^{(j)}(\cdot|s_t, \pi^{(j)}(s_t))$ on the source MDP, we further convert D_{KL} between two Gaussian distributions to $\mathcal{L}_{trans} = -\log F(s^{(j)}_{t+1}|s_t, \pi^{(i)}(s_t))$, i.e. the negative log-likelihood of observing the next state $s_{t+1}^{(j)}$ under the approximated distribution $F(\cdot|s_t, \pi^{(i)}(s_t))$. Experiments in Sec. 5.1 suggest that this objective function works well for policy transfer across two MDPs. Sec. 2.3 explains the motivation behind $\mathcal{L}_{trans}$ (Equation 3) to learn an action translator among multiple MDPs instead of only two MDPs.

4 Related Work

Context-based Meta-RL Meta reinforcement learning has been extensively studied in the literature [7, 27, 28, 34] with many works developing the context-based approaches [21, 22, 13]. Duan et al. [4], Wang et al. [33], Fakoor et al. [5] employ recurrent neural networks to encode context transitions and formulate the policy conditioning on the context variables. The objective function of maximizing expected return trains the context encoder and policy jointly. Rakelly et al. [21] leverage a permutation-invariant encoder to aggregate experiences as probabilistic context variables and optimizes it with variational inference. The posterior sampling is beneficial for exploration on sparse-reward tasks in the adaptation phase, but there is access to dense rewards during training phase. Lee et al. [12], Seo et al. [25] trains the context encoder with forward dynamics prediction. These model-based meta-RL algorithms assume the reward function is accessible for planning. In the sparse-reward setting without ground-truth reward functions, they may struggle to discover non-zero rewards and accurately estimating the reward for model-based planning may be problematic as well.

Policy Transfer in RL Policy transfer studies the knowledge transfer in target tasks given a set of source tasks and their expert policies. Policy distillation algorithms [23, 35, 19] minimize the divergence of action distributions between the source policy and the learned policy on the target task. Along this line of works, Teh et al. [30] create a centroid policy in multi-task reinforcement learning and distills the knowledge from the task-specific policies to this centroid policy. Alternatively, inter-task mapping between the source and target tasks [43] can assist the policy transfer. Most of these works [9, 11, 2] assume existence of correspondence over the state space and learn the state mapping between tasks. Recent work [41] learns the state correspondence as well as action correspondence with dynamic cycle-consistency loss. Our method differs from this approach, in that we enable action translation among multiple tasks with a simpler objective function and learn the forward dynamics model with more advanced techniques. Importantly, our approach is novel to utilize the policy transfer for any pair of source and target tasks in meta-RL algorithms.

Bisimulation for States in MDPs Recent works on state representation learning [6, 39, 1] investigate the bismilarity metrics for states on MDPs and consider how to learn a representation for states leading to almost identical behaviors under the same action in diverse MDPs. In multi-task reinforcement learning and meta reinforcement learning problems, Zhang et al. [39, 40] derives transfer and generalization bounds based on the task and state similarity. We bound the value of policy transfer across tasks and our approach is to establish action equivalence instead of state equivalence.

5 Experiment

We design and conduct experiments to answer the following questions:

- Does the transferred policy perform well on the target task (Tab. 1, Fig. 3)?
- Can we transfer the good policy for any pair of source and target tasks (Fig. 4)?
- Does policy transfer improve context-based Meta-RL algorithms (Fig. 5, Tab. 2, Tab. 3)?
- Is the policy transfer more beneficial when the training tasks have sparser rewards (Tab. 4)?

Experimental details can be found in Appendix C.

5.1 Policy Transfer with Fixed Dataset

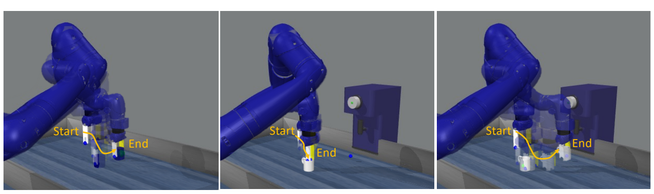
We test our proposed action translator with fixed datasets of transitions aggregated from pairs of source and target tasks. On MuJoCo environments HalfCheetah and Ant, we create tasks with varying dynamics as in [42, 12, 41]. We keep default physics parameters in source tasks and modify them to yield noticeable changes in the dynamics for target tasks. On HalfCheetah, the tasks differ in the armature. On Ant, we set different legs crippled. A well-performing policy is pre-trained on the source task with TD3 algorithm [8] and dense rewards. We then gather training data with

Setting	Source policy	Transferred policy [41]	Transferred policy (Ours)
HalfCheetah	2355.0	**3017.1**(±44.2)	2937.2(±9.5)
Ant	55.8	97.2(±2.5)	**208.1**(±8.2)
Cylinder-Mug	0.0	308.1(±75.3)	**395.6**(±19.4)
Cylinder-Cube	0.0	262.4(±48.1)	**446.1**(±1.1)

Table 1: Performance of source and transferred policy on target task over 3 runs.

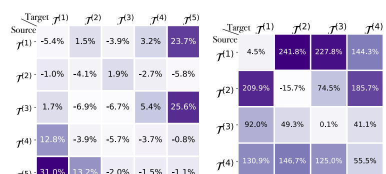
mediocre policies on the source and target tasks. We also include object manipulation tasks on MetaWorld benchmark [37]. Operating objects with varied physics properties requires the agent to handle different dynamics. The knowledge in grasping and pushing a cylinder might be transferrable to tasks of moving a coffee mug or a cube. The agent gets a reward of 1.0 if the object is in the goal location. Otherwise, the reward is 0. We use the manually-designed good policy as the source policy and collect transition data by adding noise to the action drawn from the good policy.

As presented in Tab. 1, directly applying a good source policy on the target task performs poorly with low episode rewards. We learn dynamics model F on target task with $\mathcal{L}_{forw}$ and action translator H with $\mathcal{L}_{trans}$. From a single source task to a single target task, the transferred policy with our action translator (without conditioning on the task context) yields episode rewards significantly better than the source policy on the target task. Fig. 3 visualizes moving paths of robot arms. The transferred policy on target task resembles the source policy on source task, while the source policy has trouble grasping the coffee mug on target task. Videos of agents' behavior are on the webpage [1]. Tab. 1 reports experimental results of baseline [41] transferring the source policy based on action correspondence. It proposes to learn an action translator with three loss terms: adversarial loss, domain cycle-consistency loss, and dynamic cycle-consistency loss. Our loss $\mathcal{L}_{trans}$ (Equation 3) draws upon an idea analogous to dynamic cycle-consistency though we have a more expressive forward model F with context variables. When F is strong and reasonably generalizable, domain cycle-consistency loss training the inverse action translator and adversarial loss constraining the distribution of translated action may not be necessary. Ours with a simpler objective function is competitive with Zhang et al. [41].



(a) Source policy on source task (b) Source policy on target task (c) Transferred policy on target

Figure 3: Moving paths of robot hand according to policies on source task (*soccer* shooting a goal) or target task (*push* a cylinder to a goal).



(a) HalfCheetah (b) Ant

Figure 4: Improvement of transferred policy over source policy on target tasks.

We extend the action translator to multiple tasks by conditioning H on context variables of source and target tasks. We measure the improvement of our transferred policy over the source policy on the target tasks. On HalfCheetah tasks $\mathcal{T}^{(1)} \cdots \mathcal{T}^{(5)}$, the armature becomes larger. As the physics parameter in the target task deviates more from source task, the advantage of transferred policy tends to be more significant (Fig. 4a), because the performance of transferred policy does not drop as much as source policy. We remark that the unified action translator is for any pair of source-target tasks. So improvements at the diagonal elements might be less than 0%. For each task on Ant (Fig. 4b), we set one of its four legs crippled, so any action applied to the crippled leg joints is set as 0. Ideal equivalent action does not always exist across tasks with different crippled legs in this setting. Therefore, it is impossible to minimize d in Proposition 1 as 0. Nevertheless, the inequality proved in Proposition 1 still holds and policy transfer empirically shows positive improvement on most source-target pairs.

5.2 Comparison with Context-based Meta-RL

We evaluate MCAT combining policy transfer with context-based TD3 in meta-RL problems. The action translator is trained dynamically with data maintained in replay buffer and the source policy

[1] videos: https://sites.google.com/view/policy-transfer-meta-rl

Setting	Hopper Size	HalfCheetah Armature	HalfCheetah Mass	Ant Damping	Ant Cripple
MQL	1607.5(±327.5)	-77.9(±214.2)	-413.9(±11.0)	103.1(±35.7)	38.2(±4.0)
PEARL	1755.8(±115.3)	-18.8(±69.3)	25.9(±69.2)	73.2(±13.3)	3.5(±2.4)
Distral	1319.8(±162.2)	566.9(±246.7)	-29.5(±3.0)	90.5(±28.4)	-0.1(±0.7)
HiP-BMDP	1368.3(±150.7)	-102.4(±24.9)	-74.8(±35.4)	33.1(±6.0)	7.3(±2.6)
MCAT(Ours)	**1914.8**(±373.2)	**2071.5**(±447.4)	**1771.1**(±617.7)	**624.6**(±218.8)	**281.6**(±65.6)

Table 2: Episode rewards on test tasks at 2M timesteps.

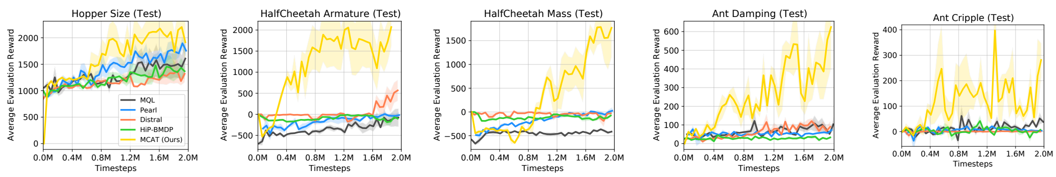


Figure 5: Learning curves of test rewards, averaged over 3 runs. Shadow areas indicate standard error.

keeps being updated. On MuJoCo, we modify environment physics parameters (e.g. size, mass, damping) that affect the transition dynamics to design tasks. We predefine a fixed set of physics parameters for training tasks and unseen test tasks. In order to test algorithms' ability in tackling difficult tasks, environment rewards are delayed to create sparse-reward RL problems [18, 29]. In particular, we accumulate dense rewards over n consecutive steps, and the agent receives the delayed feedback every n step or when the episode terminates. To fully exploit the good data collected from our transferred policy, we empirically incorporate self-imitation learning [18], which imitates the agent's own successful past experiences to further improve the policy learning in the sparse-reward setting. We additionally analyze its effect in Appendix D.4. We compare with several context-based meta-RL methods: MQL [5], PEARL [21], Distral [30], and HiP-BMDP [40]. We run experiments of these baselines using official implementations which are publicly available. Although the baselines perform well on MuJoCo environments with dense rewards, the delayed environment rewards degrade policy learning (Tab. 2, Fig. 5) because the rare transitions with positive rewards are not fully exploited. In contrast, MCAT shows a substantial advantage in performance and sample complexity on both the training tasks and the test tasks. Notably, the performance gap is more significant in more complex environments (e.g. HalfCheetah and Ant with higher-dimensional state and sparser rewards).

5.3 Ablative Study

Effect of Policy Transfer Our MCAT is implemented by combining context-based TD3, self-imitation learning, and policy transfer (PT). We investigate the effect of policy transfer. In Tab. 3. MCAT significantly outperforms MCAT w/o PT, because PT facilitates more balanced performance across training tasks and hence better generalization to test tasks. This empirically confirms that policy transfer is beneficial in meta-RL on sparse-reward tasks.

Setting	Hopper Size	HalfCheetah Armature	HalfCheetah Mass	Ant Damping	Ant Cripple
MCAT w/o PT	1497.5(±282.8)	579.1(±527.1)	-364.3(±198.5)	187.7(±44.8)	92.4(±72.2)
MCAT	1982.1(±341.5)	1776.8(±680.8)	67.1(±152.9)	211.8(±39.8)	155.7(±65.7)
Improvement(%)	32.3	206.8	118.4	12.8	68.5

Table 3: Mean (± standard error) of test rewards at 1M timesteps. We report improvements brought by PT.

More Sparse Rewards We analyze MCAT when rewards are delayed for different numbers of steps (Tab. 4). When rewards are relatively dense (i.e. delay step is 200), during training, the learned policy can reach a higher score on each task without the issue of imbalanced performance among multiple tasks. MCAT w/o PT and MCAT perform comparably well within the standard error. However, as the rewards become more sparse, it requires a longer sequence of correct actions to obtain potentially high rewards. Policy learning struggles on some tasks and policy transfer plays an important role to exploit the precious good experiences on source tasks. Tab. 4 suggests that policy transfer brings more improvement on sparser-reward tasks. In Appendix, we further provide ablative study about More Diverse Tasks (D.3), Effect of Self-Imitation Learning (D.4), Effect of Contrastive Loss (D.5), and Design Choice of Action Translator (D.6).

Setting	Armature			Mass		
Delay steps	200	350	500	200	350	500
MCAT w/o PT	2583.2($\pm$280.4)	1771.7($\pm$121.9)	579.1($\pm$527.1)	709.6($\pm$386.6)	156.6($\pm$434.9)	-364.2($\pm$198.5)
MCAT	2251.8($\pm$556.9)	2004.5($\pm$392.5)	1776.8($\pm$680.8)	666.7($\pm$471.0)	247.8($\pm$176.1)	67.1($\pm$152.9)
Improvement(%)	-12.8	13.1	206.9	-6.1	58.2	118.4

Table 4: Test rewards at 1M timestpes averaged over 3 runs, on HalfCheetah with *armature* / *mass* changes.

Source Task	Target Task	Source policy	Transferred policy [41]	Transferred policy (Ours)
$[-0.1, 0.8, 0.2]$	$[0.1, 0.8, 0.2]$	947.5	1798.2($\pm$ 592.4)	**3124.3**($\pm$ 1042.0)
$[-0.1, 0.8, 0.2]$	$[0.05, 0.8, 0.2]$	1470.2	1764.0($\pm$ 316.3)	**1937.1**($\pm$ 424.5)
$[-0.1, 0.8, 0.2]$	$[0.1, 0.8, 0.05]$	1040.8	**2393.7**($\pm$ 869.8)	2315.7($\pm$ 1061.5)
2-leg HalfCheetah	3-leg HalfCheetah	NA	1957.8($\pm$298.4)	**2018.2**($\pm$50.8)

Table 5: Performance of source and transferred policy on target task, over 3 runs.

6 Discussion

While the scope of MCAT is for tasks with varying dynamics functions (same as many prior works [38, 16, 17, 42, 12]), our theory of policy transfer can be extended and the method can be potentially applied on more general cases (1) tasks with varying reward functions (2) tasks with varying state spaces and action spaces.

Following the idea in Sec. 3, on two general MDPs, we are interested in equivalent state-action pairs achieving the same reward and transiting to equivalent next states. Similar to Proposition 1, we can prove that, on two general MDPs, for two correspondent states $s^{(i)}$ and $s^{(j)}$, the value difference $|V^{\pi^{(i)}}(s^{(i)}, \mathcal{T}^{(i)}) - V^{\pi^{(j)}}(s^{(j)}, \mathcal{T}^{(j)})|$ is upper bounded by $\frac{d}{1-\gamma}$, where d depends on D_{TV} between the next state distribution on source task and the probability distribution of correspondent next state on target task. As an extension, we learn a state translator jointly with our action translator to capture state and action correspondence. Compared with Zhang et al. [41] learning both state and action translator, we simplify the objective function training action translator and afford the theoretical foundation. For (1) tasks with varying reward functions, we conduct experiments on MetaWorld moving the robot arm to a goal location. The reward at each step is inversely proportional to its distance from the goal location. We fix a goal location $[-0.1, 0.8, 0.2]$ on source task. We set target tasks with distinct goal locations (coordinates $[x, y, z]$ in Tab. 5) and hence with reward functions different from source task. Furthermore, we evaluate our approach on 2-leg and 3-leg HalfCheetah. We can test our idea on (2) tasks with varying state and action spaces of different dimensions because the agents have different numbers of joints on the source and target task. Tab. 5 demonstrates that ours with a simpler objective function than the baseline [41] can transfer the source policy to perform well on the target task. Details of theorems, proofs, and experiments are in Appendix E. Videos of the agents' behavior are on the webpage[1].

7 Conclusion

Meta-RL with long-horizon, sparse-reward tasks is challenging because an agent can rarely obtain positive rewards, and handling multiple tasks simultaneously requires massive samples from distinctive tasks. We propose a simple yet effective objective function to learn an action translator for multiple tasks and provide the theoretical ground. We develop a novel algorithm MCAT using the action translator for policy transfer to improve the performance of off-policy, context-based meta-RL algorithms. We empirically show its efficacy in various environments and verify that our policy transfer can offer substantial gains in sample complexity.

References

[1] R. Agarwal, M. C. Machado, P. S. Castro, and M. G. Bellemare. Contrastive behavioral similarity embeddings for generalization in reinforcement learning. *arXiv preprint arXiv:2101.05265*, 2021.

[2] H. B. Ammar and M. E. Taylor. Reinforcement learning transfer via common subspaces. In *International Workshop on Adaptive and Learning Agents*, pages 21–36. Springer, 2011.

[3] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym, 2016.

[4] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel. Rl 2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

[5] R. Fakoor, P. Chaudhari, S. Soatto, and A. J. Smola. Meta-q-learning. *arXiv preprint arXiv:1910.00125*, 2019.

[6] N. Ferns, P. Panangaden, and D. Precup. Metrics for finite markov decision processes. In *UAI*, volume 4, pages 162–169, 2004.

[7] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR, 2017.

[8] S. Fujimoto, H. Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pages 1587–1596. PMLR, 2018.

[9] A. Gupta, C. Devin, Y. Liu, P. Abbeel, and S. Levine. Learning invariant feature spaces to transfer skills with reinforcement learning. *arXiv preprint arXiv:1703.02949*, 2017.

[10] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[11] G. Konidaris and A. Barto. Autonomous shaping: Knowledge transfer in reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 489–496, 2006.

[12] K. Lee, Y. Seo, S. Lee, H. Lee, and J. Shin. Context-aware dynamics model for generalization in model-based reinforcement learning. In *International Conference on Machine Learning*, pages 5757–5766. PMLR, 2020.

[13] E. Z. Liu, A. Raghunathan, P. Liang, and C. Finn. Explore then execute: Adapting without rewards via factorized meta-reinforcement learning. *arXiv preprint arXiv:2008.02790*, 2020.

[14] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel. A simple neural attentive meta-learner. *arXiv preprint arXiv:1707.03141*, 2017.

[15] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 2015.

[16] A. Nagabandi, I. Clavera, S. Liu, R. S. Fearing, P. Abbeel, S. Levine, and C. Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

[17] A. Nagabandi, C. Finn, and S. Levine. Deep online learning via meta-learning: Continual adaptation for model-based rl. *arXiv preprint arXiv:1812.07671*, 2018.

[18] J. Oh, Y. Guo, S. Singh, and H. Lee. Self-imitation learning. In *International Conference on Machine Learning*, pages 3878–3887. PMLR, 2018.

[19] E. Parisotto, J. L. Ba, and R. Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv preprint arXiv:1511.06342*, 2015.

[20] D. Pollard. Asymptopia: an exposition of statistical asymptotic theory, 2000.

[21] K. Rakelly, A. Zhou, C. Finn, S. Levine, and D. Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pages 5331–5340. PMLR, 2019.

[22] H. Ren, Y. Zhu, J. Leskovec, A. Anandkumar, and A. Garg. Ocean: Online task inference for compositional tasks with context adaptation. In *Conference on Uncertainty in Artificial Intelligence*, pages 1378–1387. PMLR, 2020.

[23] A. A. Rusu, S. G. Colmenarejo, C. Gulcehre, G. Desjardins, J. Kirkpatrick, R. Pascanu, V. Mnih, K. Kavukcuoglu, and R. Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015.

[24] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[25] Y. Seo, K. Lee, I. Clavera, T. Kurutach, J. Shin, and P. Abbeel. Trajectory-wise multiple choice learning for dynamics generalization in reinforcement learning. *arXiv preprint arXiv:2010.13303*, 2020.

[26] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[27] B. C. Stadie, G. Yang, R. Houthooft, X. Chen, Y. Duan, Y. Wu, P. Abbeel, and I. Sutskever. Some considerations on learning to explore via meta-reinforcement learning. *arXiv preprint arXiv:1803.01118*, 2018.

[28] F. Sung, L. Zhang, T. Xiang, T. Hospedales, and Y. Yang. Learning to learn: Meta-critic networks for sample efficient learning. *arXiv preprint arXiv:1706.09529*, 2017.

[29] Y. Tang. Self-imitation learning via generalized lower bound q-learning. *arXiv preprint arXiv:2006.07442*, 2020.

[30] Y. W. Teh, V. Bapst, W. M. Czarnecki, J. Quan, J. Kirkpatrick, R. Hadsell, N. Heess, and R. Pascanu. Distral: Robust multitask reinforcement learning. *arXiv preprint arXiv:1707.04175*, 2017.

[31] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

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

[33] J. X. Wang, Z. Kurth-Nelson, D. Tirumala, H. Soyer, J. Z. Leibo, R. Munos, C. Blundell, D. Kumaran, and M. Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

[34] Z. Xu, H. van Hasselt, and D. Silver. Meta-gradient reinforcement learning. *arXiv preprint arXiv:1805.09801*, 2018.

[35] H. Yin and S. Pan. Knowledge transfer for deep reinforcement learning with hierarchical experience replay. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[36] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn. Gradient surgery for multi-task learning. *arXiv preprint arXiv:2001.06782*, 2020.

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

[38] W. Yu, J. Tan, C. K. Liu, and G. Turk. Preparing for the unknown: Learning a universal policy with online system identification. *arXiv preprint arXiv:1702.02453*, 2017.

[39] A. Zhang, C. Lyle, S. Sodhani, A. Filos, M. Kwiatkowska, J. Pineau, Y. Gal, and D. Precup. Invariant causal prediction for block mdps. In *International Conference on Machine Learning*, pages 11214–11224. PMLR, 2020.

[40] A. Zhang, S. Sodhani, K. Khetarpal, and J. Pineau. Learning robust state abstractions for hidden-parameter block $\{$mdp$\}$ s. In *International Conference on Learning Representations*, 2020.

[41] Q. Zhang, T. Xiao, A. A. Efros, L. Pinto, and X. Wang. Learning cross-domain correspondence for control with dynamics cycle-consistency. *arXiv preprint arXiv:2012.09811*, 2020.

[42] W. Zhou, L. Pinto, and A. Gupta. Environment probing interaction policies. *arXiv preprint arXiv:1907.11740*, 2019.

[43] Z. Zhu, K. Lin, and J. Zhou. Transfer learning in deep reinforcement learning: A survey. *arXiv preprint arXiv:2009.07888*, 2020.