ADARL: WHAT, WHERE, AND HOW TO ADAPT IN TRANSFER REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

One practical challenge in reinforcement learning (RL) is how to make quick adaptations when faced with new environments. In this paper, we propose a principled framework for adaptive RL, called *AdaRL*, that adapts reliably and efficiently to changes across domains with a few samples from the target domain, even in partially observable environments. Specifically, we leverage a parsimonious graphical representation that characterizes structural relationships over variables in the RL system. Such graphical representations provide a compact way to encode what and where the changes across domains are, and furthermore inform us with a minimal set of changes that one has to consider for the purpose of policy adaptation. We show that by explicitly leveraging this compact representation to encode changes, we can efficiently adapt the policy to the target domain, in which only a few samples are needed and further policy optimization is avoided. We illustrate the efficacy of AdaRL through a series of experiments that vary factors in the observation, transition and reward functions for Cartpole and Atari games.

1 INTRODUCTION AND RELATED WORK

Over the last decades, reinforcement learning (RL) (Sutton and Barto, 1998) has been successful in many tasks (Mnih et al., 2013; Silver et al., 2016). Most of these early successes focus on a fixed task in a fixed environment. However, in real applications we often have changing environments, and it has been demonstrated that the optimal policy learned in a specific domain may not be generalized to other domains (Taylor and Stone, 2009). In contrast, humans are usually good at transferring acquired knowledge to new environments and tasks both efficiently and effectively (Pearl and Mackenzie, 2018), thanks to the ability to understand the environments. Generally speaking, to achieve reliable, low-cost, and interpretable transfer, it is essential to understand the underlying process—which decision-making factors have changes, where the changes are, and how they change, instead of transferring blindly (e.g., transferring the distribution of high-dimensional images directly).

There are roughly two research lines in transfer RL (Taylor and Stone, 2009; Zhu et al., 2020): (1) finding policies that are robust to environment variations, and (2) adapting policies from the source domain to the target domain as efficiently as possible. For the first line, the focus is on learning policies that are robust to environment variations, e.g., by maximizing a risk-sensitive objective over a distribution of environments (Tamar et al., 2015) or by extracting a set of invariant states (Zhang et al., 2020a; 2021a; Tomar et al., 2021). A more recent method encodes task-relevant invariances by putting behaviorally equivalent states together, which helps better generalization (Agarwal et al., 2021). On the other hand, with the increase of the number of domains, the common part may get even smaller, running counter to the intention of collecting more information with more domains. Moreover, focusing only on the invariant part and disregarding domain-specific information may not be optimal; for instance, in the context of domain adaptation, it has been demonstrated that the variable part also contains information helpful to improve prediction accuracy (Zhang et al., 2020b).

In this paper, we propose a method along the second line, adapting source policies to the target. Approaches along this line adapt knowledge from source domains and reuse it in the target domain to improve data efficiency, i.e., in order for the agent to require fewer explorations to learn the target policy. For example, an agent could use importance reweighting on samples $\langle s, a, r, s' \rangle$ from sources (Tirinzoni et al., 2018; 2019) or start from the optimal source policy to initialize a learner in the target domain, as a near-optimal initializer (Taylor et al., 2007; Fernández et al., 2010). Another widely-used technique is finetuning: a model is pretrained on a source domain and the output layers are finetuned via backpropagation in the target domain (Hinton and Salakhutdinov, 2006; Mesnil

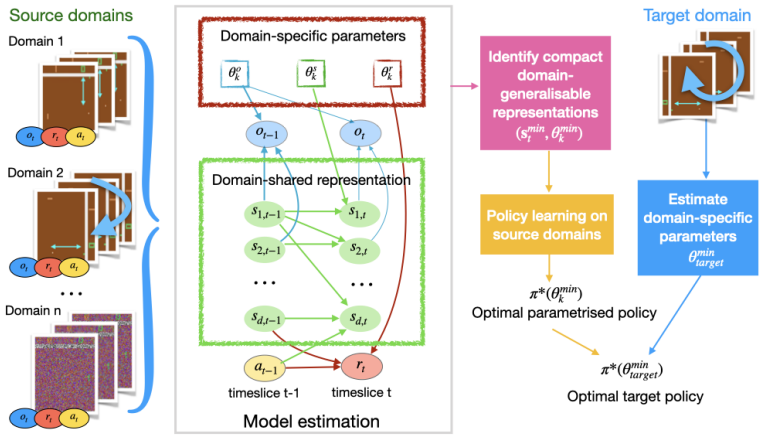

Figure 1: The overall AdaRL framework. We learn a Dynamic Bayesian Network (DBN) over the observations, latent states, reward, actions and domain-specific change factors that is shared across the domains. We then characterize a minimal set of representations that suffice for policy transfer, so that we can quickly adapt the optimal source policy with only a few samples from the target domain.

et al., 2012). PNNs (Rusu et al., 2016), instead, retain a pool of pretrained models and learn lateral connections from them to extract useful features for a new task. Moreover, a set of approaches focus on sim2real transfer by adapting the parameters (Yu et al., 2017; Peng et al., 2020). However, many of these approaches still require a large amount of explorations and optimization in the target domain.

Recently, meta-RL approches such as MAML (Finn et al., 2017), PEARL (Rakelly et al., 2019), CAVIA (Zintgraf et al., 2019), Meta-Q learning (Fakoor et al., 2020), and others (Mendonca et al., 2019; Nagabandi et al., 2018; Duan et al., 2016) have been successfully applied to learn an inductive bias that accelerates the learning of a new task by training on a large number of tasks. Some of these methods (e.g., CAVIA and PEARL), as well as some prior work (e.g., HiMDPs (Doshi-Velez and Konidaris, 2016)) and recent follow-ups (Zhang et al., 2021b), have a similar motivation to our work: in a new environment not all parameters need to be updated, so we can force the model to only adapt a set of *context parameters*. However, these methods mostly focus on MDPs (except the Block MDP assumption in Zhang et al. (2021b)) and model all changes as a black-box, which may be less efficient for adaptation, as opposed to a factorized representation of change factors.

Considering these limitations, we propose AdaRL, a transfer RL approach that achieves low-cost, reliable, and interpretable transfer for partially observable Markov decision processes (POMDPs), with MDPs as a special case. In contrast to state-of-the-art approaches, we learn a parsimonious graphical representation that is able to characterize structural relationships among different dimensions of states, change factors, the perception, the reward variable, and the action variable. It allows us to model changes in transition, observation, and reward functions in a component-wise way. This representation is related to Factored MDPs (Kearns and Koller, 1999; Boutilier et al., 2000; Strehl et al., 2007) and Factored POMDPs (Katt et al., 2019), but augmented with change factors that represent a low-dimensional embedding of the changes across domains. Our main motivation is that distribution shifts are usually localized – they are often due to the changes of only a few variables in the generative processes, so we can just adapt the distribution of a small portion of variables (Huang et al., 2020; Schölkopf et al., 2021) and, furthermore, factorized according to the graph structure, each distribution module can be adapted separately (Schölkopf, 2019; Zhang et al., 2020b).

In Fig. 1 we give a motivating example and a general description of AdaRL. In this example, we consider learning policies for Pong (Bellemare et al., 2013) that can easily generalize to different rotations ω and to images corrupted with white noise. Specifically, given data from n source domains with different rotations and noise variances, we learn a parsimonious latent state representation shared by all domains, denoted by $\mathbf{s}_t$, and characterize the changes across domains by a two-dimensional factor $\boldsymbol{\theta}_k$. We identify a set of minimal sufficient representations $(\mathbf{s}_t^{min}, \boldsymbol{\theta}_k^{min})$ for policy transfer. For instance, here only the rotation factor ω needs adapting (i.e., $\boldsymbol{\theta}_k^{min} = \omega_k$), since the noise factor does not affect the optimal policy. Similarly, as we will show formally in the rest of the paper, not all components $s_{i,t}$ of the state vector $\mathbf{s_t}$ are necessary for policy transfer. For example, $s_{2,t} \notin \mathbf{s}_t^{min}$,

since it never affects the future reward. We learn an optimal policy $\pi^*(\cdot|\boldsymbol{\theta}_k^{min})$ on source domains [1]. In the target domain, we only need a few samples to quickly estimate the value of the low-dimensional $\boldsymbol{\theta}_{\text{target}}^{min}$, and then we can apply $\pi^*(\cdot|\boldsymbol{\theta}_{target}^{min})$ directly. Our main contributions are summarized below:

- We assume a generative environment model, which explicitly takes into account the structural relationships among variables in the RL system. Such graphical representations provide a compact way to encode what and where the changes across domains are.
- Based on this model, we characterize a minimal set of representations that suffice for policy learning across domains, including the domain-specific change factors and domain-shared state representations. With this characterization, we adapt the policy with only a few target samples and without policy optimization in the target domain, achieving low-cost and reliable policy transfer.
- By leveraging a compact way to encode the changes, we also benefit from multi-task learning in model estimation. In particular, we propose the Multi-model Structured Sequential Variational Auto-Encoder (MiSS-VAE) for reliable model estimation in general cases.

2 A Compact Representation of Environmental Shifts

Suppose there are n source domains and n' target domains. In each source domain, we observe sequences $\{\langle o_t, a_t, r_t\rangle\}_{t=1}^T$, where $o_t \in \mathcal{O}$ are the perceived signals at time t (e.g., images), $a_t \in \mathcal{A}$ is the executed action, and $r_t \in \mathcal{R}$ is the reward signal. We denote the underlying latent states by $\mathbf{s}_t = (s_{1,t}, \cdots, s_{d,t})^\top$, where d is the dimensionality of latent states. We assume that the generative process of the environment in the k-th domain (with $k = 1, \dots n + n'$) can be described in terms of the transition function for each dimension of $\mathbf{s}$ and the observation and reward functions as

$$\begin{cases} s_{i,t} &= f_i(\mathbf{c}_i^{\mathbf{s}\to\mathbf{s}} \odot \mathbf{s}_{t-1}, c_i^{a\to\mathbf{s}} \cdot a_{t-1}, \mathbf{c}_i^{\theta_k\to\mathbf{s}} \odot \boldsymbol{\theta}_k^{\mathbf{s}}, \epsilon_{i,t}^s), \text{ for } i = 1, \cdots, d, \\ o_t &= g(\mathbf{c}^{\mathbf{s}\to o} \odot \mathbf{s}_t, c^{\theta_k\to o} \cdot \theta_k^o, \epsilon_t^o), \\ r_t &= h(\mathbf{c}^{\mathbf{s}\to r} \odot \mathbf{s}_{t-1}, c^{a\to r} \cdot a_{t-1}, c^{\theta_k\to r} \cdot \theta_k^r, \epsilon_t^r), \end{cases} \tag{1}$$

where $\odot$ denotes the element-wise product, the $\epsilon_{i,t}^s, \epsilon_t^o, \epsilon_t^r$ terms are i.i.d. random noises. As explained below, $\mathbf{c}^{\cdot\to\cdot}$ are *masks* (binary vectors or scalars that represent structural relationships from one variable to the other), and $\boldsymbol{\theta_k} = (\theta_k^{\mathbf{s}}, \theta_k^o, \theta_k^r)$ are the change factors that have a constant value in each domain, but vary across domains in the transition, observation, and reward function, respectively. The latent states $\mathbf{s}_{t+1}$ form an MDP: given $\mathbf{s}_t$ and a_t, $\mathbf{s}_{t+1}$ is independent of previous states and actions. The perceived signals o_t are generated from the underlying states $\mathbf{s}_t$. The actions a_t directly influence the latent states $\mathbf{s}_{t+1}$, instead of the observed signals o_t, and the reward is determined by the latent states and the action. Eq. 1 can also represent MDPs as a special case if states $\mathbf{s_t}$ are directly observed, in which case the observation function of o_t is not needed.

Structural relationships and graphs. Often the action variable a_{t-1} does not influence every dimension of $\mathbf{s}_t$, and similarly, the reward r_t may not be influenced by every dimension of $\mathbf{s}_{t-1}$. Furthermore, there are structural relationships between different dimensions of $\mathbf{s}_{t-1}$ and $\mathbf{s}_t$. To characterize these constraints, we explicitly take into account the graph structure $\mathcal{G}$ over the variables in the system characterized by a Dynamic Bayesian Network (Murphy, 2002) and encode the edges with masks $\mathbf{c}^{\cdot\to\cdot}$. In the first equation in Eq. 1 the transition function for the state component s_i, where the jth entry of $\mathbf{c}_i^{\mathbf{s}\to\mathbf{s}} \in \{0,1\}^d$ is 1 if and only if $s_{j,t}$ influences $s_{i,t+1}$ (graphically represented by an edge), while $c_i^{a\to\mathbf{s}} \in \{0,1\}$ is 1 if and only if the action a_t has any effect on $s_{i,t+1}$. Similarly, the binary vector $\mathbf{c}_i^{\theta_k\to\mathbf{s}} \in \{0,1\}^p$ encodes which components of the change factor $\boldsymbol{\theta}_k^{\mathbf{s}} = (\theta_{1,k}^s, \dots, \theta_{p,k}^s)^\top$ affect $s_{i,t+1}$. The masks in the observation function g and reward function h have similar functions. The masks and the parameters of the functions f, g, and h, are invariant; all changes are encoded in $\boldsymbol{\theta}_k$. For simplicity of notation, we collect all the transition mask vectors in the matrices $\mathbf{C}^{\mathbf{s}\to\mathbf{s}} := [\mathbf{c}_i^{\mathbf{s}\to\mathbf{s}}]_{i=1}^d$ and $\mathbf{C}^{\theta_k\to\mathbf{s}} := [\mathbf{c}_i^{\theta_k\to\mathbf{s}}]_{i=1}^d$ and the scalars in the vector $\mathbf{c}^{a\to\mathbf{s}} := [c_i^{a\to\mathbf{s}}]_{i=1}^d$.

Characterization of change factors in a compact way. In practical scenarios, the environment model may change across domains. Moreover, it is often the case that given a high-dimensional input, only a few factors may change, which is known as *minimal change principle* (Ghassami et al., 2018) or *sparse mechanism shift assumption* (Schölkopf et al., 2021). In such a case, instead of learning the distribution shift over the high-dimensional input, thanks to the parsimonious graphical representation, we introduce a low-dimensional vector $\boldsymbol{\theta}_k$ to characterize the domain-specific information in a compact way (Zhang et al., 2020b). Specifically, θ_k^o, θ_k^r, and $\boldsymbol{\theta}_k^{\mathbf{s}}$ capture the change factors in the observation function, reward function, and transition dynamics, respectively; each of them can be

[1] We consider optimality of the policy w.r.t. the model estimated on source domains and AdaRL assumptions.

multi-dimensional and that they are constant within each domain. In general, $\boldsymbol{\theta}_k$ can capture both the changes in the influencing strength and those in the graph structure, e.g., some edges may appear only in some domains. Since we assume that the structural relationships in Eq. 1 are invariant across domains, this means that the masks $\mathbf{c}^{\cdot\to\cdot}$ have to encode an edge even if it presents only in one domain, and furthermore, since $\boldsymbol{\theta}_k$ encodes the changes, it can switch the edge off in other domains. Fig. 1 shows an example of the graphical representation of the (estimated) environment model. Specifically, in this example, θ_k^s only influences $s_{1,t}$, a_{t-1} does not have an edge to $s_{1,t}$, and among the states, only $s_{d,t-1}$ has an edge to r_t. In this example, we consider the case when the control signals are random, so there is no edge between $\mathbf{s}_t$ and a_t.

3 What, Where, and How to Adapt in RL

We first assume that the environment model in Eq. 1 is known (we will explain how to learn it in Sec. 3.1), and characterize which changes have an effect on the policy transfer to the target domain. In Eq. 1, we allow the model to change across domains, including all involved functions, and we leverage $\boldsymbol{\theta}_k$ to capture the changes in a compact way. The varying model implies that the optimal policy function may also vary across domains. How can we then characterize the changes in the optimal policy function in a compact way, as we did in the model? Interestingly, we find that the change factor $\boldsymbol{\theta}_k$ and the latent state $\mathbf{s}_t$ are sufficient for policy learning, but not every dimension of $\boldsymbol{\theta}_k$ or $\mathbf{s}_t$ is necessary, since they may not ever have an effect on the reward, even in future steps. We first give the definitions of *compact domain-shared representations* and *compact domain-specific representations*, according to the graph structure, and we further show that they are the minimal set of dimensions that suffice for policy learning across domains (proof in Appendix).

Definition 1. *Given the graphical representation of an environment model $\mathcal{G}$ that is encoded in the binary masks $\mathbf{c}^{\cdot\to\cdot}$, we define recursively the representations that affect the reward in the future as:*

- compact domain-shared representations $\mathbf{s}_t^{min}$: *the latent state components $s_{i,t} \in \mathbf{s}_t$ that either*
  - *have an edge to the reward in the next time-step r_{t+1}, i.e., $\mathbf{c}_i^{\mathbf{s}\to r} = 1$, or*
  - *have an edge to another state component in the next time-step $s_{j,t+1}$, i.e., $\mathbf{c}_{j,i}^{\mathbf{s}\to\mathbf{s}} = 1$, such that the same component at time t is a compact domain-shared representation, i.e., $s_{j,t} \in \mathbf{s}_t^{min}$;*
- compact domain-specific representations $\boldsymbol{\theta}_k^{min}$: *the latent change factors $\theta_{i,k} \in \boldsymbol{\theta}_k$ that either:*
  - *have an edge to the reward in the next time-step r_{t+1}, i.e., $\theta_{i,k} = \theta_k^r$ and $c^{\theta_k\to r} = 1$, or*
  - *have an edge to a state component $s_{j,t} \in \mathbf{s}_t^{min}$, i.e., $\mathbf{c}_{j,i}^{\theta_k\to s} = 1$.*

Proposition 1. *Under the assumption that the graph $\mathcal{G}$ is Markov and faithful to the measured data, the union of compact domain-specific $\boldsymbol{\theta}_k^{min}$ and compact shared representations $\boldsymbol{s}_t^{min}$ are the minimal and sufficient dimensions for policy learning across domains.*

For the example in Fig. 1, $\mathbf{s}_t^{min} = (s_{1,t}, s_{d,t})$ and $\boldsymbol{\theta}_k^{min} = \{\theta_k^s, \theta_k^r\}$. Note that θ_k^o is never in $\boldsymbol{\theta}_k^{min}$, and thus if only the observation function changes, the optimal policy function π_k^* remains the same across domains. For example in Cartpole a change of color does not affect the optimal policy. Moreover, if $c^{\theta_k\to r} = 1$, then $\theta_k^r \in \boldsymbol{\theta}_k^{min}$, which is the case for multi-task learning.

3.1 Simultaneous Estimation of Domain-Varying Models

In this section, we give the estimation procedure of the environment model in Eq. 1 from observed sequences $\{\{\langle \mathbf{y}_{t,k}, a_{t,k}\rangle\}_{t=1}^T\}_{k=1}^n$ from each source domain k, where $\mathbf{y}_{t,k} = (o_{t,k}^\top, r_{t,k}^\top)^\top$ are the observations and reward at time t in domain k. Instead of estimating the model in each domain separately, we estimate models from different domains simultaneously, by exploiting commonalities across domains while at the same time preserving specific information for each domain. In particular, we propose the Multi-model Structured Sequential Variational Auto-Encoder **(MiSS-VAE)**, which contains the following three essential components. (1) *"Sequential VAE" component* handles the sequential data, with the underlying latent states satisfying an MDP. It is implemented by adding an LSTM (Hochreiter and Schmidhuber, 1997) to encode the sequential information in the encoder to learn the inference model $q_\phi(\mathbf{s}_{t,k}|\mathbf{s}_{t-1,k}, \mathbf{y}_{1:t,k}, a_{1:t-1,k}; \boldsymbol{\theta}_k)$. (2) *"Multi-model" component* handles models from different domains at the same time, using the domain index k as an input and learning the domain-specific factors $\boldsymbol{\theta}_k$. (3) *"Structured" component*: exploits the structural information that is explicitly encoded with the binary masks, i.e., $\mathbf{c}^{\cdot\to\cdot}$ in Eq. 1. Here, the joint distribution of latent states are factorized according to structures, instead of being marginally independent as in traditional VAEs (Kingma and Welling, 2013). Figure 2 gives the diagram of neural network architecture in

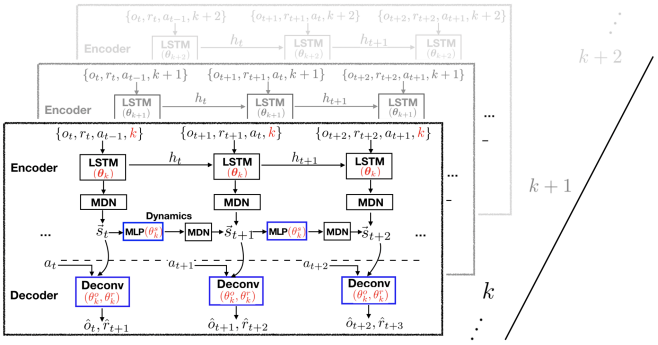

Figure 2: Diagram of MiSS-VAE neural network architecture. The "sequential VAE" component, "multi-model" component, and "structure" component are marked with black, red, and blue, respectively.

model training. Let $\mathbf{y}_{1:T}^{1:n} = \{\{\mathbf{y}_{t,k}\}_{t=1}^{T}\}_{k=1}^{n}$. By taking into account the above three components, we maximize the following objective function $\mathcal{L}$:

$$\mathcal{L}(\mathbf{y}_{1:T}^{1:n}; (\beta_1, \beta_2, \phi, \gamma, \mathbf{c}^{\cdot}) = \mathcal{L}^{\text{rec}}(\mathbf{y}_{1:T}^{1:n}; (\beta_1, \phi, \mathbf{c}^{\cdot})) + \mathcal{L}^{\text{pred}}(\mathbf{y}_{1:T}^{1:n}; (\beta_2, \phi)) - \mathcal{L}^{\text{KL}}(\mathbf{y}_{1:T}^{1:n}; (\phi, \gamma, \mathbf{c}^{\cdot})) - \mathcal{L}^{\text{reg}}.$$

In particular, $\mathcal{L}^{\text{rec}}$ is the reconstruction loss for both observed images and rewards, to learn the observation and the reward function, respectively. We also consider the one-step prediction loss $\mathcal{L}^{\text{pred}}$.

$$\mathcal{L}^{\text{rec}} = \sum_{k=1}^{n} \sum_{t=1}^{T-2} \mathbb{E}_{\mathbf{s}_{t,k} \sim q_\phi(\cdot|\boldsymbol{\theta}_k)} \{\log p_{\beta_1}(o_{t,k}|\mathbf{s}_{t,k}; \theta_k^o, c^{\theta_k \to o}, \mathbf{c}^{\mathbf{s} \to o}) + \log p_{\beta_1}(r_{t+1,k}|\mathbf{s}_{t,k}, a_{t,k}; \theta_k^r, c^{\theta_k \to r}, \mathbf{c}^{\mathbf{s} \to r}, c^{a \to r})\},$$

$$\mathcal{L}^{\text{pred}} = \sum_{k=1}^{n} \sum_{t=1}^{T-2} \mathbb{E}_{\mathbf{s}_{t,k} \sim q_\phi(\cdot|\boldsymbol{\theta}_k)} \{\log p_{\beta_2}(o_{t+1,k}|\mathbf{s}_{t,k}, \theta_k^o, \theta_k^s) + \log p_{\beta_2}(r_{t+2,k}|\mathbf{s}_{t,k}, a_{t+1,k}; \theta_k^r, \theta_k^s)\},$$

where p_{β_1} and p_{β_2} denote the generative models with parameters β_1 and β_2, respectively, that are shared across domains, and q_ϕ the inference model with shared parameters ϕ. We also use the following KL-divergence loss to constrain the latent space:

$$\mathcal{L}^{\text{KL}} = \lambda_0 \sum_{k=1}^{n} \sum_{t=2}^{T} \text{KL}\big(q_\phi(\mathbf{s}_{t,k}|\mathbf{s}_{t-1,k}, \mathbf{y}_{1:t,k}, a_{1:t-1,k}; \boldsymbol{\theta}_k) \| p_\gamma(\mathbf{s}_{t,k}|\mathbf{s}_{t-1,k}, a_{t-1,k}; \theta_k^s, \mathbf{C}^{\mathbf{s} \to \mathbf{s}}, \mathbf{c}^{a \to \mathbf{s}}, \mathbf{C}^{\theta_k \to \mathbf{s}})\big),$$

where we explicitly model the transition dynamics p_γ with the parameters γ shared across domains; this is essential for establishing a Markov chain in latent space and learning a representation for long-term predictions. Moreover, the KL loss helps to constrain the latent space to (1) ensure that the disentanglement between the inferred latent factors $q(s_{i,t}|\cdot)$ and $q(s_{j,t}|\cdot)$ for $i \neq j$, since we do not consider the instantaneous connections among state dimensions, and (2) ensure that the latent representations $\mathbf{s}_t$ are maximally compressive about the observed high-dimensional data. Furthermore, according to the edge-minimality property (Zhang and Spirtes, 2011) and the minimal change principle (Ghassami et al., 2018), we add sparsity constraints on structural matrices and on the change of domain-specific factors across domains, respectively, to achieve better identifiability:

$$\mathcal{L}^{\text{reg}} = \lambda_1 \|\mathbf{c}^{\mathbf{s} \to o}\|_1 + \lambda_2 \|\mathbf{c}^{\mathbf{s} \to r}\|_1 + \lambda_3 \|c^{a \to r}\|_1 + \lambda_4 \|\mathbf{C}^{\mathbf{s} \to \mathbf{s}}\|_1 + \lambda_5 \|\mathbf{c}^{a \to \mathbf{s}}\|_1 + \lambda_6 \|\mathbf{C}^{\theta_k \to \mathbf{s}}\|_1 + \lambda_7 \sum_{1 \leq j,k \leq n} |\boldsymbol{\theta}_j - \boldsymbol{\theta}_k|.$$

Note that besides the shared parameters $\{\beta_1, \beta_2, \phi, \gamma\}$, the structural relationships (encoded in binary masks $\mathbf{c}$) are also involved in the shared parameters. Each factor in p_ϕ, p_{β_i}, and p_γ is modeled with a mixture of Gaussians, because with a suitable number of Gaussians, it can approximate a wide class of continuous distributions. Moreover, in model estimation, the domain-specific factors $\boldsymbol{\theta}_k = \{\theta_k^o, \theta_k^s, \theta_k^r\}$ are treated as parameters; they are constant within the same domain, but may differ in different domains. We explicitly consider $\boldsymbol{\theta}_k$ not only in the generative models p_{β_i} and p_γ, but also in the inference model q_ϕ. In this way, except for $\boldsymbol{\theta}_k$, all other parameters in MiSS-VAE are shared across domains, so that all we need to update in the target domain is the low-dimensional $\boldsymbol{\theta}_k$, which greatly improves the sample efficiency and the statistical efficiency in the target domain.

3.2 Low-Cost and Interpretable Policy Transfer

After identifying what and where to transfer, we show how to adapt. Instead of learning the optimal policy in each domain separately, which is time and sample inefficient, we leverage a multi-task learning strategy: policies in different domains are optimized at the same time exploiting both commonalities and differences across domains. Given the compact domain-shared $\mathbf{s}_t^{min}$ and domain-specific representations $\boldsymbol{\theta}_k^{min}$, we represent the optimal policies across domains in a unified way:

$$a_t = \pi^*(\mathbf{s}_t^{min}, \boldsymbol{\theta}_k^{min}), \tag{2}$$

where $\boldsymbol{\theta}_k^{min}$ explicitly and compactly encodes the changes in the policy function in each domain k, and all other parameters in the optimal policy function π^* are shared across domains. In other

words, by learning π^* in the source domains, and estimating the value of the change factor $\boldsymbol{\theta}^{min}_{\text{target}}$ and inferring latent states $\mathbf{s}^{min}_{\text{target}}$ from the target domain, we can immediately derive the optimal policy in the target domain without further policy optimization by just applying Eq. 2.

The AdaRL framework answers what and where the change factors are and which change factors need to adapt across domains in an interpretable way. Moreover, AdaRL only requires a few samples to update the low-dimensional domain-specific parameters $\boldsymbol{\theta}^{min}_{\text{target}}$ to achieve the optimal policy in the target domain, without further policy optimization, achieving the low cost. We provide the pseudocode for the AdaRL algorithm in Alg. 1. The algorithm has three parts: (1) data collection with a random policy or any initial policy from n source domains (line 2), (2) model estimation from the n source domains with multi-task learning (lines 2-3, see Sec. 3.1 for details), and (3) learning the optimal policy π^* with deep Q-learning, by making use of domain-specific factors and the inferred domain-shared state representations (lines 4-21). Specifically, because we do not directly observe the states $\mathbf{s}_t$, we infer $q(\mathbf{s}^{min}_{t+1,k}|o_{\leq t+1,k}, r_{\leq t+1,k}, a_{\leq t,k}, \boldsymbol{\theta}^{min}_k)$ and sample $\mathbf{s}^{min}_{t+1,k}$ from its posterior, for the kth domain (lines 7 and 13). Moreover, the action-value function Q is learned by considering the averaged error over the n source domains (line 18). AdaRL can be implemented with a wide class of policy-learning algorithms, e.g., DDPG (Lillicrap et al., 2015), Q-learning (Mnih et al., 2015), and Actor-Critic methods (Schulman et al., 2015; Mnih et al., 2016). Then, in the target domain, we only need to collect a few rollouts to estimate the low-dimensional domain-specific representations $\boldsymbol{\theta}^{min}_{\text{target}}$, with all other parameters being fixed (lines 22-23).

Algorithm 1 (AdaRL with Domains Shifts)

1: Initialize action-value function Q, target action-value function Q', and replay buffer $\mathcal{B}$.
2: Record multiple rollouts for each source domain $k (k = 1, \cdots, n)$ and estimate the model in Eq.1.
3: Identify the dimension indices of $\mathbf{s}^{min}_t$ and the values of $\boldsymbol{\theta}^{min}_k$ according to the learned model.
4: **for** episode = 1, ..., M **do**
5: **for** source domain k = 1, ..., n **do**
6: Receive initial observations $o_{1,k}$ and $r_{1,k}$ for the k-th domain.
7: Infer the posterior $q(\mathbf{s}^{min}_{1,k}|o_{1,k}, r_{1,k}, \boldsymbol{\theta}^{min}_k)$ and sample initial inferred state $\mathbf{s}^{min}_{1,k}$.
8: **end for**
9: **for** timestep t = 1, ..., T **do**
10: **for** source domain k = 1, ..., n **do**
11: Select $a_{t,k}$ randomly with probability ϵ; otherwise $a_{t,k} = \arg\max_a Q(\mathbf{s}^{min}_{t,k}, a, \boldsymbol{\theta}^{min}_k)$.
12: Execute action $a_{t,k}$, and receive reward $r_{t+1,k}$ and observation $o_{t+1,k}$ in the kth domain.
13: Infer the posterior $q(\mathbf{s}^{min}_{t+1,k}|o_{\leq t+1,k}, r_{\leq t+1,k}, a_{\leq t,k}, \boldsymbol{\theta}^{min}_k)$ and sample $\mathbf{s}^{min}_{t+1,k}$.
14: Store transition $(\mathbf{s}^{min}_{t,k}, a_{t,k}, r_{t+1,k}, \mathbf{s}^{min}_{t+1,k}, \boldsymbol{\theta}^{min}_k)$ in reply buffer $\mathcal{B}$.
15: **end for**
16: Randomly sample a minibatch of N transitions $(\mathbf{s}^{min}_{i,j}, a_{i,j}, r_{i+1,j}, \mathbf{s}^{min}_{i+1,j}, \boldsymbol{\theta}^{min}_j)$ from $\mathcal{B}$.
17: Set $y_{i,j} = r_{i+1,j} + \lambda \max_{a'} Q'(s^{min}_{i+1,j}, a', \boldsymbol{\theta}^{min}_j)$.
18: Update action-value function Q by minimizing the loss:

$$L = \frac{1}{n * N} \sum_{i,j} (y_{i,j} - Q(s^{min}_{i,j}, a_{i,j}, \boldsymbol{\theta}^{min}_j))^2.$$

19: **end for**
20: Update the target network Q': $Q' = Q$.
21: **end for**
22: Record a few rollouts from the target domain.
23: Estimate the values of $\boldsymbol{\theta}^{min}_{\text{target}}$ for the target domain, with all other parameters fixed.

3.3 Theoretical Properties

Below we show the conditions under which we can identify the true graph $\mathcal{G}$ from observational data, even when the model in Eq. 1 is unknown. Furthermore, we derive a generalization bound of the state-value function under the PAC-Bayes framework (McAllester, 1999).

Theorem 1 (Structural Identifiability)**.** *Suppose the underlying states $\mathbf{s}_t$ are observed, i.e., Eq. (1) is an MDP. Then under the Markov condition and faithfulness assumption, the structural matrices $\mathbf{C}^{\mathbf{s}\to\mathbf{s}}$, $\mathbf{c}^{a\to\mathbf{s}}$, $\mathbf{c}^{\mathbf{s}\to r}$, $c^{a\to r}$, $\mathbf{C}^{\theta_k\to\mathbf{s}}$, and $c^{\theta_k\to r}$ are identifiable.*

This theorem shows that in the MDP scenario, where the underlying states are observed and $c^{\theta_k \to o}$ and $\mathbf{c}^{\mathbf{s} \to o}$ are not considered by definition, we can uniquely determine the structural relationships over $\{\mathbf{s}_{t-1}, \mathbf{s}_t, a_{t-1}, r_t, \boldsymbol{\theta}_k\}$, i.e., the Dynamic Bayesian network $\mathcal{G}$, from observed data under mild conditions, without knowing the generative environment model. Even if $\boldsymbol{\theta}_k$ is not directly observed, we can identify which state dimension changes and if there is a change in the reward function.

Suppose there are n source domains, and for the kth domain, we have $S_k = \big((\mathbf{s}_{1,k}, v^*(\mathbf{s}_{1,k})), \cdots, (\mathbf{s}_{m_k,k}, v^*(\mathbf{s}_{m_k,k}))\big)$, where m_k is the number of samples from the kth domain, $\mathbf{s}_{\cdot,k}$ is a state sampled from the kth domain, and $v^*(\mathbf{s}_{\cdot,k})$ is its corresponding optimal state-value. For any value function $h_{\boldsymbol{\theta}_k^{min}}(\cdot)$ parameterized by $\boldsymbol{\theta}_k^{min}$, we define the loss function $\ell(h_{\boldsymbol{\theta}_k^{min}}, (\mathbf{s}_{k,i}, v^*(\mathbf{s}_{i,k}))) = D_{dist}(h_{\boldsymbol{\theta}_k^{min}}(\mathbf{s}_{i,k}), v^*(\mathbf{s}_{i,k}))$, where D_{dist} is a distance function that measures the discrepancy between the learned value and the optimal value. The following theorem gives a generalization bound of the state-value function under the PAC-Bayes framework.

Theorem 2 (Generalization Bound)**.** *Let $\mathcal{Q}$ be an arbitrary distribution over $\boldsymbol{\theta}_k^{min}$ and $\mathcal{P}$ the prior distribution over $\boldsymbol{\theta}_k^{min}$. Then for any $\delta \in (0, 1]$, with probability at least $1 - \delta$, the following inequality holds uniformly for all $\mathcal{Q}$,*

$$er(\mathcal{Q}) \leq \frac{1}{n} \sum_{k=1}^{n} \left\{ \hat{er}(\mathcal{Q}, S_k) + \sqrt{\frac{1}{2(m_k-1)} \left(D_{KL}(\mathcal{Q}||\mathcal{P}) + \log \frac{2nm_k}{\delta} \right)} + \sqrt{\frac{1}{2(n-1)} \left(D_{KL}(\mathcal{Q}||\mathcal{P}) + \log \frac{2n}{\delta} \right)} \right\},$$

where $er(\mathcal{Q})$ and $\hat{er}(\mathcal{Q}, S_k)$ are the generalization error and the training error between the estimated value and the optimal value, respectively.

Theorem 2 states that with high probability the generalization error $er(\mathcal{Q})$ is upper bounded by the empirical error plus two complexity terms. Specifically, the first one is the average of the task-complexity terms from the observed domains, which converges to zero in the limit of samples in each domain, i.e., $m_k \to \infty$. The second is an environment-complexity term, which converges to zero if infinite domains are observed, i.e., $n \to \infty$. Moreover, if assuming different dimensions of $\boldsymbol{\theta}_k^{min}$ are independent, then $D_{KL}(\mathcal{Q}||\mathcal{P}) = \sum_{i=1}^{|\boldsymbol{\theta}_k^{min}|} D_{KL}(\mathcal{Q}_i||\mathcal{P}_i)$, which indicates that a low-dimensional $\boldsymbol{\theta}_k^{min}$ usually has a smaller KL divergence, so does the upper bound of the generalization error.

4 Evaluation

We modify the Cartpole and Atari Pong environments in OpenAI Gym (Brockman et al., 2016). Here, we present a subset of the results; see Appendix for the complete results and the detailed settings. We consider changes in the state dynamics (e.g., the change of gravity or cart mass in Cartpole, change of orientation in Pong), changes in observations (e.g., different noise levels in images or different colors in Pong), and changes in reward functions (e.g., different reward functions in Pong based on the contact point of the ball), as shown in Fig. 3 for Pong. For each of these factors, we take into account both *interpolation* (where the factor value in the target domain is in the support of that in source domains), and *extrapolation* (where it is out of the support w.r.t. the source domains). We train on n source domains based on the trajectory data generated by a random policy. In Cartpole, for each domain we collect 10000 trials with 40 steps. For Pong experiments, each domain contains 40 episodes data and each of them takes a maximum of 10000 steps. In the target domain we consider different sample sizes with $N_{\text{target}} = \{20, 50, 10000\}$ to estimate $\boldsymbol{\theta}_{\text{target}}^{min}$. For both games, we evaluate the POMDP case, where the inputs are high-dimensional images; note that we did not stack multiple frames, so some properties (e.g., velocity) are not observed, resulting in a POMDP. For Cartpole, we also consider the MDP case, where the true states (cart position and velocity, pole angle and angular velocity) are used as the input to the model. In Cartpole, we also experiment with multiple factors changing at the same time (e.g., gravity and mass change concurrently in the target domain).

Modified Cartpole setting The Cartpole problem consists of a cart and a vertical pendulum attached to the cart using a passive pivot joint. The task is to prevent the vertical pendulum from falling by putting a force on the cart to move it left or right. We introduce two change factors for the state dynamics θ_k^s: varying gravity and varying mass of the cart. In terms of changes on the observation function θ_k^o, we add Gaussian noise on the images. Since θ_k^o does not influence the optimal policy (as shown in Prop. 1), we need it only for the model estimation, but not for policy optimization. Moreover, if $\boldsymbol{\theta}_k = \{\theta_k^o\}$, the optimal policy is shared across domains.

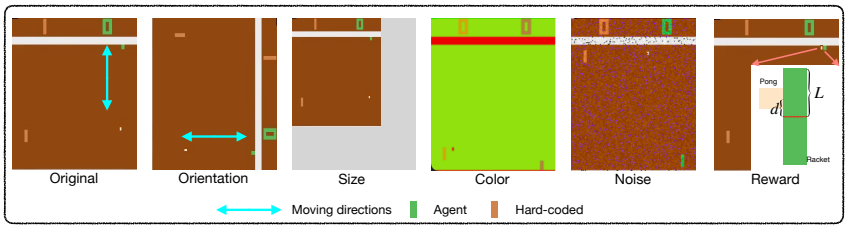

Figure 3: Illustrations of the change factors on modified Pong game.

	Oracle Upper bound	Non-t lower bound	CAVIA (Zintgraf et al., 2019)	PEARL (Rakelly et al., 2019)	AdaRL* Ours w/o masks	AdaRL Ours
G_in	2486.1 (±369.7)	1098.5 • (±472.1)	1603.0 (±877.4)	1647.4 (±617.2)	1940.5 (±841.7)	2217.6 (±981.5)
G_out	693.9 (±100.6)	204.6 • (±39.8)	392.0 • (±125.8)	434.5 • (±102.4)	439.5 • (±157.8)	**508.3** (±138.2)
M_in	2678.2 (±630.5)	748.5 • (±342.8)	2139.7 (±859.6)	1784.0 (±845.3)	1946.2 • (±496.5)	2260.2 (±682.8)
M_out	1405.6 (±368.0)	371.0 • (±92.5)	972.6 • (±401.4)	793.9 • (±394.2)	874.5 • (±290.8)	**1001.7** (±273.3)
G_in & M_in	1984.2 (±871.3)	365.0 • (±144.5)	1012.5 • (±664.9)	1260.8 • (±792.0)	1157.4 • (±578.5)	**1428.4** (±495.6)
G_out & M_out	939.4 (±270.5)	336.9 • (±139.6)	648.2 • (±481.5)	544.32 • (±175.2)	596.0 • (±184.3)	**689.4** (±272.5)

Table 1: Average final scores on modified Cartpole (MDP) with $N_{target} = 50$. The best non-oracle results w.r.t. the mean are marked in red, while **bold** indicates a statistically significant result w.r.t. all the baselines. G and M denote the gravity and mass respectively, and "$*$in" and "$*$out" denote the interpolation and extrapolation, respectively.

Modified Pong setting In Pong, one of the established Atari benchmarks (Bellemare et al., 2013), the agent controls a paddle moving up and down vertically, aiming at hitting the ball. We consider changes in observation function θ^o_k, state dynamics θ^s_k, and reward function θ^r_k, as shown in Fig. 3. We consider three change factors on perceived signals θ^o_k: different image sizes, different image colors, and different noise levels. For the setting with different image colors, we use RGB images as inputs and consider source domains with varying RGB colors {original, green, red} and target domains with colors {yellow, white}, but for other settings, we convert the images to grayscale as input. To change the state dynamics, we rotate the images ω degrees clockwise. To test the changes in the reward function, we model the reward as a function of the distance between contact point and the central point of the paddle, denoted by d, as opposed to the original Pong in which it is constant. We denote by L the half-length of the paddle and formulate two groups of reward functions: (1) Linear reward: $r_t = \frac{\alpha d}{L}$; and (2) Non-linear reward: $r_t = \frac{\alpha L}{d+3L}$, where α varies across domains.

Baselines In the MDP setting, we compare AdaRL with CAVIA (Zintgraf et al., 2019) and PEARL (Rakelly et al., 2019). In the POMDP setting, we compare with PNN (Rusu et al., 2016), PSM (Agarwal et al., 2021) and MQL (Fakoor et al., 2020). We also compare with *AdaRL**, a version of AdaRL that does not learn the binary masks $c^{\to \cdot}$ and therefore does not use any structural information. All of these methods use the same number of samples N_{target} from the target domain. We also compare with: 1) *Non-t*, a vanilla non-transfer baseline that pools data from all source domains and learns a fixed model; and 2) an oracle baseline, which is **completely** trained on the target domain with model-free exploration. For a fair comparison, we use the same policy learning algorithm, Double DQN (Van Hasselt et al., 2016), for all methods. As opposed to MAML and PNN, AdaRL only uses the N_{target} samples to estimate $\boldsymbol{\theta}^{min}_{target}$, without any policy optimization.

Results We measure performance by the mean and standard deviation of the final scores over 30 trials with different random seeds, as well as testing the significance with the Wilcoxon signed-rank test (Conover, 1999). As shown in Tables 1, 2 and 3 [2], AdaRL consistently outperforms the baselines across most change factors in the MDP and POMDP case for modified Cartpole, and in the POMDP case for Pong for $N_{target} = 50$. As ablation studies, to see the effect of learning

[2] In Table 1-4, "•" indicates the baselines for which the improvements of AdaRL are statistically significant (via Wilcoxon signed-rank test at 5% significance level).

	Oracle Upper bound	Non-t lower bound	PNN (Rusu et al., 2016)	PSM (Agarwal et al., 2021)	MTQ (Fakoor et al., 2020)	AdaRL* Ours w/o masks	AdaRL Ours
G_in	1930.5 (±1042.6)	1031.5 • (±837.9)	1268.5 • (±699.0)	1439.8 • (±427.6)	1517.9 (±883.6)	1460.6 (±497.5)	1697.4 (±1002.3)
G_out	408.6 (±67.2)	69.7 • (±19.4)	307.9 • (±100.4)	273.8 • (±92.6)	330.6 (±109.8)	298.6 • (±69.3)	353.4 (±79.6)
M_in	2004.9 (±404.3)	608.5 • (±222.8)	1600.8 • (±463.5)	1891.5 (±638.4)	1735.6 • (±398.7)	1884.5 (±429.7)	1912.8 (±378.5)
M_out	1498.6 (±625.4)	216.4 • (±77.3)	987.6 • (±368.5)	1032.7 • (±634.0)	862.2 • (±300.4)	1219.5 (±1014.3)	1467.5 (±837.2)
N_in	8640.5 (±3086.1)	942.0 • (±207.5)	3952.4 • (±1024.9)	5279.6 • (±1969.7)	6927.3 • (±2464.8)	5540.8 • (±2013.6)	**7817.4** (±3009.5)
N_out	4465.2 (±667.3)	1002.8 • (±335.2)	1137.1 • (±384.6)	2740.9 • (±511.5)	3298.5 • (±537.8)	2018.9 • (±685.4)	**3640.9** (±841.0)

Table 2: Average final scores on modified Cartpole (POMDP) with $N_{target} = 50$. The best non-oracle results are marked in **red**. G, M, and N denote the gravity, mass, and noise respectively.

	Oracle Upper bound	Non-t lower bound	PNN (Rusu et al., 2016)	PSM (Agarwal et al., 2021)	MTQ (Fakoor et al., 2020)	AdaRL* Ours w/o masks	AdaRL Ours
O_in	18.65 (±2.43)	6.18 • (±2.43)	9.70 • (±2.09)	11.61 • (±3.85)	15.79 • (±3.26)	14.27 • (±1.93)	**18.97** (±2.00)
O_out	19.86 (±1.09)	6.40 • (±3.17)	9.54 • (±2.78)	10.82 • (±3.29)	10.82 • (±4.13)	12.67 • (±2.49)	**15.75** (±3.80)
C_in	19.35 (±0.45)	8.53 • (±2.08)	14.44 • (±2.37)	19.02 (±1.17)	16.97 • (±2.02)	18.52 • (±1.41)	19.14 (±1.05)
C_out	19.78 (±0.25)	8.26 • (±3.45)	14.84 • (±1.98)	17.66 • (±2.46)	15.45 • (±3.30)	17.92 (±1.83)	19.03 (±0.97)
S_in	18.32 (±1.18)	6.91 • (±2.02)	11.80 • (±3.25)	12.65 • (±3.72)	13.68 • (±3.49)	14.23 • (±3.19)	**16.65** (±1.72)
S_out	19.01 (±1.04)	6.60 • (±3.11)	9.07 • (±4.58)	8.45 • (±4.51)	11.45 • (±2.46)	12.80 • (±2.62)	**17.82** (±2.35)
N_in	18.48 (±1.25)	5.51 • (±3.88)	12.73 • (±3.67)	11.30 • (±2.58)	12.67 • (±3.84)	13.78 • (±2.15)	**16.84** (±3.13)
N_out	18.26 (±1.11)	6.02 • (±3.19)	13.24 • (±2.55)	11.26 • (±3.15)	15.77 • (±2.12)	14.65 • (±3.01)	**18.30** (±2.24)

Table 3: Average final scores on modified Pong (POMDP) with $N_{target} = 50$. The best non-oracle are marked in **red**. O, C, S, and N denote the orientation, color, size, and noise factors, respectively.

the graphical structure, we also compare with $AdaRL*$, which does not learn the binary masks $c^{\cdot}$, but just assumes everything is fully connected. Learning the graphical structure improves the performances significantly, and without it $AdaRL*$ is generally comparable to baselines. We provide results with $N_{target} = \{20, 50, 10000\}$ in Appendix, showing that the performance gains are larger at smaller sample sizes. Furthermore, we consider the change of reward functions (see Table A12-14 in Appendix). More detailed experimental results are provided in Appendix, including the average score across different N_{target}, policy learning curves and an analysis of the estimated θ_k w.r.t. real change factor. Interestingly, in the Cartpole case, the estimated θ_k matches the physical quantities that are being changed across the domains. In particular, the estimated θ_k for gravity and noise are linear mappings of the gravity and noise level values. For the mass-varying case, the learned $\boldsymbol{\theta}_k^s$ is a nonlinear monotonic function of the mass, which matches the influence of the mass on the dynamics.

5 Conclusions and Future Work

In this paper, we proposed AdaRL, a principled framework for transfer RL. AdaRL learns a latent representation with domain-shared and domain-specific components across source domains, uses it to learn an optimal policy parameterized by the domain-specific parameters, and applies it to a new target domain. It is achieved without any further policy optimization, but just by estimating the values of the domain-specific parameters in the target domain, which can be accomplished with a few target-domain data. As opposed to previous work, AdaRL can model changes in the state dynamics, observation function and reward function in an unified manner, and exploit the factorization to improve the data efficiency and adapt faster with fewer samples. Further directions include exploiting the target domain to fine-tune the policy and handling the out-of-distribution data. Moreover, exploring an alternative to the reconstruction loss, e.g., using the contrastive loss (Srinivas et al., 2020), might also improve the training efficiency. Finally, an exciting next step is to transfer knowledge across different tasks, e.g., different Atari games.

REPRODUCIBILITY STATEMENT

Proofs of all our theoretical results are given in Appendix A1, A2, and A3 with disclosure of all assumptions. More details about the model estimation are given in Appendix A4. Appendix A5 provides the complete experimental details and results. All datasets used are publicly available or instructions are provided on how to generate them in Appendix 5 and 6. A description of the hyperparameters and network architectures used is included in Appendix 6. Source code for all our experiments are attached in the supplemental material and will be made available on GitHub after deanonymisation, providing also a complete description of our experimental environment, configuration files and instructions on the reproduction of our experiments.

REFERENCES

Rishabh Agarwal, Marlos C. Machado, Pablo Samuel Castro, and Marc G Bellemare. Contrastive behavioral similarity embeddings for generalization in reinforcement learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=qda7-sVg84.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47: 253–279, 2013.

C. Boutilier, R. Dearden, and M. Goldszmidt. Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121(1-2):49–107, 2000.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

W. J. Conover. Practical nonparametric statistics (3rd ed.). *John Wiley Sons, Inc.*, 1999.

Finale Doshi-Velez and George Konidaris. Hidden parameter markov decision processes: A semi-parametric regression approach for discovering latent task parametrizations. In *IJCAI: proceedings of the conference*, volume 2016, page 1432. NIH Public Access, 2016.

Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel. Rl ^{2}: Fast reinforcement learning via slow reinforcement learning. 2016.

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

F. Fernández, J. García, and M. Veloso. Probabilistic policy reuse for inter-task transfer learning. *Robotics and Autonomous Systems*, 58(7):866–871, 2010.

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, 2017.

A. Ghassami, N. Kiyavash, B. Huang, and K. Zhang. Multi-domain causal structure learning in linear systems. 2018.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

B. Huang, K. Zhang, J. Zhang, J. Ramsey, R. Sanchez-Romero, C. Glymour, and B. Schölkopf. Causal discovery from heterogeneous/nonstationary data. *JMLR*, 21(89):612–634, 2020.

Sammie Katt, Frans A. Oliehoek, and Christopher Amato. Bayesian reinforcement learning in factored pomdps. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '19, page 7–15, Richland, SC, 2019. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450363099.

Michael J. Kearns and Daphne Koller. Efficient reinforcement learning in factored mdps. pages 740–747, 1999. URL http://ijcai.org/Proceedings/99-2/Papers/013.pdf.

D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

David A McAllester. Pac-bayesian model averaging. In *Proceedings of the twelfth annual conference on Computational learning theory*, pages 164–170, 1999.

R. Mendonca, A. Gupta, R. Kralev, P. Abbeel, S. Levine, and C. Finn. Guided meta-policy search. 2019.

G. Mesnil, Y. Dauphin, X. Glorot, S. Rifai, Y. Bengio, I. Goodfellow, E. Lavoie, X. Muller, G. Desjardins, D. Warde-Farley, P. Vincent, A. Courville, and J. Bergstra. Unsupervised and transfer learning challenge: a deep learning approach. *JMLR W and CP: Proc. of the Unsupervised and Transfer Learning challenge and workshop*, 27, 2012.

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*, 518(7540):529–533, 2015.

V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, P. T. Lillicrap1, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. 2016.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*. 2013.

Kevin Murphy. Dynamic bayesian networks: Representation, inference and learning. *UC Berkeley, Computer Science Division*, 2002.

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. 2018.

Judea Pearl and Dana Mackenzie. *The Book of Why*. Basic Books, New York, 2018. ISBN 978-0-465-09760-9.

Xue Bin Peng, Erwin Coumans, Tingnan Zhang, Tsang-Wei Lee, Jie Tan, and Sergey Levine. Learning agile robotic locomotion skills by imitating animals. *arXiv preprint arXiv:2004.00784*, 2020.

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.

Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

B. Schölkopf. Causality for machine learning. *arXiv preprint arXiv:1911.10500*, 2019.

B. Schölkopf, F. Locatello, S. Bauer, N. R. Ke, N. Kalchbrenner, A. Goyal, and Y. Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel. High-dimensional continuous control using generalized advantage estimation. 2015.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Vedavyas Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy P. Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

A. Srinivas, M. Laskin, and P. Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *ICML*, 2020.

A. L. Strehl, C. Diuk, and M. L. Littman. Efficient structure learning in factored-state mdps. In *AAAI*, 2007.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.

A. Tamar, Y. Glassner, and S. Mannor. Optimizing the cvar via sampling. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

M. E. Taylor and P. Stone. Transfer learning for reinforcement learning domains: A survey. *JMLR*, 10(7), 2009.

M. E. Taylor, P. Stone, and Y. Liu. Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, 8(1):2125–2167, 2007.

A. Tirinzoni, A. Sessa, M. Pirotta, and M. Restelli. Importance weighted transfer of samples in reinforcement learning. In *International Conference on Machine Learning*, 2018.

A. Tirinzoni, M. Salvini, and M. Restelli. Transfer of samples in policy search via multiple importance sampling. In *International Conference on Machine Learning*, pages 6264–6274, 2019.

Manan Tomar, Amy Zhang, Roberto Calandra, Matthew E. Taylor, and Joelle Pineau. Model-invariant state abstractions for model-based reinforcement learning, 2021.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

Wenhao Yu, Jie Tan, C. Karen Liu, and Greg Turk. Preparing for the unknown: Learning a universal policy with online system identification. *arXiv preprint arXiv:1702.02453*, 2017.

A. Zhang, C. Lyle, S. Sodhani, A. Filos, M. Kwiatkowska, Y. Pineau, J.and Gal, and D. Precup. Invariant causal prediction for block mdps. *arXiv preprint arXiv:2003.06016*, 2020a.

Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. *ICLR*, 2021a.

Amy Zhang, Shagun Sodhani, Khimya Khetarpal, and Joelle Pineau. Learning robust state abstractions for hidden-parameter block {mdp}s. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=fmOOI2a3tQP.

J. Zhang and P. Spirtes. Intervention, determinism, and the causal minimality condition. *Synthese*, 182(3):335–347, 2011.

K. Zhang, M. Gong, P. Stojanov, B. Huang, Q. Liu, and C. Glymour. Domain adaptation as a problem of inference on graphical models. 2020b.

Z. Zhu, K. Lin, and J. Zhou. Transfer learning in deep reinforcement learning: A survey. 2020.

Luisa Zintgraf, Kyriacos Shiarli, Vitaly Kurin, Katja Hofmann, and Shimon Whiteson. Fast context adaptation via meta-learning. In *International Conference on Machine Learning*, pages 7693–7702. PMLR, 2019.