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

Appendix organization:

- Appendix A1: Proof of Proposition 1
- Appendix A2: Proof of Theorem 1
- Appendix A3: Proof of Theorem 2
- Appendix A4: More details for model estimation
- Appendix A5: Complete experimental results
- Appendix A6: Experimental details

A1 PROOF OF PROPOSITION 1

We first review the definitions of d-separation, the Markov condition, and the faithfulness assumption (Spirtes et al., 1993; Pearl, 2000), which will be used in the proof.

Given a directed acyclic graph $G = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V}$ is the set of nodes and $\mathbf{E}$ is the set of directed edges, we can define a graphical criterion that expresses a set of conditions on the paths.

Definition A1 (d-separation (Pearl, 2000))**.** *A path p is said to be d-separated by a set of nodes $\boldsymbol{Z} \subseteq \boldsymbol{V}$ if and only if (1) p contains a chain $i \to m \to j$ or a fork $i \leftarrow m \to j$ such that the middle node m is in Z, or (2) p contains a collider $i \to m \leftarrow j$ such that the middle node m is not in $\boldsymbol{Z}$ and such that no descendant of m is in $\boldsymbol{Z}$.*

Let $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{Z}$ be disjunct sets of nodes. $\boldsymbol{Z}$ is said to d-separate $\boldsymbol{X}$ from $\boldsymbol{Y}$ (denoted as $\boldsymbol{X} \perp_d \boldsymbol{Y}|\boldsymbol{Z}$) if and only if $\boldsymbol{Z}$ blocks every path from a node in $\boldsymbol{X}$ to a node in $\boldsymbol{Y}$.

Definition A2 (Global Markov Condition (Spirtes et al., 1993; Pearl, 2000))**.** *A distribution P over $\mathbf{V}$ satisfies the global Markov condition on graph G if for any partition $(\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{Y})$ such that $\boldsymbol{X} \perp_d \boldsymbol{Y}|\boldsymbol{Z}$*

$$P(\boldsymbol{X}, \boldsymbol{Y}|\boldsymbol{Z}) = P(\boldsymbol{X}|\boldsymbol{Z})P(\boldsymbol{Y}|\boldsymbol{Z}).$$

In other words, $\boldsymbol{X}$ is conditionally independent of $\boldsymbol{Y}$ given $\boldsymbol{Z}$, which we denote as $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Y}|\boldsymbol{Z}$.

Definition A3 (Faithfulness Assumption (Spirtes et al., 1993; Pearl, 2000))**.** *There are no independencies between variables that are not entailed by the Markov Condition.*

If we assume both of these assumptions, then we can use d-separation as a criterion to read all of the conditional independences from a given DAG G. In particular, for any disjoint subset of nodes $\mathbf{X},\mathbf{Y},\mathbf{Z} \subseteq \mathbf{V}$: $\mathbf{X} \perp\!\!\!\perp \mathbf{Y}|\mathbf{Z} \iff \mathbf{X} \perp_d \mathbf{Y}|\mathbf{Z}$.

In our case we can represent the generative model in Eq. 1 as a Dynamic Bayesian Network(DBN) $\mathcal{G}$ (Murphy, 2002) over the variables $\{\mathbf{s}_{t-1}, a_{t-1}, o_{t-1}, r_t, \mathbf{s}_t, \boldsymbol{\theta}_k\}$, where the binary masks $c^{\cdot\to\cdot}$ represent edges or sets of edges, as shown in Fig. A1. As is typical in DBN we assume that the graph is invariant across different timesteps. We assume $\boldsymbol{\theta}_k$ are constant across the different timesteps. We add to the image also the cumulative reward R. In practice we will focus instead on the cumulative *future* reward $R_{t+1} = \sum_{\tau=t+1}^{T} \gamma^{\tau-t-1} r_\tau$, which only considers the contributions of the r_τ in the future with respect to the current timestep t.

In order to prove Proposition 1, we first need to prove that the compact shared representations $\mathbf{s}_t^{min}$ and compact shared representations $\boldsymbol{\theta}_k^{min}$ are all the state and change factors dimensions, respectively, that are conditionally independent of a_t given the future cumulative reward R_{t+1}, even given all other variables:

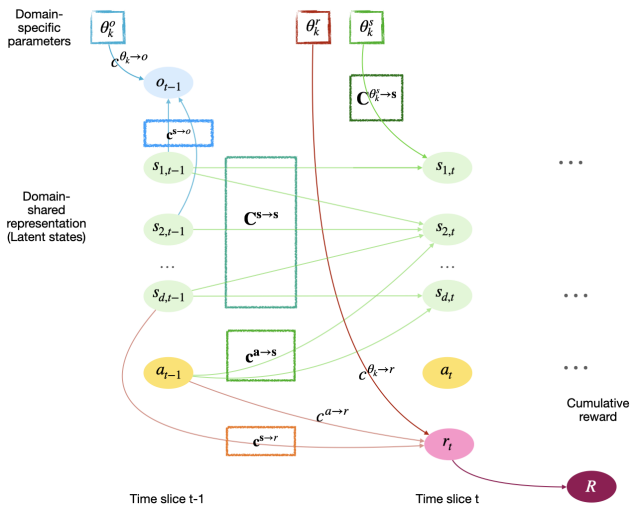


Figure A1: Graphical representation of the generative model in Eq. 1. In the top of the figure, the square boxes are the domain-specific parameters $\boldsymbol{\theta}_k$, which are constant in time, while the rectangular boxes represent the binary masks that encode the edges. R represents the cumulative reward

Lemma A1. *Under the assumption that the graph $\mathcal{G}$ is Markov and faithful to the measured data, a state dimension $s_{i,t} \in \mathbf{s}_t$ is part of $\mathbf{s}_t^{min}$ iff:*

$$s_{i,t} \not\perp\!\!\!\perp a_t | R_{t+1}, \tilde{s}_t \;\; \forall \tilde{s}_t \subseteq \{\mathbf{s}_t \setminus s_{i,t}\},$$

Similarly, a change factor dimension $\theta_{i,k} \in \theta_k$ is part of $\boldsymbol{\theta}_k^{min}$ iff:

$$\theta_{i,k} \not\perp\!\!\!\perp a_t | R_{t+1}, \tilde{l}_t, \;\; \forall \tilde{l}_t \subseteq \{\mathbf{s}_t, \{\boldsymbol{\theta}_k \setminus \theta_{i,k}\}\}.$$

Proof. We split the proof in two parts, the "only if" and the "if" part:

"If conditionally dependent on a_t given R_{t+1} then in compact representation ":

We first show that if $s_{i,t}$ satisfies the conditional dependence $s_{i,t} \not\perp\!\!\!\perp a_t | R_{t+1}, \tilde{s}_t, \;\; \forall \tilde{s}_t \subseteq \{\mathbf{s}_t \setminus s_{i,t}\}$, then it is part of $\mathbf{s}_t^{min}$, i.e. $s_{i,t}$ either has an edge to the reward in the next time-step r_{t+1}, or, recursively, it has an edge to another state component in the next time-step $s_{j,t+1}$, such that the same component at time step t, $s_{j,t+1} \in \mathbf{s}_t^{min}$. Note that this recursive definition collects all of the states $s_{i,t}$ that have an effect on future reward $r_{t+\tau}, \tau = \{1, \dots, T-t\}$, either directly, or through the influence on other state components. Since all of these $r_{t+\tau}$ are influencing the cumulative future reward R_{t+1}, all of these components have an edge to R_{t+1} as well. We prove it by contradiction. Suppose that $s_{i,t} \not\perp\!\!\!\perp a_t | R_{t+1}$ does not have a direct or indirect path to $r_{t+\tau}$, i.e. $s_{i,t} \in \mathbf{s}^{min}$. By assumption a_t only affects future state $\mathbf{s}_{t+1}$, so $s_{i,t} \perp\!\!\!\perp a_t$. Then, according to the Markov and faithfulness conditions, $s_{i,t}$ is independent of a_t conditioning on R_{t+1}, since there is no path that connects $s_{i,t} \rightarrow \dots \rightarrow r_{t+\tau} \rightarrow R_{t+1} \leftarrow r_{t+1} \leftarrow a_t$ on which R_{t+1} is a collider (i.e. a variable with two incoming edges), which is the only path which could introduce a conditional dependence. This contradicts the assumption.

Similarly we show that if $\forall \tilde{l}_t \subseteq \{\mathbf{s}_t, \{\boldsymbol{\theta}_k \setminus \theta_{i,k}\}\}$ the change factor dimension $\theta_{i,k} \not\perp\!\!\!\perp a_t | R_{t+1}, \tilde{l}_t$, then $\theta_{i,k} \in \boldsymbol{\theta}_k^{min}$, which similarly to previous case means it has a direct or indirect effect on $r_{t+\tau}$ and therefore R_{t+1}. By contradiction suppose that $\theta_{i,k} \not\perp\!\!\!\perp a_t | R_{t+1}, \tilde{l}_t$ for all previously defined $\tilde{l}_t$, but it is not a change parameter for the reward function $\theta_{i,k} \notin \theta_k^r$ with $c^{\theta_k \rightarrow r} = 1$, nor it is a change parameter for the state dynamics $\theta_{i,k} \in \theta_k^s$ with a direct or indirect path to $r_{t+\tau}$ for $\tau = 1, \dots, T-t$. By assumption of our model, $\theta_{i,k}$ is never connected to a_t directly, nor they might have a common cause, so $\theta_{i,k} \perp\!\!\!\perp a_t$. Then, according to the Markov condition, $\theta_{i,k}$ is independent of a_t conditioning on R_{t+1}, which contradicts the assumption, since:

- if $\theta_{i,k} \notin \theta_k^r$ or $c^{\theta_k \rightarrow r} = 0$, then it means there is no path $\theta_{i,k} \rightarrow s_{i,t+\tau} \rightarrow r_{t+\tau} \rightarrow R_{t+1} \leftarrow r_{t+1} \leftarrow a_t$ for any $\tau \in \mathbb{N}$ that would be open by conditioning on R_{t+1};

- if $\theta_{i,k} \in \theta_k^s$ but there is no directed path to $r_{t+\tau}$ for any $\tau \geq 1$, i.e. then there is also no directed path $\theta_{i,k} \rightarrow \cdots \rightarrow r_{t+\tau} \rightarrow R_{t+1} \leftarrow r_{t+1} \leftarrow a_t$ that would be open when we condition on R_{t+1}.

"If in compact representation then conditionally dependent on a_t given R_{t+1}":

We next show that if $s_{i,t} \in \mathbf{s}_t^{min}$, which mean $s_{i,t}$ has a direct or indirect edge to $r_{t+\tau}$, $\tau = \{1, \ldots, T-t\}$, then $s_{i,t}$ satisfies the conditional dependence $s_{i,t} \not\perp\!\!\!\perp a_t | R_{t+1}, \tilde{s}_t, \ \forall \tilde{s}_t \subseteq \{\mathbf{s}_t \setminus s_{i,t}\}$. We prove it by contradiction. Suppose $s_{i,t}$ has a directed path to $r_{t+\tau}$ and $s_{i,t}$ is independent on a_t given R_{t+1} and a subset of other variables $\tilde{s}_t \subseteq \mathbf{s}_t \setminus s_{i,t}$. Since we assume that there are no instantaneous causal relations across the state dimensions, if $s_{i,t} \not\perp\!\!\!\perp a_t | R_{t+1}$ there can never be an $s_{j,t}$ such that $s_{i,t} \perp\!\!\!\perp a_t | R_{t+1}, s_{j,t}$. In this case, this means that $s_{i,t} \perp\!\!\!\perp a_t | R_{t+1}$ has to hold. Then according to the Markov and faithfulness assumptions, $s_{i,t}$ cannot have any directed path to any $r_{t+\tau} \forall \tau \geq 1$, because that any such path create a v-structure in the collider R_{t+1}, which would be open if we condition on R_{t+1}, contradicting the assumption.

Similarly, suppose $\theta_{i,k}$ is a dimension in $\boldsymbol{\theta}_k$ that has a directed path to $r_{t+\tau}$. We distinguish two cases, and show in neither can $\theta_{i,k}$ be independent of a_t given R_{t+1} and a subset of the other variables:

- if $\theta_{i,k} \in \theta_k^r$, then it cannot be independent of a_t when we condition on R_{t+1}, which is a descendant of r_{t+1} and therefore opens the collider path $\theta_k^r \leftarrow r_{t+1} \leftarrow a_t$;
- if $\theta_{i,k} \in \theta_k^s$, then at timestep t it is always only connected to the corresponding $s_{i,t}$. So if there is a directed path π to $r_{t+\tau}$, it has to go through $s_{i,t}$. While π cannot be blocked by any subset of $\{\boldsymbol{\theta}_k \setminus \theta_{i,k}\}$, it can be blocked by conditioning on $s_{i,t}$, there are infinite future paths with the same structure, e.g. through $s_{i,t+1}$ that will not be blocked by conditioning only on variables at timestep t. Under the faithfulness and Markov assumption this means that $\theta_{i,k}$ cannot be independent from a_t by conditioning on any subset of state variables at timestep t or any other change parameters, which is a contradiction.

□

We can now prove our main proposition:

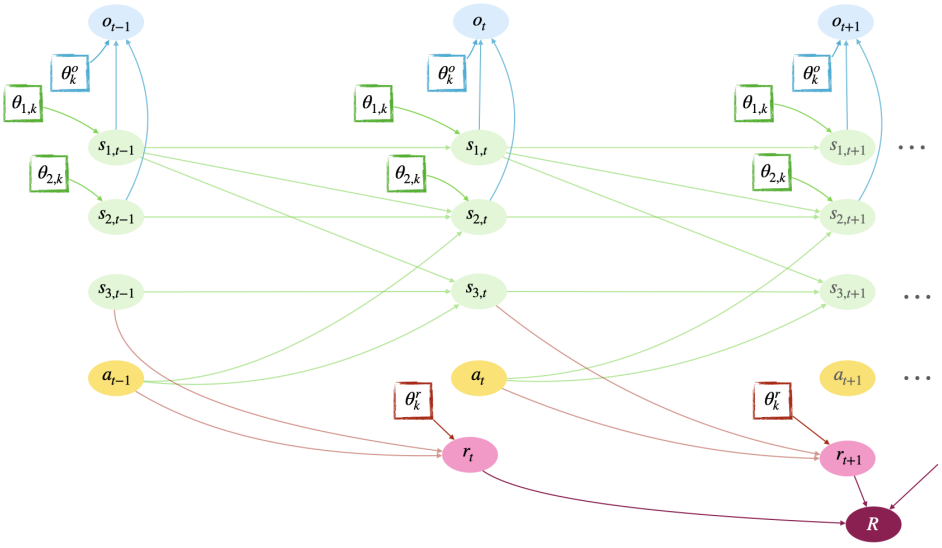
Proposition 1. *Under the assumption that the graph $\mathcal{G}$ is Markov and faithful to the measured data, the union of compact domain-specific $\boldsymbol{\theta}_k^{min}$ and compact shared representations $\boldsymbol{s}_t^{min}$ are the minimal and sufficient dimensions for policy learning across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

Proof. As shown in the previous lemma, in *compact domain-generalizable representations* $\mathbf{s}_t^{min}$ every dimension is dependent on a_t given R_{t+1} and any other variables, and every other dimension is independent of a_t given R_{t+1} and some other variables. Furthermore, because every dimension that is dependent on a_t is necessary for the policy learning and every dimension that is (conditionally) independent of a_t for at least a subset of other variables is not necessary for the policy learning, compact domain-specific $\boldsymbol{\theta}_k^{min}$ and compact shared representations $\mathbf{s}_t^{min}$ contain minimal and sufficient dimensions for policy learning across domains. Note that the agents determine the action under the condition of maximizing cumulative reward, which policy learning aims to achieve, so we always consider the situation when the discounted cumulative future reward R_{t+1} is given. □

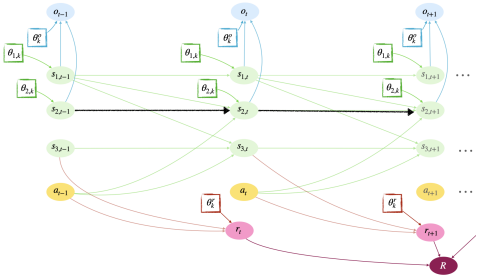
A2 Proof of Theorem 1

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 .*

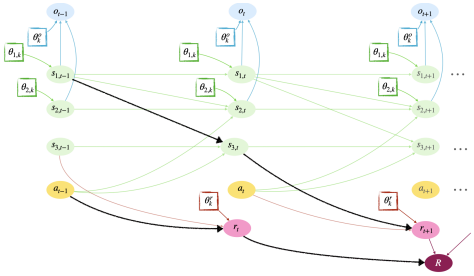
Proof. We concatenate data from different domains and denote by k be the variable that takes the domain index $1, \cdots, n$. Since the data distribution changes across domains and the change is due to the unobserved change factors $\boldsymbol{\theta}_k$ that influence the observed variables, we can represent the change factors as a function of k. In other words, we use the domain index k as a surrogate variable to characterize the unobserved change factors.



(a) An example of a ground Bayesian network (unrolled DBN over time).



(b) $s_{2,t}$ is not a compact domain-specific representation, since there is no directed path to any $r_{t+\tau}$, i.e. $s_{2,t} \perp\!\!\!\perp a_t | R$. Similarly, there is no directed path from $\theta_{1,k}$ to R.



(c) $s_{1,t}$ is a compact domain-specific representation, since there exists a path to a_t that is d-connected when we condition on the collider R. Similarly $\theta_{1,k}$ is d-connected to a_t when conditioning on R.

Figure A2: Example model in which $s_{1,t}$ and $s_{3,t}$ are compact domain-specific representations for policy learning, while $s_{2,t}$ is not. This does not mean that $s_{2,t}$ and $\theta_{2,k}$ are not useful in the model estimation part, especially in estimating θ^o_k.

We denote the variable set in the system by $\mathbf{V}$, with $\mathbf{V} = \{s_{1,t-1}, \ldots, s_{d,t-1}, s_{1,t}, \ldots, s_{d,t}, a_{t-1}, r_t\}$, and the variables form a dynamic Bayesian network $\mathcal{G}$. Note that in our particular problem, according to the generative environment model in Eq. (1), the possible edges in $\mathcal{G}$ are only those from $s_{i,t-1} \in \mathbf{s}_{t-1}$ to $s_{j,t} \in \mathbf{s}_t$, from a_{t-1} to $s_{j,t} \in \mathbf{s}_t$, from $s_{i,t-1} \in \mathbf{s}_{t-1}$ to r_t, and from a_{t-1} to r_t. We further include the domain index k into the system to characterize the unobserved change factors.

It has been shown that under the Markov condition and faithfulness assumption, for every $V_i, V_j \in \mathbf{V}$, V_i and V_j are not adjacent in $\mathcal{G}$ if and only if they are independent conditional on some subset of $\{V_l | l \neq i, l \neq j\} \cup k$ (Huang et al., 2020). Thus, we can asymptotically identity the correct graph skeleton over $\mathbf{V}$.

Moreover, since we assume a dynamic Bayesian network, there the direction of an edge between a variable at time t to one at time $t+1$ is fixed. Therefore, the structural matrices $\mathbf{C}^{\mathbf{s}\to\mathbf{s}}$, $\mathbf{c}^{a\to\mathbf{s}}$, $\mathbf{c}^{\mathbf{s}\to r}$, and $c^{a\to r}$, which are parts of the graph $\mathcal{G}$ over $\mathbf{V}$, are identifiable.

Furthermore, we want to show the identifiability of $\mathbf{C}^{\theta_k\to\mathbf{s}}$, and $c^{\theta_k\to r}$; that is, to identify which distribution modules have changes. Whether a variable V_i has a changing module is decided by whether V_i and k are independent conditional on some subset of other variables. The justification for one side of this decision is trivial. If V_i's module does not change, that means $P(V_i \mid \mathrm{PA}^i)$ remains the same for every value of k, and so $V_i \perp\!\!\!\perp k \mid \mathrm{PA}^i$. Thus, if V_i and k are not independent conditional on any subset of other variables, V_i's module changes with k, which is represented by an edge between V_i and k. Conversely, we assume that if V_i's module changes, which entails that V_i and k are not independent given PA^i, then V_i and k are not independent given any other subset of $\mathbf{V}\backslash\{V_i\}$. If this assumption does not hold, then we only claim to detect some (but not necessarily all) variables with changing modules. □

A3 Proof of Theorem 2

In this section, we derive the generalization bound under the PAC-Bayes framework (McAllester, 1999; Shalev-Shwartz and Ben-David, 2014), and our formulation follows (Pentina and Lampert, 2014) and (Amit and Meir, 2018). We assume that all domains share the sample space $\mathcal{Z}$, hypothesis space $\mathcal{H}$, and loss function $\ell : \mathcal{H} \times \mathcal{Z} \to [0, 1]$. All domains differ in the unknown sample distribution E_k parameterized by $\boldsymbol{\theta}_k^{\min}$ associated with each domain k. We observe the training sets $S_1, \ldots, S_n$ corresponding to n different domains. The number of samples in domain k is denoted by m_k. Each dataset S_k is assumed to be generated from an unknown sample distribution $E_k^{m_k}$. We also assume that the sample distribution E_k are generated $i.i.d.$ from an unknown domain distribution τ. More specifically, we have $S_k = (z_{1,k}, \ldots, z_{i,k}, \ldots, z_{m_k,k})$, where $z_{i,k} = (\mathbf{s}_{i,k}, v^*(\mathbf{s}_{i,k}))$. Note that, $\mathbf{s}_{i,k}$ is the i-th state sampled from k-th domain and $v^*(\mathbf{s}_{i,k})$ 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 $\ell(h_{\boldsymbol{\theta}_k^{\min}}, z_{i,k}) = D_{dist}(h_{\boldsymbol{\theta}_k^{\min}}(\mathbf{s}_{i,k}), v^*(\mathbf{s}_{i,k}))$, where D_{dist} is a distance function that measures the discrepancy between the learned value and the optimal state-value. We also let P be the prior distribution over $\mathcal{H}$ and Q the posterior over $\mathcal{H}$.

Theorem 1. *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 $\mathcal{Q}$,*

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

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.

Proof. This proof consists of two steps, both using the classical PAC-Bayes bound (McAllester, 1999; Shalev-Shwartz and Ben-David, 2014). Therefore, we start by restating the classical PCA-Bayes bound.

Theorem A1 (Classical PAC-Bayes Bound, General Notations)**.** *Let $\mathcal{X}$ be a sample space, $P(X)$ a distribution over $\mathcal{X}$, Θ a hypothesis space. Given a loss function $\ell(\theta, X) : \Theta \times \mathcal{X} \to [0, 1]$ and a collection of M i.i.d random variables $(X_1, \ldots, X_M)$ sampled from $P(X)$, let π be a prior distribution over hypothesis in Θ. Then, for any $\delta \in (0, 1]$, the following bound holds uniformly for all posterior distributions ρ over Θ,*

$$P\left(\mathop{\mathbb{E}}_{X_i\sim P(X),\theta\sim\rho}[\ell(\theta, X_i)] \leq \frac{1}{M}\sum_{m=1}^{M}\mathop{\mathbb{E}}_{\theta\sim\rho}[\ell(\theta, X_m)] + \sqrt{\frac{1}{2(M-1)}\left(D_{KL}(\rho||\pi) + \log\frac{M}{\delta}\right)}, \forall\rho\right) \geq 1-\delta.$$

Between-domain Generalization Bound First, we bound the between-domain generalization, i.e., relating $er(\mathcal{Q})$ to $er(\mathcal{Q}, E_k)$.

We first expand the generalization error as below,

$$\begin{aligned} er(\mathcal{Q}) &= \mathop{\mathbb{E}}_{(E,m)\sim\tau}\ \mathop{\mathbb{E}}_{S\sim E^m}\ \mathop{\mathbb{E}}_{\boldsymbol{\theta}\sim\mathcal{Q}}\ \mathop{\mathbb{E}}_{h\sim Q(S,\boldsymbol{\theta})}\ \mathop{\mathbb{E}}_{z\sim E}\ell(h, z) \\ &= \mathop{\mathbb{E}}_{(E,m)\sim\tau}\ \mathop{\mathbb{E}}_{S\sim E^m}\ \mathop{\mathbb{E}}_{\boldsymbol{\theta}\sim\mathcal{Q}}\ell(\boldsymbol{\theta}, E) \\ &= \mathop{\mathbb{E}}_{(E,m)\sim\tau}\ \mathop{\mathbb{E}}_{S\sim E^m} er(\mathcal{Q}, E). \end{aligned} \tag{A1}$$

Then we compute the error across the training domains,

$$\frac{1}{n}\sum_{k=1}^{n}\mathop{\mathbb{E}}_{\boldsymbol{\theta}\sim\mathcal{Q}}\ \mathop{\mathbb{E}}_{h\sim Q(S_k,\boldsymbol{\theta})}\ \mathop{\mathbb{E}}_{z\sim E_k}\ell(h, z) = \frac{1}{n}\sum_{k=1}^{n} er(\mathcal{Q}, E_k). \tag{A2}$$

Then Theorem A1 says that for any $\delta_0 \sim (0, 1]$, we have

$$P\left(er(\mathcal{Q}) \leq \frac{1}{n}\sum_{k=1}^{n} er(\mathcal{Q}, E_k) + \sqrt{\frac{1}{2(n-1)}\left(D_{KL}(\mathcal{Q}||\mathcal{P}) + \log\frac{n}{\delta_0}\right)}\right) \geq 1-\delta_0, \tag{A3}$$

where $\mathcal{P}$ is a prior distribution over $\boldsymbol{\theta}$.

Within-domain Generalization Bound Then, we bound the the within-domain generalization, i.e., relating $er(\mathcal{Q}, E_k)$ to $\hat{er}(\mathcal{Q}, S_k)$.

We first have

$$er(\mathcal{Q}, E_k) = \mathop{\mathbb{E}}_{\boldsymbol{\theta}\sim\mathcal{Q}}\ \mathop{\mathbb{E}}_{h\sim Q(S_k,\boldsymbol{\theta})}\ \mathop{\mathbb{E}}_{z\sim E_k}\ell(h, z). \tag{A4}$$

Then we compute the empirical error across the training domains,

$$\hat{er}(\mathcal{Q}, S_k) = \frac{1}{m_k}\sum_{j=1}^{m_k}\mathop{\mathbb{E}}_{h\sim Q(S_k,\boldsymbol{\theta})}\ \mathop{\mathbb{E}}_{z\sim E_k}\ell(h, z_{i,j}). \tag{A5}$$

According to Theorem A1, for any $\delta_D \sim (0, 1]$, we have

$$P\left(er(\mathcal{Q}, E_k) \leq \hat{er}(\mathcal{Q}, S_k) + \sqrt{\frac{1}{2(m_k-1)}\left(D_{KL}(\rho||\pi) + \log\frac{m_k}{\delta_k}\right)}\right) \geq 1-\delta_k. \tag{A6}$$

With the choice of $\pi = \int\mathcal{P}(\boldsymbol{\theta})Q(S_D, \boldsymbol{\theta})d\boldsymbol{\theta}$ and $\rho = \int\mathcal{Q}(\boldsymbol{\theta})Q(S_D, \boldsymbol{\theta})d\boldsymbol{\theta}$, we have that $D_{KL}(\rho||\pi) \leq D_{KL}(\mathcal{Q}||\mathcal{P})$ (Yin et al., 2019). Thus, the above inequality can be further written as,

$$P\left(er(\mathcal{Q}, E_k) \leq \hat{er}(\mathcal{Q}, S_k) + \sqrt{\frac{1}{2(m_k-1)}\left(D_{KL}(\mathcal{Q}||\mathcal{P}) + \log\frac{m_k}{\delta_k}\right)}\right) \geq 1-\delta_k. \tag{A7}$$

Overall Generalization Bound Combining Eq. (A3) and (A7) using the union bound and choosing that for any $\delta > 0$, set $\delta_0 \doteq \frac{\delta}{2}$ and $\delta_k \doteq \frac{\delta}{2n}$ for $k = 1, \ldots, n$, then we finally obtain,

$$P\left(er(\mathcal{Q}) \leq \frac{1}{n}\sum_{k=1}^{n} \hat{er}(\mathcal{Q}, S_k) + \frac{1}{n}\sum_{k=1}^{n}\sqrt{\frac{1}{2(m_k-1)}\left(D_{KL}(\mathcal{Q}||\mathcal{P}) + \log\frac{2nm_k}{\delta}\right)} + \sqrt{\frac{1}{2(n-1)}\left(D_{KL}(\mathcal{Q}||\mathcal{P}) + \log\frac{2n}{\delta}\right)}\right) \geq 1-\delta. \tag{A8}$$

□

A4 More details for model estimation

A4.1 Locating model changes

In real-world scenarios, it is often the case that changes to the environment are sparse and localized. Instead of assuming every function to change arbitrarily, which is inefficient and unnecessarily complex, we first identify possible locations of the changes. To this end, we concatenate data from different domains and denote by k the variable that takes distinct values $1, \cdots, n$ to represent the domain index. Then, we exploit (conditional) independencies/dependencies to locate model changes. These (conditional) independencies/dependencies can be tested by kernel-based conditional independence tests (Zhang et al., 2011), which allows for both linear or nonlinear relationships between variables. Below we show that in some cases, we can identify the location of $\boldsymbol{\theta}_k$, by using the conditional independence relationships from concatenated data.

Proposition A1. *In POMDP, where the underlying states are latent, we can localize the changes by conditional independence relationships from concatenated observed data $\mathcal{D}$ in the following cases:*

- C_1: *if $o_t \perp\!\!\!\perp k$, then there is neither a change in the observation function nor in the state dynamics for any state that is an input to the observation function;*
- C_2: *if $o_t \perp\!\!\!\perp k$ and $a_t \not\perp\!\!\!\perp k | r_{t+1}$, then there is only a change in the reward function;*
- C_3: *if $a_t \perp\!\!\!\perp k | r_{t+1}$, then there is neither a change in the reward function nor in the state dynamics for any state in $\mathbf{s}^{min}$;*
- C_4: *if $a_t \perp\!\!\!\perp k | r_{t+1}$ and $o_t \not\perp\!\!\!\perp k$, then there is a change in the observation function, or there exists a state that is not in $\mathbf{s}^{min}$ but is an input to the observation function, whose dynamics has a change.*

Proof. We first formulate the problem as follows. We concatenate data from different domains and use domain-index variable k to indicate whether the corresponding distribution module has changes across domains. Specifically, by assuming the Markov condition and faithfulness assumption, $s_{i,t}$ has an edge with k if and only if $p(s_{i,t}|PA(s_{i,t}))$ changes across domains, where $PA(\cdot)$ denotes its parents. Similarly, r_t has an edge with k if and only if $p(r_t|PA(r_t))$ changes across domains, and o_t has an edge with k if and only if $p(o_t|PA(o_t))$ changes across domains. Under this setting, locating changes is equivalent to identify which variables have an edge with k from the data.

We consider the scenario of POMDP, where we only observe $\{o_t, r_t, a_t\}$ and the underlying states $\mathbf{s}_t$ are not observed. Below, we consider each case separately.

Case 1: Show that if $o_t \perp\!\!\!\perp k$, then there is neither a change in the observation function nor a change in the state dynamics for any state that is an input to the observation function.

We prove it by contradiction. Suppose that there is a change in the observation function and a change in the state dynamics for any state that is an input to the observation function. That is, o_t has an edge with k, and $s_{i,t}$ that has a direct edge to o_t also connects with k. Based on faithfulness assumption, $o_t \not\perp\!\!\!\perp k$, which contradicts to the assumption. Since we have a contradiction, it must be that there is neither a change in the observation function nor a change in the state dynamics for any state that is an input to the observation function.

Case 2: Show that if $o_t \perp\!\!\!\perp k$ and $a_t \not\perp\!\!\!\perp k | r_{t+1}$, then there is only a change in the reward function.

If $o_t \perp\!\!\!\perp k$ and $a_{t-1} \not\perp\!\!\!\perp k | r_t$, based on the Markov condition and faithfulness assumption, r_t has an edge with k, and $s_{i,t}$ and o_t do not have edges with k; that is, there are only changes in the reward function.

Case 3: Show that if $a_t \perp\!\!\!\perp k | r_{t+1}$, then there is neither a change in the reward function nor a change in the state dynamics for any state in $\mathbf{s}^{min}$.

By contradiction, suppose that there is a change in the reward function or there exists a state $s_{j,t} \in \mathbf{s}_j^{min}$ that has a change in its dynamics. That is, r_t has an edge with k or corresponding $s_{j,t}$ has an edge with k. Based on faithfulness assumption, $a_t \not\perp\!\!\!\perp k | r_{t+1}$, which contradicts to the assumption.

Case 4: Show that if $a_t \perp\!\!\!\perp k | r_{t+1}$ and $o_t \not\perp\!\!\!\perp k$, then there is a change in the observation function, or there exists a state that is not in $\mathbf{s}^{min}$ but is an input to the observation function, whose dynamics has a change.

According to Case 3, if $a_t \perp\!\!\!\perp k | r_{t+1}$, then there is neither a change in the reward function nor a change in the state dynamics for any state in $\mathbf{s}^{min}$. Furthermore, since $o_t \not\perp\!\!\!\perp k$, then based on the Markov condition, either o_t has an edge with k, or there exists a state that is not in $\mathbf{s}^{min}$ but is an input to the observation function, whose dynamics has an edge with k. That is, there is a change in the observation function, or there exists a state that is not in $\mathbf{s}^{min}$ but is an input to the observation function, whose dynamics has a change.

□

Based on Theorem 1 and Proposition A1, in MDP, we can fully determine where the changes are, so we only need to consider the corresponding $\theta_k^{(\cdot)}$ to capture the changes. In POMDP, in Case 1, we only need to involve θ_k^s and θ_k^r in model estimation, that is, $\boldsymbol{\theta}_k = \{\theta_k^s, \theta_k^r\}$; in Case 2, $\boldsymbol{\theta}_k = \{\theta_k^r\}$; and in Case 3 & 4, $\boldsymbol{\theta}_k = \{\theta_k^o, \theta_k^s\}$. For other cases, we involve $\boldsymbol{\theta}_k = \{\theta_k^o, \theta_k^s, \theta_k^r\}$ in model estimation.

A4.2 More details for estimation of domain-varying models in one step

We use MiSS-VAE to learn the environment model, which contains three components: the "sequential VAE" component, the "multi-model" component, and the "structure" component. Figure A3 gives the diagram of neural network architecture in model training.

Specifically, for the "sequential VAE" component, we include a Long Short-Term Memory (LSTM (Hochreiter and Schmidhuber, 1997)) to encode the sequential information with output h_t and a Mixture Density Network (MDN (Bishop, 1994)) to output the parameters of MoGs, and thus to learn the inference model $q_\phi(\mathbf{s}_{t,k}|\mathbf{s}_{t-1,k}, \mathbf{y}_{1:t,k}, a_{1:t-1,k}; \boldsymbol{\theta}_k)$ and infer a sample of $\mathbf{s}_{t,k}$ from q_ϕ as the output. The generated sample further acts as an input to the decoder, and the decoder outputs $\hat{o}_{t+1}$ and $\hat{r}_{t+2}$. Moreover, the state dynamics which satisfies a Markov process is modeled with an MLP and MDN.

For the "multi-model" component, we include the domain index k as an input to LSTM and involve $\boldsymbol{\theta}_k$ as free parameters in the inference model q_ϕ, by assuming that $\boldsymbol{\theta}_k$ also characterizes the changes in the inference model. Moreover, we embed θ_k^s in state dynamics p_γ, θ_k^o in observation function and θ_k^r in reward function in the decoder. With such a characterization, except $\boldsymbol{\theta}_k$, all other parameters 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–usually few samples are needed.

For the "structure" component, the latent states are organized with structures, captured by the mask $\mathbf{C}^{\mathbf{s}\to\mathbf{s}}$. Also, the structural relationships among perceived signals, latent states, the action variable, the reward variable, and the domain-specific factors are embedded as free parameters (structural vectors and scalars $\mathbf{c}^{\mathbf{s}\to\mathbf{s}}$, $c^{a\to\mathbf{s}}$, $\mathbf{c}^{\theta_k\to\mathbf{s}}$, $\mathbf{c}^{\mathbf{s}\to r}$, and $c^{a\to r}$) into MiSS-VAE.

A5 Complete experimental results

In this section we provide the complete experimental results on both of our settings, modified Cartpole and modified Pong in the OpenAI Gym (Brockman et al., 2016). In the POMDP setting, we use images as input, which for Cartpole look like Fig. A4(a) and for Pong look like Fig. A8(a). For

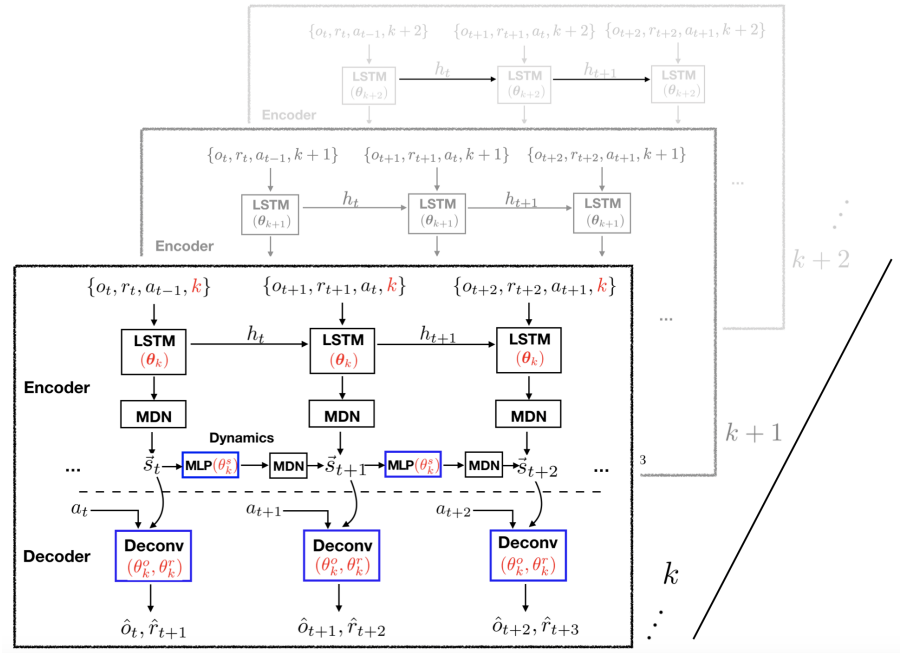


Figure A3: Diagram of MiSS-VAE neural network architecture. The "sequential VAE" component, "multi-model" component, and "structure" component are marked with black, red, and blue, respectively.

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.

We consider changes in the state dynamics (e.g., the change of gravity or cart mass in Cartpole, or the change of orientation in Atari), changes in perceived signals (e.g., different noise levels on observed images in Cartpole, as shown in Fig. A4 or 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. A8 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 experiments, 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.

A5.1 Complete results of modified Cartpole experiment

The Cartpole problem consists of a cart and a vertical pendulum attached to the cart using a passive pivot joint. The cart can move left or right. The task is to prevent the vertical pendulum from falling by putting a force on the cart to move it left or right. The action space consists of two actions: moving left or right.

We introduce two change factors for the state dynamics θ_k^s: varying gravity and varying mass of the cart, and a change factor in the observation function θ_k^o that is the image noise level. Fig. A4 gives a visual example of Cartpole game, and the image with Gaussian noise. The images of the varying gravity and mass look exactly like the original image. Specifically, in the gravity case, we consider source domains with gravity $g = \{5, 10, 20, 30, 40\}$. We take into account both interpolation (where the gravity in the target domain is in the support of that in source domains) with $g = \{15\}$, and extrapolation (where it is out of the support w.r.t. the source domains) with $g = \{55\}$. Similarly, we consider source domains where the mass of the cart is $m = \{0.5, 1.5, 2.5, 3.5, 4.5\}$, while in target domains it is $m = \{1.0, 5.5\}$. In terms of changes on the observation function θ_k^o, we add

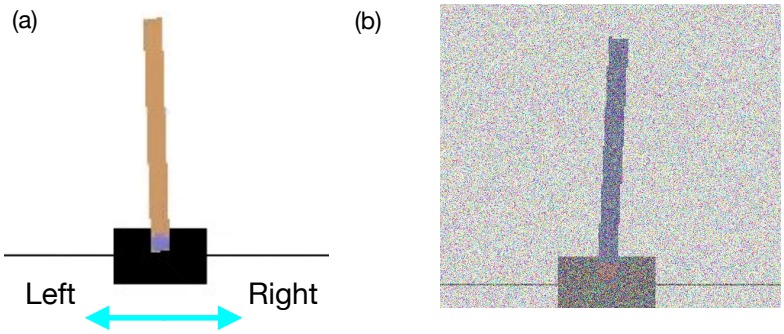


Figure A4: Visual examples of Cartpole game and change factors. (a) Cartpole game; (b) Modified Cartpole game with Gaussian noise on the image. The light blue arrows are added to show the direction in which the agent can move.

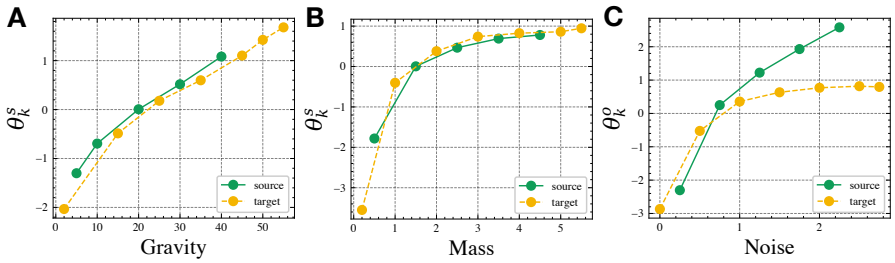


Figure A5: The estimated $\boldsymbol{\theta}_k^{min}$ for the three change factors in Cartpole (POMDP): gravity (A), mass (B), and noise level (C) for $N_{target} = 50$. .

Gaussian noise on the images with variance $\sigma = \{0.25, 0.75, 1.25, 1.75, 2.25\}$ in source domains, and $\sigma = \{0.5, 2.75\}$ in target domains.

We summarize the detailed settings in both source and target domains in Table A1. In particular in each experiment we use all source domains for each change factor and one of the target domains at a time in either the interpolation and extrapolation set.

	Gravity	Mass	Noise
Source domains	$\{5, 10, 20, 30, 40\}$	$\{0.5, 1.5, 2.5, 3.5, 4.5\}$	$\{0.25, 0.75, 1.25, 1.75, 2.25\}$
Interpolation set	$\{15\}$	$\{1.0\}$	$\{0.5\}$
Extrapolation set	$\{55\}$	$\{5.5\}$	$\{2.75\}$

Table A1: The settings of source and target domains for modified Cartpole experiments.

A5.1.1 Learned θ_k in Cartpole experiments

Fig. A5 shows the estimated $\boldsymbol{\theta_k}$ in the modified Cartpole experiments. For the gravity and mass scenarios, the learned parameters with different sample sizes are close with each other. This phenomenon indicates that even with only a few samples ($N_{target} = 50$), AdaRL can estimate these change parameters very well. For the noise level factor, the learned curves with different sample sizes have a similar behaviour, but the distance is larger. We can see that the θ_k we learn is approximately a linear function of the actual perturbation in gravity, while for the mass and noise are monotonic functions.

A5.1.2 Average final scores for multiple N_{target} in Cartpole experiments

Tables A2 and A3 show the complete results of the modified Cartpole experiments (POMDP settings) for $N_{target} = 20$ and $N_{target} = 10000$. Table A4 and A5 give the complete results of the modified

	Oracle Upper bound	Non-t lower bound	PNN (Rusu et al., 2016)	PSM (Agarwal et al., 2021a)	MTQ (Fakoor et al., 2020)	AdaRL* Ours w/o masks	AdaRL Ours
G_in	1930.5 (±1042.6)	828.5 • (±509.4)	1113.4 • (±719.2)	1008.5 • (±453.6)	1257.2 (±503.5)	1290.6 (±589.1)	1302.7 (±874.0)
G_out	408.6 (±67.2)	54.0 • (±13.6)	109.7 • (±24.2)	156.8 • (±49.5)	120.5 • (±87.4)	173.8 (±39.3)	198.4 (±54.5)
M_in	2004.8 (±404.3)	447.2 • (±39.6)	1120.6 • (±348.1)	982.5 • (±363.2)	1245.7 • (±274.0)	1095.8 • (±521.3)	**1361.0** (±327.3)
M_out	1498.6 (±625.4)	130.6 • (±39.8)	528.5 • (±251.4)	830.6 • (±317.2)	875.2 • (±262.5)	764.2 • (±320.9)	**1082.5** (±236.3)
N_in	8640.5 (±3086.1)	679.4 • (±283.5)	4170.6 • (±2202.2)	4936.5 • (±1604.9)	3985.7 • (±2387.4)	4954.3 • (±2627.8)	**5761.2** (±2341.5)
N_out	4465.2 (±667.3)	584.0 • (±429.2)	2841.5 • (±385.2)	2650.2 • (±453.6)	2654.0 • (±277.9)	1785.2 • (±470.3)	**3318.7** (±293.5)

Table A2: Average final scores in modified Cartpole (POMDP) with $N_{target} = 20$. 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., 2021a)	MTQ (Fakoor et al., 2020)	AdaRL* Ours w/o masks	AdaRL Ours
G_in	1930.5 (±1042.6)	1115.2 • (±341.8)	1637.4 • (±378.2)	1838.4 (±358.1)	1459.2 • (±688.5)	1864.5 (±694.1)	1924.6 (±874.0)
G_out	408.6 (±67.2)	161.3 • (±65.9)	329.6 • (±48.9)	457.3 (±138.5)	393.2 • (±76.5)	384.2 • (±103.7)	410.6 (±92.3)
M_in	2004.8 (±404.3)	596.0 • (±373.4)	1672.3 • (±642.9)	1798.5 • (±493.0)	1905.4 (±378.2)	1864.2 (±309.5)	1898.5 (±683.4)
M_out	1498.6 (±625.4)	325.6 • (±146.3)	1206.8 • (±394.7)	1339.4 • (±520.5)	1296.2 • (±773.1)	1297.4 • (±411.2)	**1486.3** (±598.2)
N_in	8640.5 (±3086.1)	1239.6 • (±380.5)	6476.2 • (±3132.9)	7493.4 • (±1981.5)	7932.9 (±2389.0)	7382.4 • (±2915.3)	8179.8 (±2356.0)
N_out	4465.2 (±667.3)	962.5 • (±341.8)	3043.9 • (±1098.6)	2987.2 • (±1172.3)	3892.4 • (±763.0)	4183.6 (±782.2)	4235.2 (±532.4)

Table A3: Average final scores in modified Cartpole (POMDP) with $N_{target} = 10000$. 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	CAVIA (Zintgraf et al., 2019)	PEARL (Rakelly et al., 2019)	AdaRL* Ours w/o masks	AdaRL Ours
G_in	2486.1 (±369.7)	972.6 • (±368.5)	1651.5 • (±623.8)	1720.3 • (±589.4)	1602.7 • (±393.6)	**1943.2** (±765.4)
G_out	693.9 (±100.6)	243.8 • (±45.2)	356.2 (±76.5)	362.1 (±57.3)	292.4 • (±91.8)	395.6 (±101.7)
M_in	2678.2 (±630.5)	480.3 • (±136.2)	1306.8 • (±376.5)	1589.4 • (±682.3)	1624.8 • (±531.6)	**1962.0** (±652.8)
M_out	1405.6 (±368.0)	306.5 • (±162.4)	853.2 • (±317.6)	969.4 • (±238.5)	984.6 • (±209.8)	**1113.5** (±394.2)
G_in & M_in	1984.2 (±871.3)	374.9 • (±126.8)	1174.3 • (±298.2)	964.3 • (±370.5)	1209.6 • (±425.7)	**1392.7** (±392.6)
G_out & M_out	939.4 (±270.5)	292.4 • (±127.6)	494.6 • (±201.3)	368.4 • (±259.8)	399.8 • (±242.5)	**531.2** (±272.5)

Table A4: Average final scores in modified Cartpole (MDP) with $N_{target} = 20$. The best non-oracle results are marked in red, while **bold** indicates a statistically significant result w.r.t. all the baselines. G, M, and N denote the gravity, mass, and noise respectively. "•" 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	CAVIA (Zintgraf et al., 2019)	PEARL (Rakelly et al., 2019)	AdaRL* Ours w/o masks	AdaRL Ours
G_in	2486.1 (±369.7)	986.3 • (±392.5)	1907.4 • (±526.8)	2102.3 • (±398.5)	1864.0 • (±369.2)	**2365.1** (±403.5)
G_out	693.9 (±100.6)	349.2 • (±72.0)	502.9 • (±133.2)	585.7 • (±98.6)	494.7 • (±151.4)	**604.8** (±117.6)
M_in	2678.2 (±630.5)	643.9 • (±281.3)	2008.6 • (±436.2)	2106.2 • (±436.7)	2148.9 • (±387.2)	**2415.2** (±591.4)
M_out	1405.6 (±368.0)	617.4 • (±145.3)	1182.7 • (±255.8)	1294.5 (±210.6)	1207.5 (±251.3)	1263.5 (±362.9)
G_in & M_in	1984.2 (±871.3)	452.6 • (±178.3)	1275.0 • (±432.5)	1468.7 • (±697.2)	1395.4 • (±387.2)	**1589.4** (±379.5)
G_out & M_out	939.4 (±270.5)	596.2 • (±137.5)	709.5 • (±386.0)	743.8 • (±200.9)	724.7 (±283.8)	769.3 (±208.4)

Table A5: Average final scores in modified Cartpole (MDP) with $N_{target} = 10000$. The best non-oracle results are marked in red, while **bold** indicates a statistically significant result w.r.t. all the baselines. G, M, and N denote the gravity, mass, and noise respectively.

Cartpole experiments (MDP settings with symbolic input) for $N_{target} = 20$ and $N_{target} = 10000$. We average the scores across 30 trials from different random seeds during the policy learning stage. The results suggest that, in most cases, AdaRL can outperform other baselines.

A5.1.3 Average policy learning curves in terms of steps

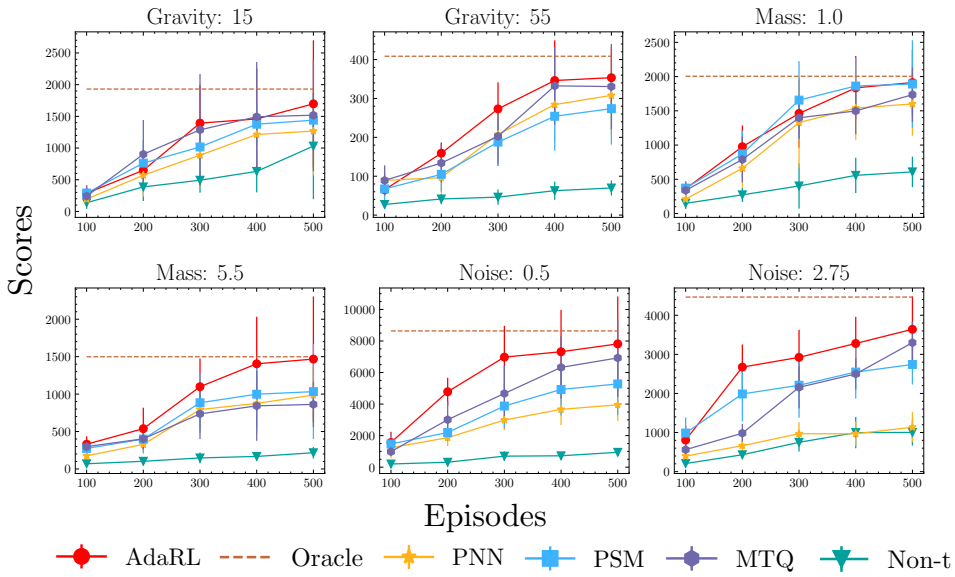


Figure A6: Learning curves for modified Cartpole experiments (POMDP version) with change factors. The reported scores are averaged across 30 trials.

Fig. A7 and A6 provide the learning curves for modified Cartpole experiments (MDP and POMDP versions) with multiple change factors. In most cases, AdaRL can converge faster than other baselines.

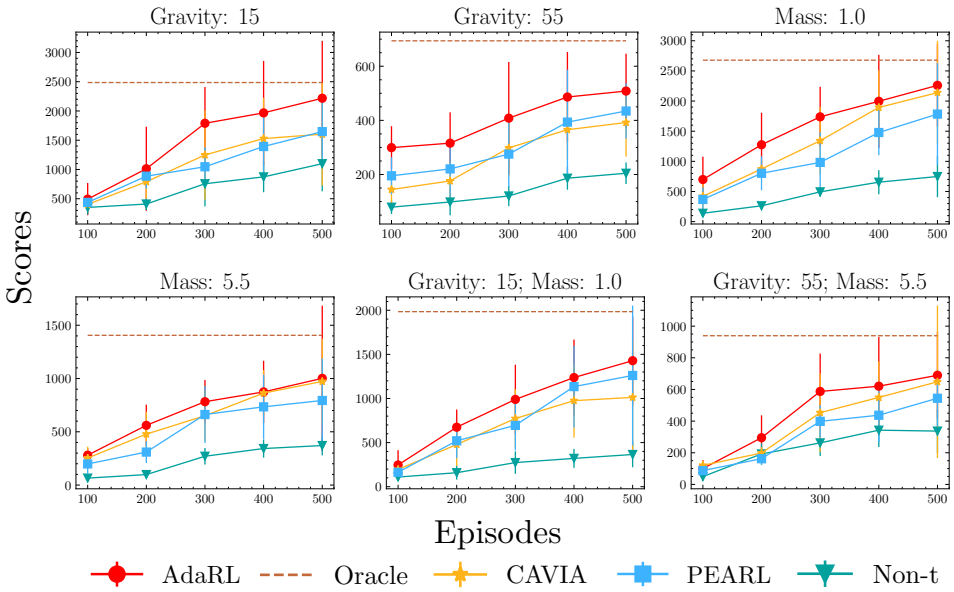


Figure A7: Learning curves for modified Cartpole experiments (MDP version) with change factors. The reported scores are averaged across 30 trials.

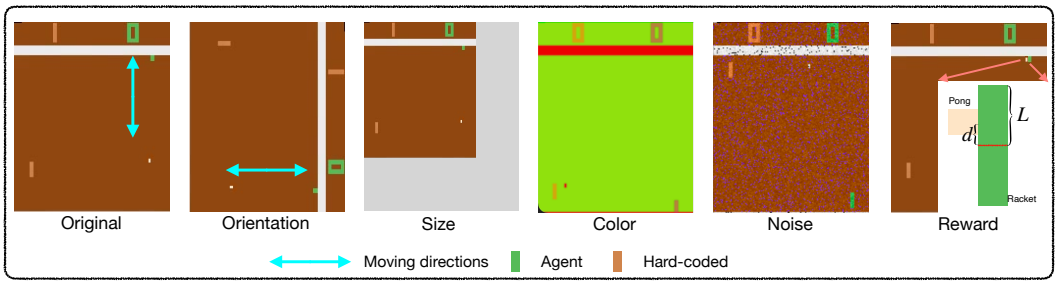


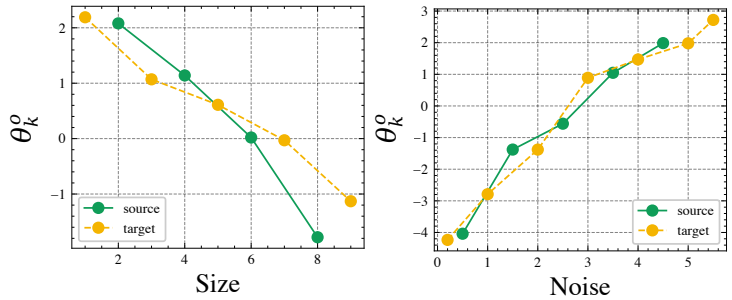
Figure A8: Visual example of the original Pong game and the various change factors. The light blue arrows are added to show the direction in which the agent can move.

A5.2 Complete results of the modified Pong experiment with changing dynamics and observations

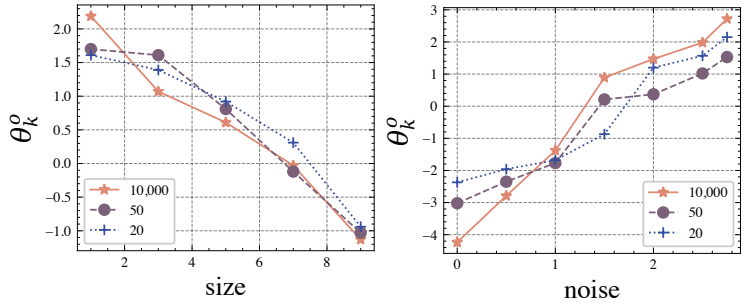
Atari Pong is a two-dimensional game that simulates table tennis. The agent controls a paddle moving up and down vertically, aiming at hitting the ball. The goal for the agent is to reach higher scores, which are earned when the other agent (hard-coded) fails to hit back the ball. We show the example of the original visual inputs and how it appears after we have changed each of the change factors in Fig. A8.

In source domains, the degrees are chosen from $\omega = \{0°, 180°\}$, and in target domains, they are chosen from $\omega = \{90°, 270°\}$. For the image size, we reduce the original image by a factor of $\{2, 4, 6, 8\}$ in source domains and by a factor of $\{3, 9\}$ in target domains.

We summarize the detailed settings in both source and target domains in Table A6. In particular, in each experiment we use all source domains for each change factor and one of the target domains at a time in either the interpolation and extrapolation set.



(a) Source vs. target domains for $N_{target} = 10000$.



(b) Different N_{target} in target domains.

Figure A9: Learned θ_k^o for the two change factors, size and noise, in modified Pong.

	Size	Orientations	Noise	Background colors
Source domains	$\{2, 4, 6, 8\}$	$0°, 180°$	$\{0.25, 0.75, 1.25, 1.75, 2.25\}$	original, green, red
Interpolation set	$\{3\}$	$90°$	$\{1.0\}$	yellow
Extrapolation set	$\{9\}$	$270°$	$\{2.75\}$	white

Table A6: The settings of source and target domains for modified Pong experiments.

A5.2.1 Learned θ_k in modified Pong experiments

Fig. A9a, Table A7, and Table A8 show the learned θ_k in modified Pong experiments across different change factors.

Table A7: The learned θ_k^s across different orientation angles with different N_{target} in modified Pong. The bold columns represent the target domains.

N_{target}	Orientations	$0°$	**$90°$**	$180°$	**$270°$**
10000	θ_k^{s1}	-2.32	-1.78	1.69	0.44
	θ_k^{s2}	-2.94	-1.86	1.47	1.59
50	θ_k^{s1}	-2.01	-0.87	1.85	0.69
	θ_k^{s2}	-2.59	-1.84	1.42	1.07
20	θ_k^{s1}	-1.69	-1.23	0.79	1.38
	θ_k^{s2}	-1.98	-0.56	0.82	1.20

We can find that each dimension of the learned $\boldsymbol{\theta}_k^s$ is a nonlinear monotonic function of the change factors. Table A7, Table A8 and Fig. A9b also give the learned θ^k with different sample sizes N_{target} in target domains. Similarly, the learned curves with different sample sizes are homologous. Even with a few samples, AdaRL can still capture the model changes well.

Table A8: The learned θ_k^o across different colors with different N_{target} in modified Pong. The bold columns represent the target domains.

N_{target}	Colors	Original	Red	Green	**Yellow**	**White**
10000	θ_k^{o1}	1.36	1.47	1.04	1.58	−0.91
	θ_k^{o2}	0.72	−1.15	1.17	0.96	−1.33
	θ_k^{o3}	0.93	−1.28	−1.31	−0.65	−1.09
50	θ_k^{o1}	0.96	1.13	0.82	1.26	−0.73
	θ_k^{o2}	0.59	−0.46	0.75	1.32	−0.59
	θ_k^{o3}	0.61	−1.02	−0.91	−0.18	−0.49
20	θ_k^{o1}	1.09	1.38	0.65	1.30	−0.46
	θ_k^{o2}	0.58	−0.72	0.39	1.60	−0.27
	θ_k^{o3}	0.38	−0.59	−0.63	−0.24	−0.33

A5.2.2 AVERAGE FINAL SCORES FOR MULTIPLE N_{target}

Table A2 and A3 provides the complete results of the modified Pong experiments with $N_{target} = 20$ and 10000, respectively. The details of both source and target domains are listed in Table A6. Similar to the results of Cartpole, AdaRL can perform the best among all baselines in Pong experiments. As shown in the results of the main paper, AdaRL consistently outperforms the other methods.

A5.2.3 AVERAGE POLICY LEARNING CURVES IN TERMS OF STEPS

Fig. A12 gives the learning curves for modified Pong experiments with multiple change factors. From the results, we can find that AdaRL can converge faster than other baselines.

A5.3 COMPLETE RESULTS OF THE MODIFIED PONG EXPERIMENT WITH CHANGING REWARD FUNCTIONS

Table A9 summarizes the detailed change factors in both linear and non-linear reward groups.

	Linear reward (k_1)	Non-linear reward (k_2)
Source domains	$\{0.1, 0.2, 0.3, 0.4, 0.6, 0.7, 0.8\}$	$\{2.0, 3.0, 5.0, 6.0, 7.0, 8.0, 9.0\}$
Interpolation set	$\{0.5\}$	$\{4.0\}$
Extrapolation set	$\{0.9\}$	$\{1.0\}$

Table A9: The settings of source and target domains for modified Pong experiments.

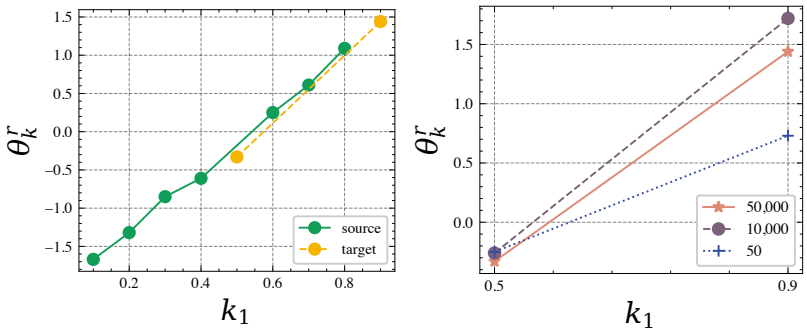
We denote with L the half-length of the paddle and then formulate the two groups of reward functions as: (1) Linear reward functions: $r_t = \frac{k_1 d}{L}$, where $k_1 \in \{0.1, 0.2, 0.3, 0.4, 0.6, 0.7, 0.8\}$ in source domains and $k_1 \in \{0.5, 0.9\}$ in target domains; and (2) Non-linear reward functions: $r_t = \frac{k_2 L}{d+3L}$, where $k_2 \in \{2.0, 3.0, 5.0, 6.0, 7.0, 8.0, 9.0\}$ in source domains and $k_2 \in \{1.0, 4.0\}$ in target domains.

A5.3.1 LEARNED θ_r IN MODIFIED PONG EXPERIMENTS

Fig. A10 and A11 give the learned θ_r with both linear and non-linear rewards. In both groups, the learned θ_r is linearly or monotonically correlated with the change factor k_1 and there is no significant gap between the learned θ_r with different N_{target}.

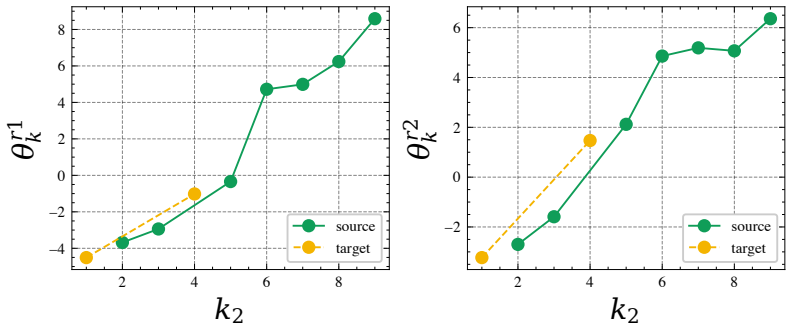
A5.3.2 AVERAGE FINAL SCORES FOR MULTIPLE N_{target}

Table A12, A13 and A14 shows the average final scores for $N_{target} = 50$, 10000 and 50000 in modified Pong experiments with changing rewards. AdaRL consistently outperforms the other methods across different N_{target}.

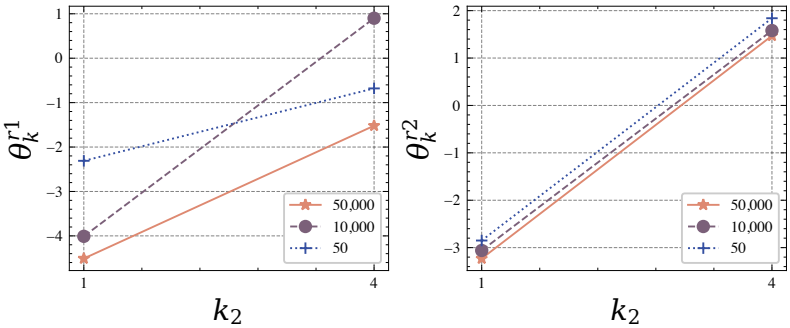


(a) Learnt θ_r in each domain. (b) Different N_{target} in target domains.

Figure A10: Learned θ_k^r for the linear changing rewards in modified Pong.



(a) Learnt θ_r in each domain.



(b) Different N_{target} in target domains.

Figure A11: Learned θ_k^r for the non-linear changing rewards in modified Pong.

A5.3.3 Average policy learning curves in terms of steps

Fig. A12 (last two rows) gives the learning curves for modified Pong experiments with changing rewards.

A5.4 Results on MuJoCo benchmarks

We also apply AdaRL on MuJoCo benchmarks (Todorov et al., 2012), including Cheetah and Ant with a set of target velocities. We follow the setup in MAML (Finn et al., 2017) and CAVIA (Zintgraf et al., 2019). In model estimation stage, we choose 20 tasks for each of the game. The goal velocity of each task is sampled between 0.0 and 2.0 for the cheetah and between 0.0 and 3.0 for ant. The reward is the negative absolute value between agent's current and the goal velocity. In policy optimization stage, we utilize Trust Region Policy Optimization (TRPO Schulman et al. (2015)). The results on

	Oracle Upper bound	Non-t lower bound	PNN (Rusu et al., 2016)	PSM (Agarwal et al., 2021a)	MTQ (Fakoor et al., 2020)	AdaRL* Ours w/o masks	AdaRL Ours
O_in	18.65 (±2.43)	4.30 • (±2.95)	8.68 • (±5.78)	9.65 • (±3.19)	14.80 • (±2.02)	15.08 • (±3.19)	**16.79** (±1.84)
O_out	19.86 (±1.09)	5.09 • (±2.41)	10.61 • (±5.26)	9.94 • (±6.23)	11.82 (±2.46)	11.92 (±3.09)	12.70 (±4.38)
C_in	19.35 (±0.45)	7.72 • (±2.63)	13.75 • (±4.16)	10.87 • (±5.15)	14.80 • (±3.07)	16.07 (±2.86)	16.29 (±3.35)
C_out	19.78 (±0.25)	7.09 • (±3.21)	13.37 • (±4.42)	12.59 • ±3.80)	15.34 (±3.22)	15.84 (±3.10)	16.55 (±2.09)
S_in	18.32 (±1.18)	6.25 • (±3.42)	12.93 • (±2.72)	10.67 • (±1.85)	12.78 • (±3.46)	13.86 (±2.95)	14.92 (±4.48)
S_out	19.01 (±1.04)	5.45 • (±2.75)	9.69 • (±6.27)	13.80 • (±3.15)	12.62 • (±2.41)	15.31 (±2.13)	15.88 (±3.72)
N_in	18.48 (±1.25)	4.29 • (±2.22)	13.85 • (2.83)	13.69 • (±2.21)	10.96 • (±3.27)	13.51 • (±3.07)	**15.57** (±2.95)
N_out	18.26 (±1.11)	5.19 • (±2.47)	11.83 • (±3.82)	14.07 • (±2.56)	12.75 • (±3.18)	14.29 • (±3.10)	**16.38** (±2.72)

Table A10: Average final scores on modified Pong (POMDP) with $N_{targets} = 20$. The best non-oracle results are marked in red, while **bold** indicates a statistically significant result w.r.t. all the baselines. O, C, S, and N denote the orientation, color, size, and noise factors, respectively.

	Oracle Upper bound	Non-t lower bound	PNN (Rusu et al., 2016)	PSM (Agarwal et al., 2021a)	MTQ (Fakoor et al., 2020)	AdaRL* Ours w/o masks	AdaRL Ours
O_in	18.65 (±2.43)	8.04 • (±1.78)	12.19 • (±3.07)	12.37 • (±2.92)	14.64 • (±3.01)	17.42 • (±2.20)	**18.85** (±1.63)
O_out	19.86 (±1.09)	6.97 • (±1.88)	16.48 (±3.10)	15.79 • (±2.29)	12.75 • (±4.93)	17.25 (±1.85)	17.93 (±2.41)
C_in	19.35 (±0.45)	8.09 • (±3.11)	15.89 • (±3.49)	16.70 • (±2.38)	17.85 (±2.16)	17.73 • (±2.01)	18.93 (±1.37)
C_out	19.78 (±0.25)	7.48 • (±2.09)	16.85 • (±3.17)	16.29 • (±2.64)	17.93 • (±2.35)	18.49 (±2.04)	19.28 (±1.36)
S_in	18.32 (±1.18)	7.45 • (±3.15)	12.89 • (±2.04)	13.84 • (±3.27)	15.33 • (±2.03)	15.79 • (±2.62)	**17.49** (±2.18)
S_out	19.01 (±1.04)	7.04 • (±2.36)	14.69 • (±2.03)	17.25 • (±2.30)	18.48 (±1.36)	17.82 • (±1.98)	19.21 (±0.63)
N_in	18.48 (±1.25)	6.82 • (±2.09)	13.84 • (±2.82)	16.80 • (±1.73)	17.58 (±2.19)	15.93 • (±3.68)	18.25 (±1.81)
N_out	18.26 (±1.11)	7.82 • (±2.46)	14.89 • (±2.98)	16.85 (±3.94)	17.03 (±2.36)	16.49 • (±3.25)	17.85 (±2.16)

Table A11: Average final scores on modified Pong (POMDP) with $N_{targets} = 10000$. The best non-oracle results are marked in red, while **bold** indicates a statistically significant result w.r.t. all the baselines. O, C, S, and N denote the orientation, color, size, and noise factors, respectively.

	Oracle Upper bound	Non-t lower bound	PNN (Rusu et al., 2016)	PSM (Agarwal et al., 2021a)	MTQ (Fakoor et al., 2020)	AdaRL* Ours w/o masks	AdaRL Ours
Rl_in	7.98 (±3.81)	3.19 • (±2.27)	4.95 (±1.08)	5.04 (±2.11)	4.78 (±2.10)	3.49 • (±1.97)	5.81 (±2.06)
Rl_out	9.61 (±4.78)	5.19 • (±2.80)	5.89 (±1.93)	6.03 (±2.71)	6.21 (±3.14)	5.64 (±2.59)	6.12 (±3.45)
Rn_in	7.62 (±2.16)	2.85 • (±1.71)	5.31 (±2.78)	5.06 (±3.89)	5.52 (±3.47)	5.79 (±3.03)	5.84 (±3.17)
Rn_out	41.36 (±5.70)	21.73 • (±8.54)	27.19 (±5.82)	23.27 • (±8.01)	25.49 • (±6.18)	26.33 • (±7.94)	29.92 (±6.39)

Table A12: Results on modified Pong game with $N_{targets} = 50$. The best non-oracle results are marked in red, while **bold** indicates a statistically significant result w.r.t. all the baselines. Rl and Ro denote the linear reward and nonlinear reward-changing cases, respectively.

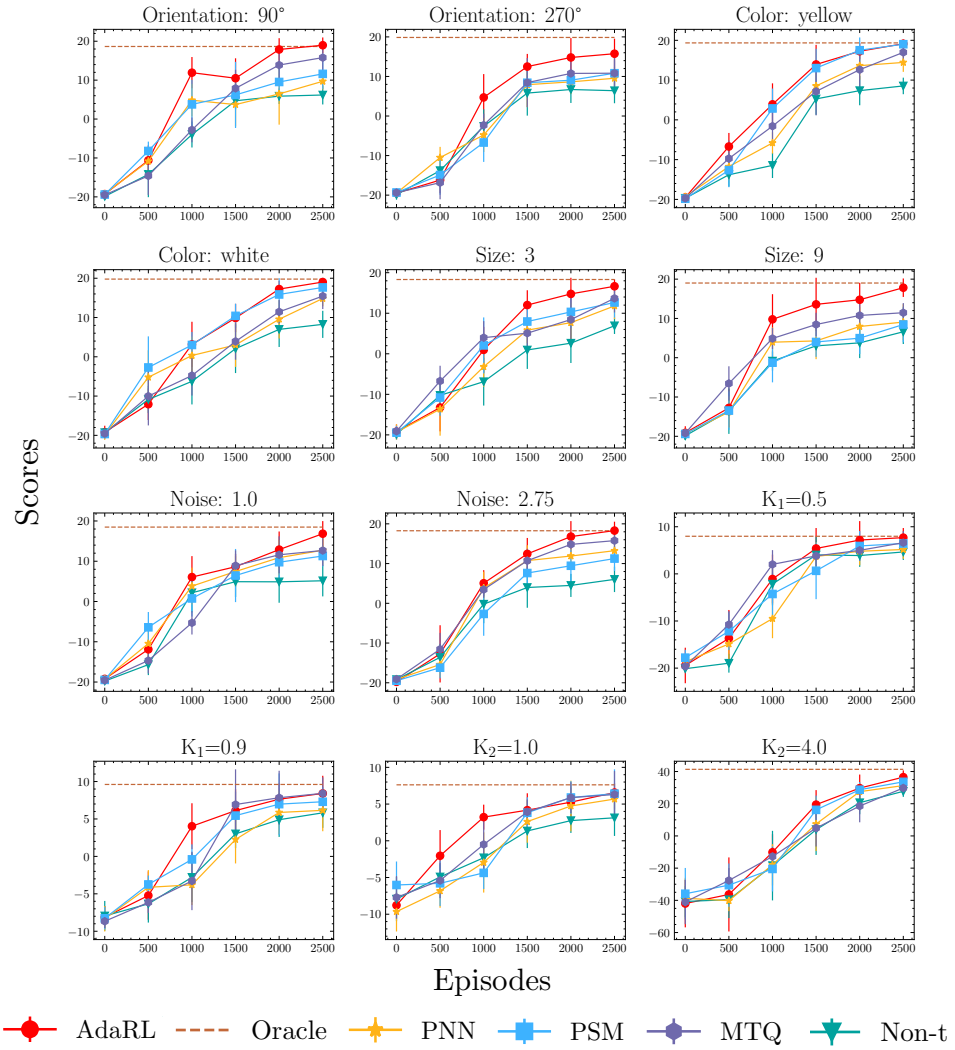


Figure A12: Learning curves for modified Pong experiments with change factors. The reported scores are averaged across 30 trials.

	Oracle Upper bound	Non-t lower bound	PNN (Rusu et al., 2016)	PSM (Agarwal et al., 2021a)	MTQ (Fakoor et al., 2020)	AdaRL* Ours w/o masks	AdaRL Ours
Rl_in	7.98 (±3.81)	4.65 • (±1.70)	5.17 • (±1.98)	6.45 • (±1.82)	6.62 • (±2.45)	6.88 • (±3.19)	**7.69** (±2.04)
Rl_out	9.61 (±4.78)	5.82 • (±2.01)	6.15 • (±2.79)	7.30 • (±1.98)	8.42 (±2.14)	7.04 • (±2.52)	8.41 (±2.36)
Rn_in	7.62 (±2.16)	3.13 • (±2.47)	5.68 • (±1.42)	6.42 (±3.31)	6.30 (±3.19)	5.52 • (±1.09)	6.57 (±1.24)
Rn_out	41.36 (±5.70)	27.70 • (±3.45)	31.28 • (±4.09)	33.60 • (±5.52)	29.77 • (±3.85)	33.83 • (±5.02)	**36.52** (±4.18)

Table A13: Average final scores on modified Pong (POMDP) with $N_{target} = 10000$. The best non-oracle results are marked in red, while **bold** indicates a statistically significant result w.r.t. all the baselines. Rl and Ro denote the linear and nonlinear reward changes, respectively.

the target domains are specified in Table A15, suggesting that AdaRL can achieve better performance than the meta-learning approaches (i.e., MAML and CAVIA).

	Oracle Upper bound	Non-t lower bound	PNN (Rusu et al., 2016)	PSM (Agarwal et al., 2021a)	MTQ (Fakoor et al., 2020)	AdaRL* Ours w/o masks	AdaRL Ours
Rl_in	7.98 ($\pm$3.81)	4.81 $\bullet$ ($\pm$2.03)	5.94 $\bullet$ ($\pm$4.87)	6.90 ($\pm$3.47)	7.34 ($\pm$3.18)	6.46 $\bullet$ ($\pm$3.12)	7.93 ($\pm$2.09)
Rl_out	9.61 ($\pm$4.78)	3.89 $\bullet$ ($\pm$2.16)	7.85 ($\pm$2.88)	7.37 ($\pm$3.75)	8.77 ($\pm$2.61)	7.78 ($\pm$3.10)	8.94 ($\pm$2.02)
Rn_in	7.62 ($\pm$2.16)	3.58 $\bullet$ ($\pm$1.09)	6.91 ($\pm$2.85)	7.01 ($\pm$2.46)	6.28 $\bullet$ ($\pm$3.14)	6.30 $\bullet$ ($\pm$2.63)	7.57 ($\pm$1.94)
Rn_out	41.36 ($\pm$5.70)	29.98 $\bullet$ ($\pm$3.02)	36.08 $\bullet$ ($\pm$10.35)	37.26 ($\pm$11.25)	38.48 ($\pm$12.59)	34.19 $\bullet$ ($\pm$9.36)	41.25 ($\pm$6.92)

Table A14: Results on modified Pong game with $N_{targets} = 50000$. The best non-oracle results are marked in red, while **bold** indicates a statistically significant result w.r.t. all the baselines. Rl and Ro denote the linear reward and nonlinear reward-changing cases, respectively.

	MAML (Finn et al., 2017)	CAVIA (Zintgraf et al., 2019)	AdaRL Ours
Cheetah (vel)	-89.8 ($\pm$4.1)	-86.5 ($\pm$2.0)	**-81.7** ($\pm$3.2)
Ant (vel)	100.4 ($\pm$10.9)	95.7 ($\pm$6.92)	**106.8** ($\pm$8.4)

Table A15: Results on MuJoCo benchmarks (Cheetah and Ant experiments with different target velocities, with 30 trials each) with $N_{targets} = 50,000$. The best results are marked in **red**.

A5.5 Effect of the policy used for data collection during model estimation

In our framework, we use the random policy to generate trajectories for each domain. The generated trajectories are further used to conduct the model estimation. To study whether the random policy will affect the effectiveness of model estimation, we compare the learnt parameters of latent space in model estimation with the trajectories generated via (1) the random policy in our framework, and (2) the optimal policies learnt on the source domains. Here we show the case with changing gravity in the modified Cartpole game. In Fig. A13, we give the learnt parameters (μ and $\log \sigma$) of the first and second components in the Gaussian mixtures of the 10-th latent state at 20-th epoch. The results demonstrate that the difference between the two sets of learnt parameters is limited.

A5.6 More statistical evaluation protocols on the performance

In this section, we provide a more detailed and comprehensive comparison on the performances of AdaRL and other baseline methods. Following the recent published work on reliable evaluation for RL (Agarwal et al., 2021b), we utilize 4 evaluation metrics: median, interquartile mean (IQM), mean, and optimality gap. Median and mean are the sample median and mean, respectively. IQM discards the top and bottom 25% samples and then computes the mean value of the remaining ones. Thus, this factor is insensitive to outliers. Optimality gap is quantified via the gap between the performance of each method and the mean score obtained by the oracle agent. Therefore, higher mean, median and IQM and lower optimality are the indications of better methods. All metrics are with 95% bootstrap confidence intervals (CIs) (Efron, 1992).

Fig. A14, A15, A16, and A17 give the comparison on AdaRL and baselines using the set of evaluation protocols in different games and settings. The results suggest that in most cases, AdaRL performs consistently better than the baselines across all evaluation metrics.

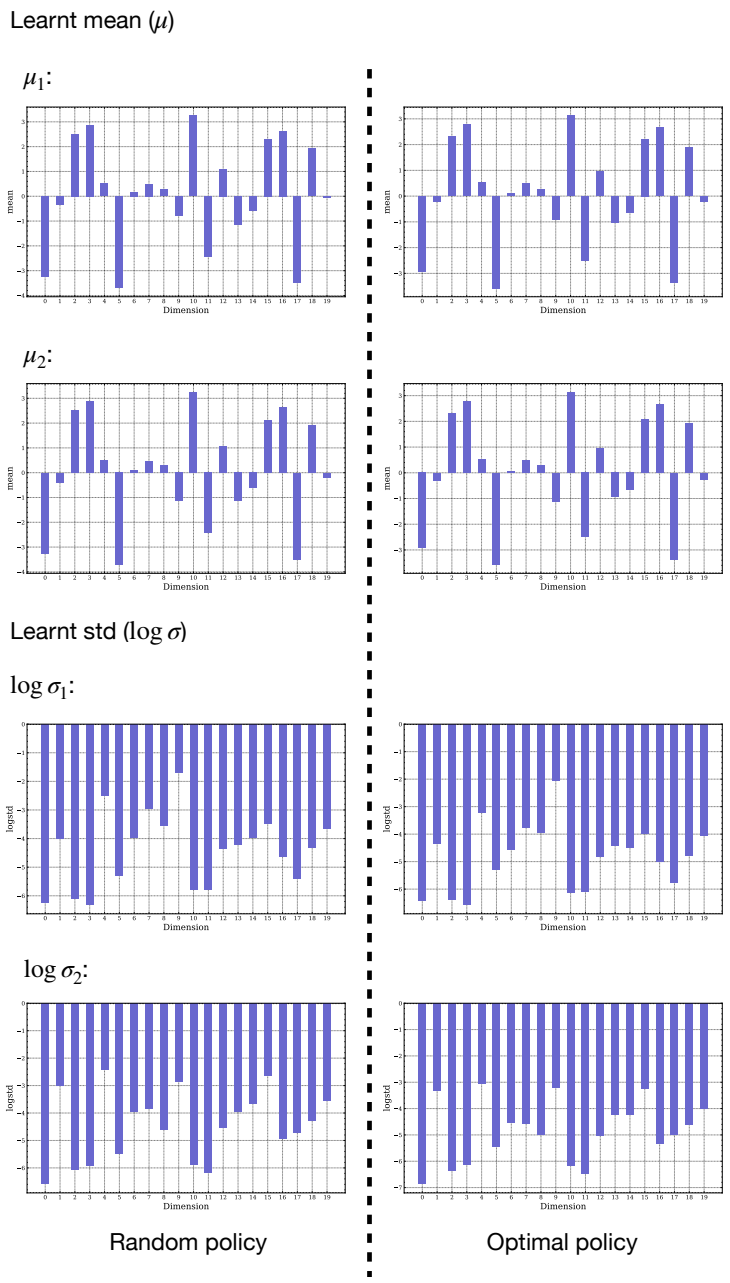


Figure A13: Learnt μ and $\log \sigma$ in model estimation with both random policy (left), and optimal policy (right).

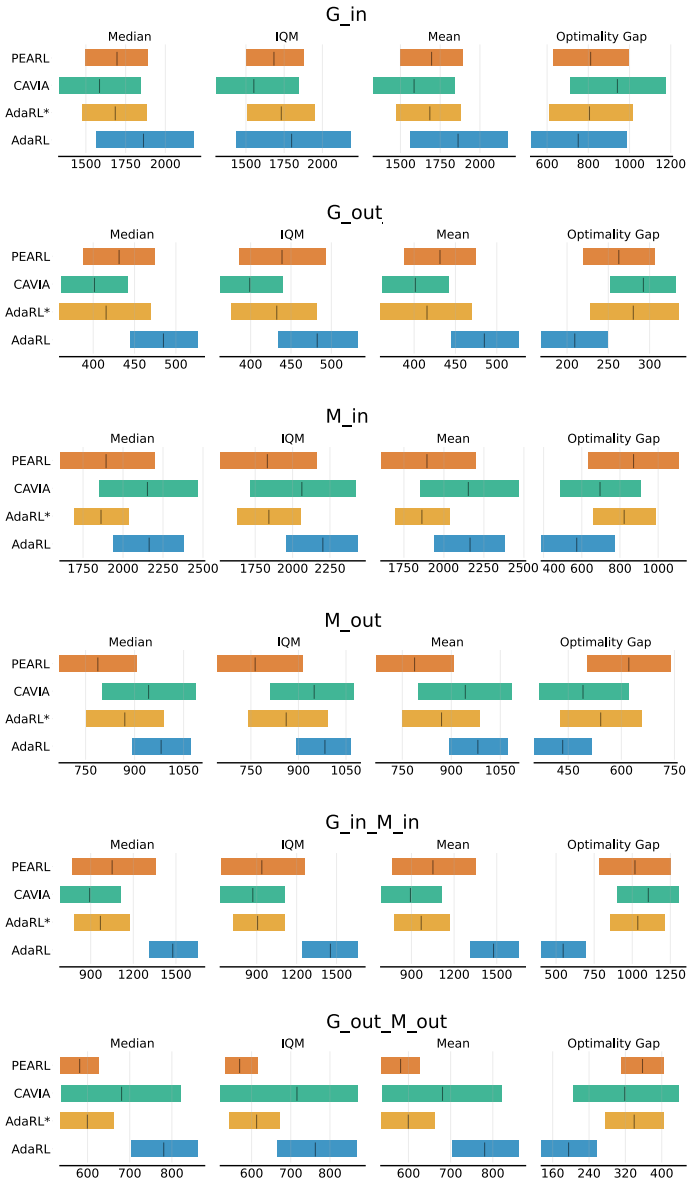


Figure A14: Evaluation on the results of modified CartPole game under MDP settings for $N_{target} = 50$.

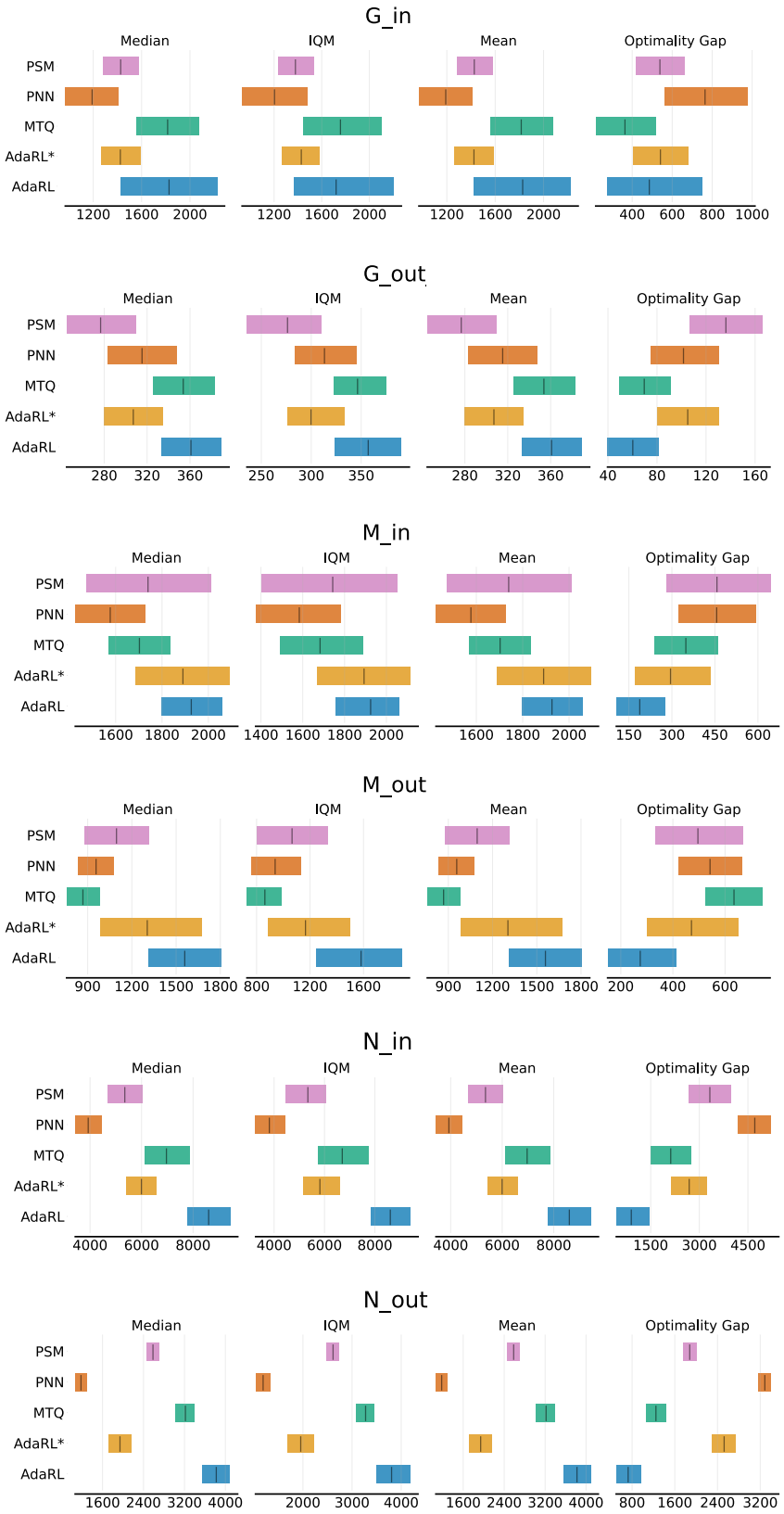


Figure A15: Evaluation on the results of modified CartPole game under POMDP settings for $N_{target} = 50$.

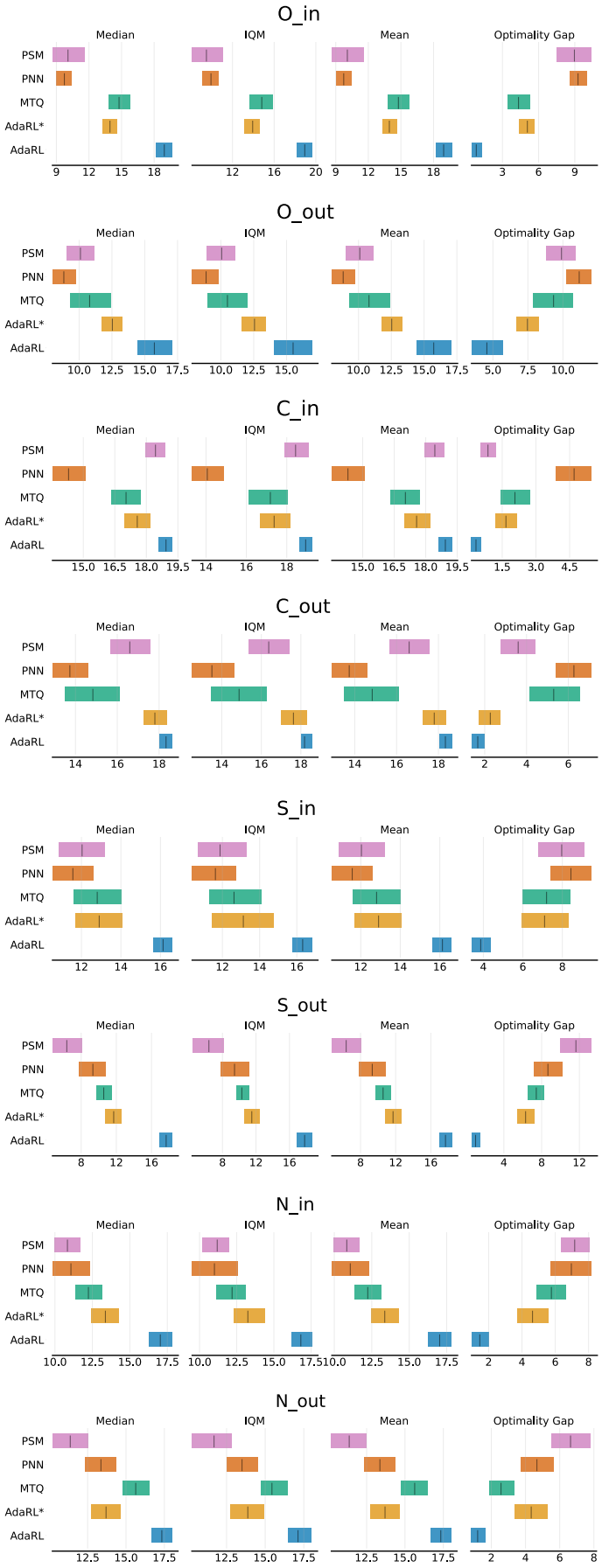


Figure A16: Evaluation on the results of modified Atari Pong game under POMDP settings with changing orientation, color, size, and noise levels for $N_{target} = 50$.

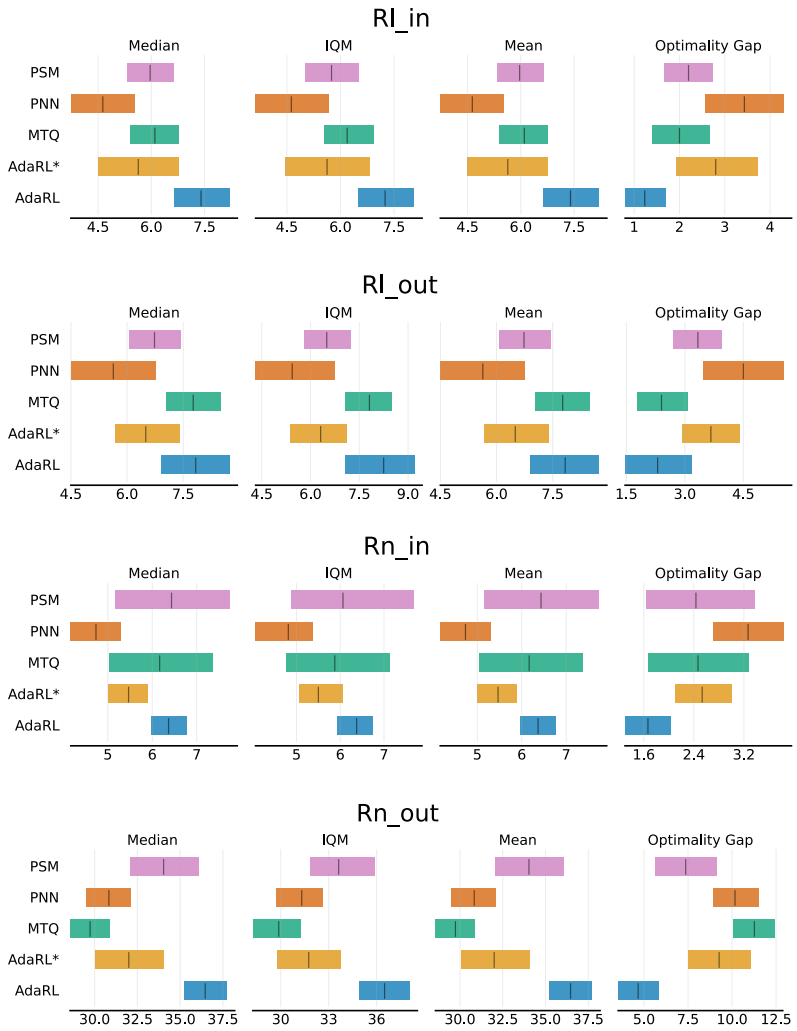


Figure A17: Evaluation on the results of modified Atari Pong game under POMDP settings with changing reward functions for $N_{target} = 10000$.

A5.7 PSEUDO CODE OF MISS-VAE

Algorithm A1 Pseudo code of Miss-VAE.

```
# s: latent state; theta_o, theta_s, theta_r: changing factors;
# C_so, C_sr, C_ar, C_ss, C_as, C_theta_oo, C_theta_rr, C_theta_ss: structure masks;
# o: observation; a: action; r: reward;
# num_mix: the number of components in mixture Gaussian model

######################################## Encoder ########################################
for X in loader: # load a batch of data from K domains
   ho = conv(o)
   ha = conv(a)
   hr = conv(r)
   hd = fc(theta_o, theta_s, theta_r)
   h = concat[ho, ha, hr, hd]
   output, last_state = LSTM(h) # output the logmix, mean, and logstd of state
   s = sample(output) # sample the state

######################################## Decoder ########################################
   # Reconstruct o_{t}
   s_o = multiply(s, C_so)
   theta_o_o = multiply(theta_o, C_theta_oo)
   h_s_theta = concat(fc(s_o), fc(theta_o_o))
   o_t = conv_transpose(h_s_theta) # observation of current step

   # Reconstruct r_{t+1}
   s_r = multiply(s, C_sr)
   a_r = multiply(a, C_ar)
   theta_r_r = multiply(theta_r, C_theta_rr)
   h_as = fc(concat[s_r, a_r])
   h_theta_r = fc(theta_r_r)
   r_t+1 = fc(concat[h_as, h_theta_r]) # reward of current step

   # Predict O_{t+1}
   o_t+1 = conv_transpose(s, theta_o, theta_s) # reward of next step

   # Predict r_{t+2}
   r_t+2 = fc(concat[fc(s), fc(a), fc(theta_o), fc(theta_s)]) # reward in the next step

   # Markovian transition
   s_s = multiply(s, C_ss)
   theta_s_s = multiply(theta_s, C_theta_ss)
   a_s = multiply(a, C_as)
   h_dyn = fc(concat[s_s, theta_s_s, a_s])
   s_output = linear(h_dyn)
   s_logmix, s_mean, log_logstd = s_output.split # state in next step

######################################## Loss ########################################
   # Reconstruction loss
   Rec_loss = mean(MSE(o - o_t) + MSE(r - r_t+1))

   # Prediction loss
   Pred_loss = mean(MSE(o[:,:-1,:] - o_t+1) + MSE(r[:,:-1,:] - r_t+2))

   # KL loss
   for idx in range(num_mixture):
      g_logmix, g_mean, g_logstd = s_logmix[:,idx], s_mean[:,idx], log_logstd[:,idx]
      KL_Loss += KL(g_logmix, g_mean, g_logstd)
   KL_loss = mean(log(1 / (KL_Loss + 1e-10) + 1e-10))

   # Sparsity constraints
   Reg_loss = mean(C_so) + mean(C_sr) + mean(C_ar) + mean(C_ss) + mean(C_theta_oo) +
        mean(C_theta_rr) + mean(C_theta_ss) + mean(theta_o) + mean(theta_s) + mean(
        theta_r)

   # Optimization step
   loss = Rec_loss + Pred_loss + KL_loss + Reg_loss
   loss.backward()
   optimizer.step()
```

A6 EXPERIMENTAL DETAILS

A6.1 HYPERPARAMETERS SELECTIONS

Model estimation We use a random policy to collect sequence data from source domains. For both modified Cartpole and Pong experiments, the sequence length is 40 and the number of sequence is 10000 for each domain. The sampling resolution is set to be 0.02. Other details are summarized in Table A16.

Settings	Cartpole	Pong
# Dimensions of latent space	20	25
# Dimensions of θ	1	Size & noise: 1, orientation: 2, color: 3, Reward: 1 (linear), 2 (non-linear)
# Epochs	1000	reward-varying: 4000, others: 1500
Batch size	20	80
# RNN cells	256	256
Initial learning rate	0.01	0.01
Learning rate decay rate	0.999	0.999
Dropout	0.90	0.90
KL-tolerance	0.50	0.50

Table A16: Experimental details on the model estimation part.

Policy learning We adopt Double DQN (Van Hasselt et al., 2016) during policy learning stage. The detailed hyper-parameters are summarized in Table A17. For a fair comparison, we use the same set of hyperparameters for training other baseline methods.

Settings	Cartpole	Pong
Discount factor	0.99	0.99
Exploration rate	1.0	1.0
Initial learning rate	0.01	0.01
Learning rate decay rate	0.999	0.999
Dropout	0.10	0.10

Table A17: Experimental details on the policy learning part.

A6.2 EXPERIMENTAL PLATFORMS

For the model estimation, Cartpole and Pong experiments are implemented on 1 NVIDIA P100 GPUs and 4 NVidia V100 GPUs, respectively. The policy learning stages in both experiments are implemented on 8 Nvidia RTX 1080Ti GPUs.

A6.3 LICENSES

In our code, we have used the following libraries which are covered by the corresponding licenses:

- Tensorflow (Apache License 2.0),
- Pytorch (BSD 3-Clause "New" or "Revised" License),
- OpenAI Gym (MIT License),
- OpenCV (Apache 2 License),
- Numpy (BSD 3-Clause "New" or "Revised" License)
- Keras (Apache License).

We plan to release our code under the MIT License.

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, 2021a. URL https://openreview.net/forum?id=qda7-sVg84.

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in Neural Information Processing Systems*, 2021b.

Ron Amit and Ron Meir. Meta-learning by adjusting priors based on extended pac-bayes theory. In *International Conference on Machine Learning*, pages 205–214, 2018.

C. M. Bishop. Mixture density networks. In *Technical Report NCRG/4288, Aston University, Birmingham, UK*, 1994.

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

Bradley Efron. Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics*, pages 569–593. Springer, 1992.

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

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.

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.

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

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

J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge, 2000.

Anastasia Pentina and Christoph Lampert. A pac-bayesian bound for lifelong learning. In *International Conference on Machine Learning*, pages 991–999, 2014.

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.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. Spring-Verlag Lectures in Statistics, 1993.

Emanuel Todorov, Tom Erez, and Yuval 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.

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.

Mingzhang Yin, George Tucker, Mingyuan Zhou, Sergey Levine, and Chelsea Finn. Meta-learning without memorization. *arXiv preprint arXiv:1912.03820*, 2019.

Kun Zhang, Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. Kernel-based conditional independence test and application in causal discovery. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, UAI'11, pages 804–813, Arlington, Virginia, USA, 2011. AUAI Press. ISBN 9780974903972.

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.