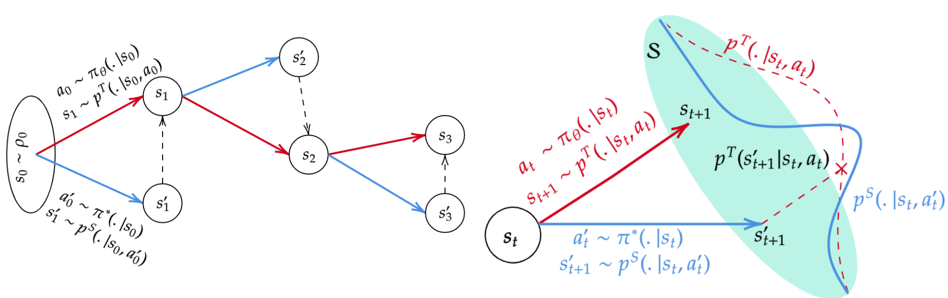
Adaptive Policy Transfer in Reinforcement Learning Supplementary

Girish Joshi
Department of Aerospace Engineering & Department of Computer Science
University of Illinois Urbana Champaign
Urbana 61802
girishj2@illinois.edu.edu, girishc@illinois.edu

A Related work

Deep Reinforcement Learning (D-RL) has recently enabled agents to learn policies for complex robotic tasks in simulation [1, 2, 3, 4]. However, D-RL has been plagued by the curse of sample complexity. Therefore, the capabilities demonstrated in the simulated environment are hard to replicate in the real world. This learning inefficiency of D-RL has led to significant work in the field of Transfer Learning (TL) [5]. A significant body of literature on transfer in RL is focused on initialized RL in the target domain using learned source policy; known as jump-start/warm-start methods [6, 7, 8]. Some examples of these transfer architectures include transfer between similar tasks [9], transfer from human demonstrations [10] and transfer from simulation to real [11, 12, 13]. Efforts have also been made in exploring accelerated learning directly on real robots, through Guided Policy Search (GPS) [14] and parallelizing the training across multiple agents using meta-learning [15, 16, 17]. Sim-to-Real transfers have been widely adopted in the recent works and can be viewed as a subset of same domain transfer problems. Daftry et al. [18] demonstrated the policy transfer for control of aerial vehicles across different vehicle models and environments. Policy transfer from simulation to real using an inverse dynamics model estimated interacting with the real robot is presented by [19]. The agents trained to achieve robust policies across various environments through learning over an adversarial loss is presented in [20].

B Derivation of Behavioral Adaptation KL Divergence Intrinsic Reward



(a) (b)

Figure 1: (a) Target Trajectory under policy π_θ and local trajectory deviation under source optimal policy π^* and source transition p^S (b) One step Target and Source transition(simulated) starting from state s_t. Transition likelihood $p^T(s'_{t+1}|s_t, a_t)$ is the probability of landing in state s'_{t+1} starting from state s_t and using action target a_t under target transition model.

Preprint. Under review.

The adaptation objective can be formalized as minimizing the average KL-divergence [21] between source and target transition trajectories as follows,

$$\begin{aligned} \eta_{KL}(\pi_\theta, \pi^*) &= D_{KL}(p_{\pi_\theta}(\tau) \| q_{\pi^*}(\tau)), \\ \eta_{KL}(\pi_\theta, \pi^*) &= \int_{\mathcal{S},\mathcal{A}} p_{\pi_\theta}(\tau) \log \left(\frac{p_{\pi_\theta}(\tau)}{q_{\pi^*}(\tau)} \right) d\tau \end{aligned} \tag{1}$$

where $\tau = (s_0, s_1, s_2, \ldots)$ is the trajectory in the target domain under the policy $\pi_\theta(.|s)$ defined as collection states visited starting from state $s_0 \sim \rho_0$ and making transitions under target transition model $p^T(.|s_t, a_t)$.

In the above defined KL divergence term the random variable is the trajectory $\tau = (s_0, s_1, s_2, \ldots)$. We explain the flow of the algorithm in Figure 1. The algorithm starts with some random state $s_0 \sim \rho_0$ and using source optimal policy $\pi_\theta(.|s_0)$ and target transition model $p^T(.|s_0, \pi_\theta(s_0))$, make a transition to state s_1 (Red arrow, Figure-1a). The source simulator is now initialized to state s_0 and using source optimal policy $\pi^*(.|s_0)$ we make optimal transition under source transition model $p^S(.|s_0, \pi^*(s_0))$ to state s'_1 (Blue arrow, this is a simulated step using source simulator Figure-1a).

The likelihood of landing in the reference state s'_1(obtained from optimal source transition) is now evaluated, under target transition model and target policy. We call this likelihood as trajectory deviation likelihood. The trajectory deviation likelihood can be expressed as $p^T(s'_1|s_0, \pi_\theta(s_0))$. Note that we obtain this likelihood by evaluation the target transition probability at state s'_1. For the next step of learning the source model is reinitialized to the target transitioned state s_1 (dotted arrow in Figure-1b and the process is repeated as above.

One step KL divergence between transition probabilities or one-step Intrinsic reward can be written as

$$\zeta_t = \pi_\theta(a_0|s_0) p^T(s'_1|s_0, \pi_\theta(s_0)) \left(\frac{\pi_\theta(a_0|s_0) p^T(s'_1|s_0, \pi_\theta(s_0))}{\pi^* a(a'_0|s_0) p^S(s'_1|s_0, \pi^*(s_0))} \right). \tag{2}$$

Note that above expression is a proper definition of KL divergence where the random variable for two probabilities $p_{\pi_\theta}(.)$ and $q_{\pi^*}(.)$ is the trajectory $\tau = (s_0, s_1, s_2, \ldots)$. The KL divergence expression also satisfies the absolute continuity condition owing to fact that the state space for source and target are same.

Computing the KL divergence over the entire trajectory, we can derive the expression for total behavioral adaptation intrinsic return as follows

$$\int_\tau p_{\pi_\theta}(\tau) \log \left(\frac{p_{\pi_\theta}(\tau)}{q_{\pi^*}(\tau)} \right) d\tau = \mathop{\mathbb{E}}_{s_t \sim \tau} \left(\log \left(\frac{\rho(s_0) \pi(a_0|s_0) p^T(s'_1|s_0, \pi_\theta(s_0)) \ldots}{\rho(s_0) \pi^*(a'_0|s_0) p^S(s'_1|s_0, \pi^*(s_0)) \ldots} \right) \right). \tag{3}$$

C Total return gradient with respect to policy parameters

The total return which we aim to maximize in adapting the source policy to target is the mixture of environmental rewards and Intrinsic KL divergence reward as follows,

$$\bar{\eta}_{KL,\beta}(\pi_\theta, \pi^*) = \mathop{\mathbb{E}}_{s_t, a_t \sim \tau} \left(p_{\pi_\theta}(\tau) \sum_{t=0}^{\mathcal{H}} r'_t \right), \tag{4}$$

Taking the expectation over policy and transition distribution we can write the above expression

$$\bar{\eta}_{KL,\beta}(\pi_\theta, \pi^*) = \mathop{\mathbb{E}}_{s_t \sim p^T, a_t \sim \pi_\theta} \left(\sum_{t=0}^{\infty} \gamma^t r'_t \right) = V^{\pi_\theta}(s). \tag{5}$$

Using the definition of the state-value function, the above objective function can be re-written as

$$\bar{\eta}_{KL,\beta}(\pi_\theta, \pi^*) = \sum_a \left(\pi_\theta(a|s) Q^{\pi_\theta}(s, a) \right). \tag{6}$$

The adaptive policy update methods work by computing an estimator of the gradient of the return and plugging it into a stochastic gradient ascent algorithm

$$\pi_\theta^{*T} = arg \max_{\pi_\theta \in \Pi} P_{Z^n}(\bar{\eta}_{KL,\beta}). \tag{7}$$

$$\theta = \theta + \alpha \hat{g},$$

where α is the learning rate and $\hat{g}$ is the empirical estimate of the gradient of the total discounted return $\eta_{_{KL}}$.

Taking the derivative of the total return term

$$\begin{aligned} \nabla_\theta(\bar{\eta}_{_{KL,\beta}}) &= \nabla_\theta V^{\pi_\theta}(s) = \nabla_\theta \left(\sum_a \left(\pi_\theta(a|s) Q^{\pi_\theta}(s,a) \right) \right), \\ \nabla_\theta V^{\pi_\theta}(s) &= \sum_a \nabla_\theta \pi_\theta(a|s) Q^{\pi_\theta}(s,a) + \sum_a \pi_\theta(a|s) \nabla_\theta Q^{\pi_\theta}(s,a). \end{aligned} \tag{8}$$

Using the following definition in above expression,

$$Q^{\pi_\theta}(s_i, a) = p^T(s_{i+1}, |s_i, a)(r + \gamma V_\theta^\pi(s_{i+1})).$$

We can rewrite the gradient to total return over policy π_θ as,

$$\begin{aligned} \nabla_\theta V^{\pi_\theta}(s_0) &= \sum_a \nabla_\theta \pi_\theta(a|s_0) Q^{\pi_\theta}(s_0, a) \\ &+ \sum_{s_1} p^T(s_1, |s_0, a) \sum_a \pi_\theta(a|s_0) \nabla_\theta(r_0 + \gamma V_\theta^\pi(s_1)). \end{aligned} \tag{9}$$

As the reward r_t is independent of θ, we can simplify the above expression and can be re-written as

$$\begin{aligned} \nabla_\theta V^{\pi_\theta}(s_0) &= \sum_a \nabla_\theta \pi_\theta(a|s_0) Q^{\pi_\theta}(s_0, a) \\ &+ \sum_{s_1} \gamma p^T(s_1, |s_0, a) \sum_a \pi_\theta(a|s_0) \nabla_\theta V_\theta^\pi(s_1). \end{aligned} \tag{10}$$

As we can see the above expression has a recursive property involving term $\nabla_\theta V^{\pi_\theta}(s)$. Using the following definition of a discounted state visitation distribution d^{π_θ}

$$\begin{aligned} d^{\pi_\theta}(s_0) &= \rho(s_0) + \gamma \sum_a \pi(a|s_0) \sum_{s_1} p^T(s_1|s_0, a) \\ &\quad + \gamma^2 \sum_a \pi(a|s_1) \sum_{s_2} p^T(s_2|s_1, a) \ldots \end{aligned} \tag{11}$$

we can write the gradient of transfer objective as follows,

$$\nabla_\theta(\eta_{_{KL,\beta}}) = \sum_{s \in \mathcal{S}} d^{\pi_\theta}(s) \sum_{a \in \mathcal{A}} \nabla_\theta \pi_\theta(a|s) Q^{\pi_\theta}(s,a) \tag{12}$$

Considering an off-policy RL update, where π_{θ^-} is used for collecting trajectories over which the state-value function is estimated, we can rewrite the above gradient for offline update as follows,

Multiplying and dividing Eq-12 by $\pi_{\theta^-}(a|s)$ and $\pi_\theta(a|s)$ we form a gradient estimate for offline update,

$$= \sum_{s \in \mathcal{S}} d^{\pi_{\theta^-}}(s) \sum_{a \in \mathcal{A}} \pi_{\theta^-}(a|s) \frac{\pi_\theta(a|s)}{\pi_{\theta^-}(a|s)} \frac{\nabla_\theta \pi_\theta(a|s)}{\pi_\theta(a|s)} Q^{\pi_{\theta^-}}(s,a) \tag{13}$$

where the ratio $\left(\frac{\pi_\theta(a|s)}{\pi_{\theta^-}(a|s)}\right)$ is importance sampling term, and using the following identity the above expression can be rewritten as

$$\begin{aligned} \frac{\nabla_\theta \pi_\theta(a|s)}{\pi_\theta(a|s)} &= \nabla_\theta \log \pi_\theta(a|s) \\ &= \mathop{\mathbb{E}}_{s_t \sim d^{\pi_{\theta^-}}, a_t \sim \pi_{\theta^-}} \left(\frac{\pi_\theta(a|s)}{\pi_{\theta^-}(a|s)} Q^{\pi_{\theta^-}}(s,a) \nabla_\theta \log \pi_\theta(a|s) \right). \end{aligned} \tag{14}$$

D Theoretical bounds on sample complexity

Although there is some empirical evidence that transfer can improve performance in subsequent reinforcement-learning tasks, there are not many theoretical guarantees in the literature. Since many of the existing transfer algorithms approach the problem of transfer as a method of providing good initialization to target task RL, we can expect the sample complexity of those algorithms to still be a function of the cardinality of state-action pairs $|N| = |\mathcal{S}| \times |\mathcal{A}|$. On the other hand, in a supervised learning setting, the theoretical guarantees of the most algorithms have no dependency on size (or dimensionality) of the input domain (which is analogous to $|N|$ in RL). Having formulated a policy transfer algorithm using labeled reference trajectories derived from optimal source policy, we construct supervised learning like PAC property of the proposed method. For deriving, the lower bound on the sample complexity of the proposed transfer problem, we consider only the adaptation part of the learning i.e., the case when $\beta = 1$. This is because, in ATL, adaptive learning is akin to supervised learning, since the source reference trajectories provide the target states given every (s_t, a_t) pair.

Suppose we are given the learning problem specified with training set $Z^n = (Z1, \dots Z_n)$ where each $Z_i = (\{s_i, a_i\})_{i=0}^{n}$ are independently drawn trajectories according to some distribution P. Given the data Z^n we can compute the empirical return $P_{Z^n}(\bar{\eta}_{KL,\beta})$ for every $\pi_\theta \in \Pi$, we will show that the following holds:

$$\|P_{Z^n}(\bar{\eta}_{KL,\beta}) - P(\bar{\eta}_{KL,\beta})\| \leq \epsilon. \tag{15}$$

with probability at least $1 - \delta$, for some very small δ s.t $0 \leq \delta \leq 1$. We can claim that the empirical return for all π_θ is a sufficiently accurate estimate of the true return function. Thus a reasonable learning strategy is to find a $\pi_\theta \in \Pi$ that would minimize empirical estimate of the objective

$$\pi_\theta^{*T} = \arg\max_{\pi_\theta \in \Pi, \beta} \left(\bar{\eta}_{KL,\beta}\right), \tag{16}$$

Theorem D.1 *If the induced class $\mathcal{L}_\Pi$ has uniform convergence in empirical mean property then empirical risk minimization is PAC.*

For notation simplicity we drop the superscript T (for Target domain) and subscript θ (policy parameters) in further analysis. Unless stated we will using following simplifications $\hat{\pi}^* = \hat{\pi}_\theta^{*T}$ and $\pi^* = \pi_\theta^{*T}$

Proof Fix $\epsilon, \delta > 0$ we will show that for sufficiently large $n \geq n(\epsilon, \delta)$

$$P^n(P(\bar{\eta}_{KL,\hat{\pi}^*}) - P(\bar{\eta}_{KL,\pi^*}) \geq \epsilon) \leq \delta \tag{17}$$

Let $\pi^* \in \Pi$ be the minimizer of true return $P(\bar{\eta}_{KL})$, further adding and subtracting the terms $P_{Z^n}(\bar{\eta}_{KL,\hat{\pi}^*})$ and $P_{Z^n}(\bar{\eta}_{KL,\pi^*})$ we can write

$$\begin{aligned} P(\bar{\eta}_{KL,\hat{\pi}^*}) - P(\bar{\eta}_{KL,\pi^*}) = \\ & P(\bar{\eta}_{KL,\hat{\pi}^*}) - P_{Z^n}(\bar{\eta}_{KL,\hat{\pi}^*}) \\ & + P_{Z^n}(\bar{\eta}_{KL,\hat{\pi}^*}) - P_{Z^n}(\bar{\eta}_{KL,\pi^*}) \\ & + P_{Z^n}(\bar{\eta}_{KL,\pi^*}) - P(\bar{\eta}_{KL,\pi^*}) \end{aligned} \tag{18}$$

To simplify, the three terms in the above expression can be handled individually as follows,

1. $P(\bar{\eta}_{KL,\hat{\pi}^*}) - P_{Z^n}(\bar{\eta}_{KL,\hat{\pi}^*})$
2. $P_{Z^n}(\bar{\eta}_{KL,\hat{\pi}^*}) - P_{Z^n}(\bar{\eta}_{KL,\pi^*})$
3. $P_{Z^n}(\bar{\eta}_{KL,\pi^*}) - P(\bar{\eta}_{KL,\pi^*})$

Lets consider the term $P_{Z^n}(\bar{\eta}_{KL,\hat{\pi}^*}) - P_{Z^n}(\bar{\eta}_{KL,\pi^*})$ in the above expression is always negative semi-definite, since $\hat{\pi}^*$ is a maximizer wrto $P_{Z^n}(\bar{\eta}_{KL})$, hence $P_{Z^n}(\bar{\eta}_{KL,\hat{\pi}^*}) \leq P_{Z^n}(\bar{\eta}_{KL,\pi^*})$ always, i.e

$$P_{Z^n}(\bar{\eta}_{KL,\hat{\pi}^*}) - P_{Z^n}(\bar{\eta}_{KL,\pi^*}) \leq 0$$

Next the 1st term can be bounded as

$$P(\bar{\eta}_{KL,\hat{\pi}^*}) - P_{Z^n}(\bar{\eta}_{KL,\hat{\pi}^*}) \leq \sup_{\pi \in \Pi}[P_{Z^n}(\eta_{KL}) - P(\bar{\eta}_{KL})]$$

$$\leq \sup_{\pi \in \Pi} \|P_{Z^n}(\bar{\eta}_{KL}) - P(\bar{\eta}_{KL})\|$$

Similarly upper bound can be written for the 3rd term Therefore we can upper bound the above expression as

$$P(\bar{\eta}_{KL,\hat{\pi}^*}) - P(\bar{\eta}_{KL,\pi^*}) \leq 2 \sup_{\pi \in \Pi} \|P_{Z^n}(\bar{\eta}_{KL}) - P(\bar{\eta}_{KL})\|$$

From Equation-(17) we have

$$\sup_{\pi \in \Pi} \|P_{Z^n}(\bar{\eta}_{KL}) - P(\bar{\eta}_{KL})\| \geq \epsilon/2 \tag{19}$$

Using McDiarmids inequality and union bound, we can state the probability of this event as

$$P^n(\|P_{Z^n}(\bar{\eta}_{KL}) - P(\bar{\eta}_{KL})\| \geq \epsilon/2) \leq 2|\Pi| e^{-\frac{n\epsilon^2}{2C^2}} \tag{20}$$

The finite difference bound

$$C = \frac{1}{1-\gamma}$$

Equating the RHS of the expression to δ and solving for n we get

$$n(\epsilon, \delta) \geq \frac{2}{\epsilon^2(1-\gamma)^2} \log\left(\frac{2|\Pi|}{\delta}\right) \tag{21}$$

for $n \geq n(\epsilon, \delta)$ the probability of receiving a bad sample is less than δ.

E ϵ-Optimality result under Adaptive Transfer-Learning

Consider MDP M^* and $\hat{M}$ which differ in their transition models. For the sake of analysis, let M^* be the MDP with ideal transition model, such that target follows source transition p^* precisely. Let $\hat{p}$ be the transition model achieved using the estimated policy learned over data interacting with the target model and the associated MDP be denoted as $\hat{M}$. We analyze the ϵ-optimality of return under adapted source optimal policy through ATL.

Definition E.1 *Given the value function $V^* = V^{\pi^*}$ and model M_1 and M_2, which only differ in the corresponding transition models p_1 and p_2. Define $\forall s, a \in \mathcal{S} \times \mathcal{A}$*

$$d_{M_1,M_2}^{V^*} = \sup_{s,a \in \mathcal{S} \times \mathcal{A}} \left| \mathop{\mathbb{E}}_{s' \sim P_1(s,a)}[V^*(s')] - \mathop{\mathbb{E}}_{s' \sim P_2(s,a)}[V^*(s')] \right|.$$

Lemma E.2 *Given M^*, $\hat{M}$ and value function $V_{M^*}^{\pi^*}$, $V_{\hat{M}}^{\pi^*}$ the following bound holds $\left\| V_{M^*}^{\pi^*} - V_{\hat{M}}^{\pi^*} \right\|_\infty \leq \frac{\gamma\epsilon}{(1-\gamma)^2}$*

where $\max_{s,a} \|\hat{p}(.|s,a) - p^*(.|s,a)\| \leq \epsilon$ and $\hat{p}$ and p^* are transition of MDP $\hat{M}, M^*$ respectively.

The proof of this lemma is based on the simulation lemma [22] (see Supplementary document). Similar results for RL with imperfect models were reported by [23].

Lemma E.3 *Given M^*, $\hat{M}$ and value function $V_{M^*}^{\pi^*}$, $V_{\hat{M}}^{\pi^*}$ the following bound holds $\left\| V_{M^*}^{\pi^*} - V_{\hat{M}}^{\pi^*} \right\|_\infty \leq \frac{\gamma\epsilon}{(1-\gamma)^2}$*

where $\max_{s,a} \|\hat{p}(.|s,a) - p^*(.|s,a)\| \leq \epsilon$ and $\hat{p}$ and p^* are transition of MDP $\hat{M}, M^*$ respectively.

Proof For any $s \in \mathcal{S}$

$$
\begin{aligned}
&|V_{\hat{M}}^{\pi^*}(s) - V_{M^*}^{\pi^*}(s)|_\infty \\
&= |r(s,a) + \gamma \left\langle \hat{p}(s'|s,a), V_{\hat{M}}^{\pi^*}(s') \right\rangle \\
&-r(s,a) - \gamma \left\langle p^*(s'|s,a), V_{M^*}^{\pi^*}(s') \right\rangle |_\infty
\end{aligned}
$$

Add and subtract the term $\gamma \left\langle p^*(s'|s,a), V_{\hat{M}}^{\pi^*}(s') \right\rangle$

$$
\begin{aligned}
= \quad & |\gamma \left\langle \hat{p}(s'|s,a), V_{\hat{M}}^{\pi^*}(s') \right\rangle - \gamma \left\langle p^*(s'|s,a), V_{\hat{M}}^{\pi^*}(s') \right\rangle \\
+ \; & \gamma \left\langle p^*(s'|s,a), V_{\hat{M}}^{\pi^*}(s') \right\rangle - \gamma \left\langle p^*(s'|s,a), V_{M^*}^{\pi^*}(s') \right\rangle |_\infty \\
\leq \; & \gamma| \left\langle \hat{p}(s'|s,a), V_{\hat{M}}^{\pi^*}(s') \right\rangle - \left\langle p^*(s'|s,a), V_{\hat{M}}^{\pi^*}(s') \right\rangle | \\
+ \; & \gamma| \left\langle p^*(s'|s,a), V_{\hat{M}}^{\pi^*}(s') \right\rangle - \gamma \left\langle p^*(s'|s,a), V_{M^*}^{\pi^*}(s') \right\rangle |_\infty \\
\leq \; & \gamma|\hat{p}(s'|s,a) - p^*(s'|s,a)|_\infty |V_{\hat{M}}^{\pi^*}(s')|_\infty \\
& +\gamma|V_{\hat{M}}^{\pi^*}(s) - V_{M^*}^{\pi^*}(s)|_\infty
\end{aligned}
$$

Using the definition of ϵ in above expression, we can write

$$
|V_{\hat{M}}^{\pi^*}(s) - V_{M^*}^{\pi^*}(s)|_\infty \leq \gamma\epsilon|V_{\hat{M}}^{\pi^*}(s')|_\infty + \gamma|V_{\hat{M}}^{\pi^*}(s) - V_{M^*}^{\pi^*}(s)|_\infty
$$

Therefore

$$
|V_{\hat{M}}^{\pi^*}(s) - V_{M^*}^{\pi^*}(s)|_\infty \leq \frac{\gamma\epsilon|V_{\hat{M}}^{\pi^*}(s')|_\infty}{1-\gamma}
$$

Now we solve for expression $|V_{\hat{M}}^{\pi^*}(s')|_\infty$. We know that this term is bounded as

$$
|V_{\hat{M}}^{\pi^*}(s')|_\infty \leq \frac{R_{max}}{1-\gamma}
$$

where $R_{max} = 1$, therefore we can write the complete expression as

$$
|V_{\hat{M}}^{\pi^*}(s) - V_{M^*}^{\pi^*}(s)|_\infty \leq \frac{\gamma\epsilon}{(1-\gamma)^2}
$$

Env	Property	source	Target	%Change
Hopper	Floor Friction	1.0	2.0	+100%
HalfCheetah	gravity	-9.81	-15	+52%
	Total Mass	14	35	+150%
	Back-Foot Damping	3.0	1.5	-100%
	Floor Friction	0.4	0.1	-75%
Walker2d	Density	1000	1500	+50%
	Right-Foot Friction	0.9	0.45	-50%
	Left-Foot Friction	1.9	1.0	-47.37%

Table 1: Transition Model and environment properties for Source and Target task and % change

F Learning the Mixing Coefficient β

A hierarchical update of the mixing coefficient β is carried out over n-test trajectories, collected using the updated policy network $\pi_{\theta'}(a|s)$. The mixing coefficient β is learnt by optimizing the return over trajectory as follows,

$$
\beta = arg \max_{\beta}(\bar{\eta}_{KL,\beta}(\pi_{\theta'}, \pi^*))
$$

	Hopper	Walker2d	HalfCheetah
State Space	12	18	17
Control Space	3	6	6
Number of layers	3	3	3
Layer Activations	tanh	tanh	tanh
Total num. of network params	10530	28320	26250
Discount	0.995	0.995	0.995
Learning rate (α)	1.5×10^{-5}	8.7×10^{-6}	9×10^{-6}
β initial Value	0.5	0.5	0.5
β-Learning rate ($\bar{\alpha}$)	0.1	0.1	0.1
Batch size	20	20	5
Policy Iter	3000	5000	1500

Table 2: Policy Network details and Network learning parameter details

where θ' is parameter after the policy update step.

$$\beta = arg \max_{\beta} \mathop{\mathbb{E}}_{s_t, a_t \sim \tau} \left(p_{\pi_\theta}(\tau) \sum_{t=1}^{\infty} \gamma^t r'_t \right)$$

We can use gradient ascent to update parameter β in direction of optimizing the reward mixing as follows,

$$\beta \leftarrow \beta + \bar{\alpha} \nabla_\beta (\bar{\eta}_{KL,\beta}(\pi_{\theta'}, \pi^*)).$$

Using the definition of mixed reward as $r'_t = (1-\beta) r_t - \beta \zeta_t$, we can simplify the above gradient as,

$$\beta \leftarrow \beta + \bar{\alpha} \mathop{\mathbb{E}}_{s_t, a_t \sim \tau} \left(p_{\pi_\theta}(\tau) \sum_{t=1}^{\infty} \gamma^t \nabla_\beta (r'_t) \right)$$

$$\beta \leftarrow \beta + \bar{\alpha} \mathop{\mathbb{E}}_{s_t, a_t \sim \tau} \left(\sum_{t=1}^{\infty} \gamma^t (r_t - \zeta_t) \right).$$

We use stochastic gradient ascent to update the mixing coefficient β as follows

$$\beta \leftarrow \beta + \bar{\alpha} \hat{g}_\beta, \quad s.t \;\; 0 \leq \beta \leq 1.$$

where $\bar{\alpha}$ is the learning rate and $\hat{g}_\beta = P_{Z^n_{test}}(\nabla_\beta \bar{\eta}_{_{KL,\beta}})$ is the empirical estimate of the gradient of the total return $\bar{\eta}_{_{KL,\beta}}(\pi_{\theta'}, \pi^*)$. The gradient estimate $\hat{g}_\beta$ over data ($Z^n_{test} : \{s_i, a_i, a'_i\}_i^T$) is computed as follows,

$$\hat{g}_\beta = \frac{1}{N} \sum_{i=1}^{N} \left(\sum_{t=1}^{\mathcal{H}} \gamma^t (r_t - \zeta_t) \right)$$

where $\mathcal{H}$ truncated trajectory length from experiments.

As we can see the gradient of objective with respect to mixing coefficient β is an average over difference between environmental and intrinsic rewards. If $r_t - \zeta_t \geq 0$ the update will move parameter β towards favoring learning through exploration more than learning through adaptation and visa versa.

As β update is a constrained optimization with constraint $0 \leq \beta \leq 1$. We handle this constrained optimization by modelling β as output of Sigmoidal network parameterized by parameters ϕ.

$$\beta = \sigma(\phi)$$

And the constrained optimization can be equivalently written as optimizing w.r.to ϕ as follows

$$\phi \leftarrow \phi + \bar{\alpha} \hat{g}_\beta \nabla_\phi(\beta), \quad where \;\; \beta = \sigma(\phi)$$

The reward mixing co-efficient β learned for HalfCheetah, Hopper and Walker2d envs is provided in Figure-2. For all the experiments we start with $\beta = 0.5$ that is placing equal probability of learning through adaptation and learning through exploration. As we can observe the reward mixing leans towards learning through adaptation for HalfCheetah and Hopper envs. Whereas, as for Walker2d the beta initially believes learning from exploration more, but quickly leans toward learning from source policy and adaptation.

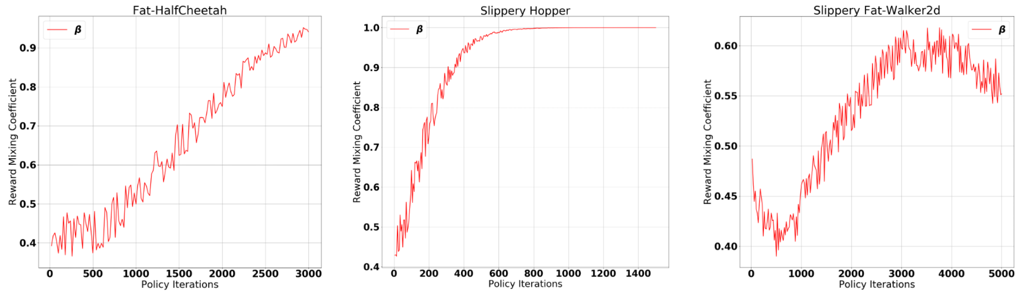


Figure 2: The Reward Mixing Co-efficient β for HalfCheetah, Hopper and Walker2d environment learnt over trajectories collected interacting with envs.

References

[1] Xue Bin Peng, Glen Berseth, and Michiel Van de Panne. Terrain-adaptive locomotion skills using deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 35(4):81, 2016.

[2] Xue Bin Peng, Glen Berseth, KangKang Yin, and Michiel Van De Panne. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 36(4):41, 2017.

[3] Libin Liu and Jessica Hodgins. Learning to schedule control fragments for physics-based characters using deep q-learning. *ACM Transactions on Graphics (TOG)*, 36(3):29, 2017.

[4] Nicolas Heess, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, Ali Eslami, Martin Riedmiller, et al. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286*, 2017.

[5] Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(Jul):1633–1685, 2009.

[6] Matthew E Taylor, Peter Stone, and Yaxin Liu. Value functions for rl-based behavior transfer: A comparative study. In *Proceedings of the National Conference on Artificial Intelligence*, volume 20, page 880. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

[7] Haitham B Ammar, Karl Tuyls, Matthew E Taylor, Kurt Driessens, and Gerhard Weiss. Reinforcement learning transfer via sparse coding. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 383–390. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

[8] Haitham Bou Ammar, Eric Eaton, Paul Ruvolo, and Matthew E Taylor. Unsupervised cross-domain transfer in policy gradient reinforcement learning via manifold alignment. In *Proc. of AAAI*, 2015.

[9] Bikramjit Banerjee and Peter Stone. General game learning using knowledge transfer. In *IJCAI*, pages 672–677, 2007.

[10] Jan Peters and Stefan Schaal. Policy gradient methods for robotics. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 2219–2225. IEEE, 2006.

[11] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. *arXiv preprint arXiv:1710.06537*, 2017.

[12] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635, 2011.

[13] Mengyuan Yan, Iuri Frosio, Stephen Tyree, and Jan Kautz. Sim-to-real transfer of accurate grasping with eye-in-hand observations and continuous control. *arXiv preprint arXiv:1712.03303*, 2017.

[14] Sergey Levine, Nolan Wagener, and Pieter Abbeel. Learning contact-rich manipulation skills with guided policy search. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 156–163. IEEE, 2015.

[15] Sergey Levine, Peter Pastor, Alex Krizhevsky, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with large-scale data collection. In *International Symposium on Experimental Robotics*, pages 173–184. Springer, 2016.

[16] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566. IEEE, 2018.

[17] Henry Zhu, Abhishek Gupta, Aravind Rajeswaran, Sergey Levine, and Vikash Kumar. Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost. *arXiv preprint arXiv:1810.06045*, 2018.

[18] Shreyansh Daftry, J Andrew Bagnell, and Martial Hebert. Learning transferable policies for monocular reactive mav control. In *International Symposium on Experimental Robotics*, pages 3–11. Springer, 2016.

[19] Paul Christiano, Zain Shah, Igor Mordatch, Jonas Schneider, Trevor Blackwell, Joshua Tobin, Pieter Abbeel, and Wojciech Zaremba. Transfer from simulation to real world through learning deep inverse dynamics model. *arXiv preprint arXiv:1610.03518*, 2016.

[20] Markus Wulfmeier, Ingmar Posner, and Pieter Abbeel. Mutual alignment transfer learning. *arXiv preprint arXiv:1707.07907*, 2017.

[21] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897, 2015.

[22] Michael Kearns and Satinder Singh. Near-optimal reinforcement learning in polynomial time. *Machine Learning*, 49(2-3):209–232, 2002.

[23] Nan Jiang. Pac reinforcement learning with an imperfect model. In *Proc. of AAAI*, 2018.