Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)

Autonomous Cross-Domain Knowledge Transfer in Lifelong Policy Gradient Reinforcement Learning

Haitham Bou Ammar
Univ. of Pennsylvania
haithamb@seas.upenn.edu

Eric Eaton
Univ. of Pennsylvania
eeaton@cis.upenn.edu

José Marcio Luna
Univ. of Pennsylvania
joseluna@seas.upenn.edu

Paul Ruvolo
Olin College of Engineering
paul.ruvolo@olin.edu

Abstract

Online multi-task learning is an important capability for lifelong learning agents, enabling them to acquire models for diverse tasks over time and rapidly learn new tasks by building upon prior experience. However, recent progress toward lifelong reinforcement learning (RL) has been limited to learning from within a single task domain. For truly versatile lifelong learning, the agent must be able to autonomously transfer knowledge between different task domains. A few methods for cross-domain transfer have been developed, but these methods are computationally inefficient for scenarios where the agent must learn tasks consecutively.

In this paper, we develop the first cross-domain lifelong RL framework. Our approach efficiently optimizes a shared repository of transferable knowledge and learns projection matrices that specialize that knowledge to different task domains. We provide rigorous theoretical guarantees on the stability of this approach, and empirically evaluate its performance on diverse dynamical systems. Our results show that the proposed method can learn effectively from interleaved task domains and rapidly acquire high performance in new domains.

1 Introduction

Reinforcement learning (RL) provides the ability to solve high-dimensional control problems when detailed knowledge of the system is not available *a priori*. Applications with these characteristics are ubiquitous in a variety of domains, from robotic control [Busoniu *et al.*, 2008; Smart & Kaelbling, 2002] to stock trading [Dempster & Leemans, 2006]. However, in many cases, RL methods require numerous lengthy interactions with the dynamical system in order to learn an acceptable controller. Unfortunately, the cost of acquiring these interactions is often prohibitively expensive (in terms of time, expense, physical wear on the robot, etc.).

This issue of obtaining adequate experience only worsens when *multiple* control problems must be solved. This need arises in two main cases: (1) when a single agent must learn to solve multiple tasks (e.g., a reconfigurable robot), and (2) when different robots must each learn to solve a single control problem. *Transfer learning* [Taylor & Stone, 2009] and *multi-task learning* (MTL) methods [Li *et al.*, 2009; Lazaric & Ghavamzadeh, 2010] have been developed to reduce the amount of experience needed for individual tasks by allowing the agent to reuse knowledge from other tasks.

In both transfer learning and MTL, the difficulty of transferring knowledge between tasks increases with the diversity of the tasks. In the extreme case, the underlying systems (and their action and state representations) are entirely different, making direct knowledge reuse impossible. Several approaches to this *cross-domain transfer* problem have been developed, but these methods require an inter-task mapping of state/action spaces that is either hand-coded [Taylor *et al.*, 2007] or learned in a computationally inefficient manner that does not scale to more than a few task domains (see Section 3). However, the problem of cross-domain transfer has not yet been studied in lifelong learning settings [Thrun & O'Sullivan, 1996; Ruvolo & Eaton, 2013], in which the agent must learn multiple tasks consecutively with the goal of optimizing performance across all previously learned tasks.

We address this problem by developing the first algorithm for lifelong RL that supports efficient and autonomous cross-domain transfer between multiple consecutive tasks from different domains. Specifically, our approach provides the following advantages over existing approaches: (1) it improves current cross-domain transfer learning methods by optimizing performance across all tasks, (2) it learns multi-task cross-domain mappings autonomously, and (3) it can share knowledge between a multitude of tasks from diverse domains in a computationally efficient manner. To enable effective cross-domain lifelong learning, our approach learns a repository of shared knowledge along with projection matrices that specialize this shared knowledge to each task domain. We provide theoretical guarantees on convergence that show this approach becomes increasingly stable as the number of domains or tasks grows large, and demonstrate the empirical effectiveness of cross-domain lifelong learning between streams of interleaved tasks from diverse dynamical systems, including bicycles and helicopters.

2 Background on Policy Gradient RL

In a reinforcement learning (RL) problem, an agent must decide how to sequentially select actions to maximize its ex-

pected return. In contrast to classic stochastic optimal control methods [Bertsekas, 1995], RL approaches do not require detailed prior knowledge of the system dynamics or goal; instead these approaches learn optimal control policies through interaction with the system itself. RL problems are typically formalized as a Markov decision process (MDP) $\langle \mathcal{X}, \mathcal{A}, P, R, \gamma \rangle$, where $\mathcal{X} \subseteq \mathbb{R}^d$ is the (potentially infinite) set of states, $\mathcal{A}$ is the set of actions that the agent may execute, $P : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \to [0, 1]$ is a state transition probability function describing the task dynamics, $R : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \to \mathbb{R}$ is the reward function measuring the performance of the agent, and $\gamma \in [0, 1)$ is the reward discount factor. At each time step m, the agent is in state $\boldsymbol{x}_m \in \mathcal{X}$ and must choose an action $\boldsymbol{a}_m \in \mathcal{A}$, transitioning it to a new state $\boldsymbol{x}_{m+1} \sim P(\boldsymbol{x}_{m+1} \mid \boldsymbol{x}_m, \boldsymbol{a}_m)$ and yielding a reward $r_{m+1} = R(\boldsymbol{x}_m, \boldsymbol{a}_m, \boldsymbol{x}_{m+1})$. The sequence of state-action pairs forms a trajectory $\boldsymbol{\tau} = [\boldsymbol{x}_{1:M}, \boldsymbol{a}_{1:M}]$ over a (possibly infinite) horizon M. A policy $\pi : \mathcal{X} \times \mathcal{A} \to [0, 1]$ specifies a conditional probability distribution over actions given the current state. The RL agent's goal is to find a policy $\pi^\star$ that maximizes the expected per-time-step reward.

Policy gradient methods [Peters *et al.*, 2005; Sutton *et al.*, 1999] represent the agent's policy π as a function defined over a vector $\boldsymbol{\theta} \in \mathbb{R}^d$ of control parameters. With this parameterized policy, we can compute the optimal parameters $\boldsymbol{\theta}^\star$ that maximize the expected average reward:

$$\mathcal{J}(\boldsymbol{\theta}) = \mathrm{E}\left[\frac{1}{M}\sum_{m=1}^{M} r_{m+1}\right] = \int_{\mathbb{T}} p_{\boldsymbol{\theta}}(\boldsymbol{\tau})\mathfrak{R}(\boldsymbol{\tau})d\boldsymbol{\tau} \ , \qquad (1)$$

where $\mathbb{T}$ is the set of all possible trajectories, $\mathfrak{R}(\boldsymbol{\tau}) = \frac{1}{M}\sum_{m=1}^{M} r_{m+1}$ is the reward of trajectory $\boldsymbol{\tau}$, and $p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) = P_0(\mathbf{x}_1)\prod_{m=1}^{M} P(\mathbf{x}_{m+1} \mid \mathbf{x}_m, \boldsymbol{a}_m)\pi_{\boldsymbol{\theta}}(\boldsymbol{a}_m \mid \mathbf{x}_m)$ is the probability of $\boldsymbol{\tau}$ with initial state distribution $P_0 : \mathcal{X} \to [0, 1]$.

To maximize $\mathcal{J}(\cdot)$, most policy gradient algorithms (e.g., episodic REINFORCE [Williams, 1992], PoWER [Rückstieß *et al.*, 2008], and Natural Actor Critic [Peters & Schaal, 2008]) employ standard supervised function approximation to learn $\boldsymbol{\theta}$ by maximizing a lower bound on $\mathcal{J}(\boldsymbol{\theta})$. To maximize this lower bound, these methods generate trajectories using the current policy $\pi_{\boldsymbol{\theta}}$, and then compare the result with a new policy $\pi_{\tilde{\boldsymbol{\theta}}}$. Kober & Peters [2011] describe how this lower bound on the expected return can be attained using Jensen's inequality and the concavity of the logarithm:

$$\begin{aligned}\log \mathcal{J}\left(\tilde{\boldsymbol{\theta}}\right) &= \log \int_{\mathbb{T}} \frac{p_{\boldsymbol{\theta}}(\boldsymbol{\tau})}{p_{\boldsymbol{\theta}}(\boldsymbol{\tau})}\, p_{\tilde{\boldsymbol{\theta}}}(\boldsymbol{\tau})\, \mathfrak{R}(\boldsymbol{\tau})\, \mathrm{d}\boldsymbol{\tau} \\ &\geq \int_{\mathbb{T}} p_{\boldsymbol{\theta}}(\boldsymbol{\tau})\, \mathfrak{R}(\boldsymbol{\tau}) \log \frac{p_{\tilde{\boldsymbol{\theta}}}(\boldsymbol{\tau})}{p_{\boldsymbol{\theta}}(\boldsymbol{\tau})}\, \mathrm{d}\boldsymbol{\tau} + \text{constant} \\ &\propto -\mathfrak{D}_{\mathrm{KL}}\big(p_{\boldsymbol{\theta}}(\boldsymbol{\tau})\, \mathfrak{R}(\boldsymbol{\tau}) \,||\, p_{\tilde{\boldsymbol{\theta}}}(\boldsymbol{\tau})\big) = \mathcal{J}_{\mathcal{L},\boldsymbol{\theta}}\left(\tilde{\boldsymbol{\theta}}\right) \ ,\end{aligned}$$

where $\mathfrak{D}_{\mathrm{KL}}\big(p(\boldsymbol{\tau}) \;||\; q(\boldsymbol{\tau})\big) = \int_{\mathbb{T}} p(\boldsymbol{\tau}) \log \frac{p(\boldsymbol{\tau})}{q(\boldsymbol{\tau})}\, \mathrm{d}\boldsymbol{\tau}$. Consequently, this process is equivalent to minimizing the KL divergence $\mathfrak{D}_{\mathrm{KL}}$ between $\pi_{\boldsymbol{\theta}}$'s reward-weighted trajectory distribution and the trajectory distribution $p_{\tilde{\boldsymbol{\theta}}}$ of $\pi_{\tilde{\boldsymbol{\theta}}}$.

Policy gradient methods have gained attention in the RL community in part due to their successful applications to robotics [Peters *et al.*, 2005]. While such methods have a low computational cost per update, high-dimensional problems require many updates (by acquiring new rollouts) to achieve good performance. Transfer learning and multi-task learning can reduce these data requirements and accelerate learning.

3 Related Work on RL Knowledge Transfer

Knowledge transfer between tasks has been explored in the context of transfer learning and multi-task learning. In all cases, each task t is described by an MDP $\mathcal{Z}^{(t)} = \langle \mathcal{X}^{(t)}, \mathcal{A}^{(t)}, P^{(t)}, R^{(t)}, \gamma^{(t)} \rangle$.

Transfer learning aims to improve the learning time and/or behavior of the agent on a new target task by transferring knowledge learned from one or more previous source tasks [Taylor & Stone, 2009]. However, transfer learning methods only focus on optimizing performance on the target task, and therefore are not ideal for agents that revisit earlier tasks. In contrast, *multi-task learning* (MTL) methods [Li *et al.*, 2009; Lazaric & Ghavamzadeh, 2010] optimize performance over all tasks, often by training task models simultaneously while sharing knowledge between tasks, but are computationally expensive in lifelong learning scenarios where the agent must learn tasks consecutively over time [Ruvolo & Eaton, 2013; Thrun & O'Sullivan, 1996], as we explore in this paper.

One exception is PG-ELLA [Bou Ammar *et al.*, 2014], a recent lifelong policy gradient RL algorithm that can efficiently learn multiple tasks consecutively while sharing knowledge between task policies to accelerate learning. In fact, PG-ELLA can be viewed as a special case of the algorithm we propose in this paper where learning is limited to only tasks from a single domain. MTL for policy gradients has also been explored by Deisenroth et al. [2014] through customizing a single parameterized controller to individual tasks that differ only in the reward function. Another closely related work is on hierarchical Bayesian MTL [Wilson *et al.*, 2007], which can learn RL tasks consecutively, but unlike our approach, requires discrete states and actions. Snel and Whiteson's [2014] representation learning approach is also related, but assumes all tasks share the same feature and action sets. All of these MTL methods operate on tasks from a single domain, and do not support cross-domain transfer.

To enable transfer between tasks with different state and/or action spaces, transfer learning and MTL methods require an inter-task mapping to translate knowledge between tasks. The earliest work on cross-domain transfer in RL, by Taylor et al., required a hand-coded mapping [2007] or a computationally expensive exploration of all permitted mappings [2008]. More recently, unsupervised techniques have been developed to autonomously learn inter-task mappings [Bou Ammar *et al.* 2012; 2013]. While these approaches enable autonomous cross-domain transfer, they only learn pairwise mappings between tasks and are computationally expensive, making them inapplicable for transfer among numerous tasks from different domains. In contrast to these methods, our approach provides a computationally efficient mechanism for transfer between multiple task domains, enabling cross-domain lifelong

RL. Cross-domain MTL has also been explored in a limited fashion in supervised settings [Han *et al.*, 2012].

4 Problem Definition

Previous work on policy gradients has focused on either single-task learning or MTL within a single task domain (i.e., all tasks share a common state and action space, but may differ in other aspects). We focus on the problem of learning multiple tasks consecutively, where the tasks may be drawn from different task domains. Specifically, the agent must learn a series of RL tasks $\mathcal{Z}^{(1)}, \ldots, \mathcal{Z}^{(T_{\max})}$ over its lifetime, where each task $\mathcal{Z}^{(t)}$ is an MDP and the tasks may have different state and/or action spaces. We can partition the series of RL tasks into task groups $\mathcal{G}^{(1)}, \ldots, \mathcal{G}^{(G_{\max})}$, such that all tasks within a particular group $\mathcal{G}^{(g)}$ (i.e., a set of tasks) share a common state and action space, and are generated by varying latent parameters [Konidaris & Doshi-Velez, 2014]. In our experiments, each task group corresponds to a task domain—a class of dynamical systems, such as helicopters, cart-poles, etc., each of which contains multiple tasks corresponding to multiple physical helicopters or cart-poles. However, this framework can easily generalize so that a particular group $\mathcal{G}^{(g)}$ could represent a subset of tasks from a domain, similar to task clustering frameworks [Kang *et al.*, 2011].

The agent must learn the tasks consecutively, acquiring multiple trajectories within each task before moving to the next. The tasks may be interleaved, offering the agent the opportunity to revisit earlier tasks (or task domains) for further experience, but the agent has no control over the task order. We assume that *a priori* the agent does not know the total number of tasks $T_{\max}$, their distribution, the task order, or the total number of task groups $G_{\max}$. The agent also has no prior knowledge about the inter-task mappings between tasks, and so it must also learn how to transfer knowledge between task domains in order to optimize overall performance.

The agent's goal is to learn a set of *optimal* policies $\mathbf{\Pi}^{\star} = \left\{\pi^{\star}_{\boldsymbol{\theta}^{(1)}}, \ldots, \pi^{\star}_{\boldsymbol{\theta}^{(T_{\max})}}\right\}$ with corresponding parameters $\mathbf{\Theta}^{\star} = \{\boldsymbol{\theta}^{(1)\star}, \ldots, \boldsymbol{\theta}^{(T_{\max})\star}\}$. Since tasks belong to different domains, the dimension of the parameter vectors will vary, with $\boldsymbol{\theta}^{(t)} \in \mathbb{R}^{d^{(t)}}$ where $d^{(t)}$ is the dimension of the state space $\mathcal{X}^{(t)}$. At any time, the agent may be evaluated on any previous task, and so must strive to optimize its learned policies for all tasks $\mathcal{Z}^{(1)}, \ldots, \mathcal{Z}^{(T)}$, where $T = \sum_{g=1}^{G} |\mathcal{G}^{(g)}|$ denotes the number of tasks seen so far ($1 \leq T \leq T_{\max}$) and G is the number of groups seen so far ($1 \leq G \leq G_{\max}$).

5 Cross-Domain Lifelong RL

This section develops our cross-domain lifelong RL approach. In order to share knowledge between the tasks, we assume that each task's policy parameters $\boldsymbol{\theta}^{(t)} \in \mathbb{R}^{d^{(t)}}$ for task $t \in \mathcal{G}^{(g)}$ can be modeled as a sparse linear combination of latent components from knowledge base $B^{(g)} \in \mathbb{R}^{d^{(t)} \times k}$ that is shared among all tasks in the group. Therefore, we have that $\boldsymbol{\theta}^{(t)} = B^{(g)} s^{(t)}$, with sparse task-specific coefficients $s^{(t)} \in \mathbb{R}^{k}$ for task t. The collection of all task-specific coefficients for tasks in $\mathcal{G}^{(g)}$ is given by $\boldsymbol{S}^{(t \in \mathcal{G}^{(g)})} \in \mathbb{R}^{k \times |\mathcal{G}^{(g)}|}$.

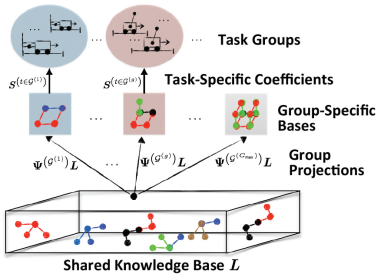


Figure 1: The knowledge framework, showing the shared repository of transferrable knowledge $\boldsymbol{L}$, group projections $\mathbf{\Psi}$ that specialize $\boldsymbol{L}$ to each task group, task-specific coefficients over the specialized basis, and task groups.

Effectively, $B^{(g)}$ forms a k-component basis over the policy parameters for all tasks in $\mathcal{G}^{(g)}$, enabling the transfer of knowledge between tasks from this group. The task-specific coefficients $\boldsymbol{s}^{(t)}$ are encouraged to be sparse to ensure that each learned basis component captures a maximal reusable chunk of knowledge. This knowledge framework has been use successfully by a number of other MTL methods [Kumar & Daumé III, 2012; Maurer *et al.*, 2013; Ruvolo & Eaton, 2013] for transfer between tasks within a single task domain.

To support cross-domain transfer, we introduce a repository of knowledge $\boldsymbol{L} \in \mathbb{R}^{d \times k}$ that is shared among all tasks (including between task domains). This matrix $\boldsymbol{L}$ represents a set of latent factors that underly the set of group-specific basis matrices $\{B^{(1)}, \ldots, B^{(G_{\max})}\}$. We introduce a set of group projection matrices $\mathbf{\Psi}^{(\mathcal{G}^{(1)})}, \ldots, \mathbf{\Psi}^{(\mathcal{G}^{(G_{\max})})}$ that map the shared latent factors $\boldsymbol{L}$ into the basis for each group of tasks. Therefore, we have that $B^{(g)} = \mathbf{\Psi}^{(\mathcal{G}^{(g)})}\boldsymbol{L}$, where the group projection matrix $\mathbf{\Psi}^{(\mathcal{G}^{(g)})} \in \mathbb{R}^{d^{(t)} \times d}$. With this construction, we see that each mapping $\mathbf{\Psi}^{(\mathcal{G}^{(g)})}$ creates an intermediate knowledge space (i.e., $B^{(g)}$) that tailors the shared repository $\boldsymbol{L}$ into a basis that is suitable for learning tasks from group $\mathcal{G}^{(g)}$. These group-specific bases are coupled together via the $\mathbf{\Psi}$ mappings and the shared knowledge base $\boldsymbol{L}$, facilitating transfer across task domains with different feature spaces. The group mappings $\mathbf{\Psi}$'s also serve to help avoid overfitting and ensure compactness of the basis, while maximizing transfer both between tasks within a group and across task groups. This construction is depicted in Figure 1.

5.1 The Cross-Domain MTL Objective

Under this shared knowledge framework, given a task $t \in \mathcal{G}^{(g)}$, its policy parameters $\boldsymbol{\theta}^{(t)} = \mathbf{\Psi}^{(\mathcal{G}^{(g)})}\boldsymbol{L}\boldsymbol{s}^{(t)}$, where $\mathbf{\Psi}^{(\mathcal{G}^{(g)})} \in \mathbb{R}^{d_{(t)} \times d}$, $\boldsymbol{L} \in \mathbb{R}^{d \times k}$, $\boldsymbol{s}^{(t)} \in \mathbb{R}^{k}$, and k is the number of shared latent knowledge components. Therefore, to train optimal policies for all tasks, we must learn the shared knowledge base $\boldsymbol{L}$, the group projections (the $\mathbf{\Psi}$'s), and the task-specific coefficients (the $\boldsymbol{s}^{(t)}$'s). We first examine this problem from a batch MTL standpoint, and then develop an efficient online algorithm for optimizing this MTL objective and enabling lifelong learning in the next section.

Given task groups $\mathcal{G}^{(1)}, \ldots, \mathcal{G}^{(G)}$, we can represent our ob-

jective of learning $T = \sum_{g=1}^{G} |\mathcal{G}^{(g)}|$ stationary policies while maximizing the amount of transfer between task policies as the problem of minimizing:

$$e_{\mathsf{T}}\left(\boldsymbol{L}, \boldsymbol{\Psi}^{(\mathcal{G}^{(1)})}, \ldots, \boldsymbol{\Psi}^{(\mathcal{G}^{(G)})}\right) \tag{2}$$
$$= \sum_{g=1}^{G} \left\{ \frac{1}{|\mathcal{G}^{(g)}|} \sum_{t \in \mathcal{G}^{(g)}} \min_{\boldsymbol{s}^{(t)}} \left[-\mathcal{J}\left(\boldsymbol{\theta}^{(t)}\right) + \mu_1 \left|\left| \boldsymbol{s}^{(t)} \right|\right|_1 \right] + \mu_2 \left|\left| \boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \right|\right|_{\mathsf{F}}^2 \right\} + \mu_3 \left|\left| \boldsymbol{L} \right|\right|_{\mathsf{F}}^2 \ ,$$

where $\boldsymbol{\theta}^{(t)} = \boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \boldsymbol{L} \boldsymbol{s}^{(t)}$ and the L_1 norm of $\boldsymbol{s}^{(t)}$ is used to approximate the true vector sparsity. We employ regularization via the Frobenius norm $||\cdot||_{\mathsf{F}}$ to avoid overfitting on both the shared knowledge base $\boldsymbol{L}$ and each of the group projections $\boldsymbol{\Psi}^{(\mathcal{G}^{(1)})}, \ldots, \boldsymbol{\Psi}^{(\mathcal{G}^{(G)})}$. Note that this objective is closely related to PG-ELLA [Bou Ammar *et al.*, 2014], with the critical difference that it incorporates cross-domain transfer.

5.2 Online Solution to Cross-Domain MTL

Although Equation 2 allows for batch cross-domain MTL, the dependence on *all* available trajectories from all tasks (via $\mathcal{J}\left(\boldsymbol{\theta}^{(t)}\right) = \int_{\boldsymbol{\tau} \in \mathbb{T}^{(t)}} p_{\boldsymbol{\theta}^{(t)}}(\boldsymbol{\tau})\, \mathfrak{R}^{(t)}(\boldsymbol{\tau})\, d\boldsymbol{\tau}$) make the batch approach unsuitable for learning tasks consecutively, since the learner requires all trajectories for acquiring a successful behavior. Here, we derive an approximate optimization algorithm that is more suitable for lifelong learning.

Standardizing the Objective

To derive the approximate optimization problem, we note that policy gradient methods maximize the lower bound of $\mathcal{J}(\boldsymbol{\theta})$. In order to use Equation 2 for lifelong cross-domain transfer with policy gradients, we must first incorporate this lower bound into our objective function. Rewriting the error term in Equation 2 in terms of the lower bound yields

$$e_{\mathsf{T}}\left(\boldsymbol{L}, \boldsymbol{\Psi}^{(\mathcal{G}^{(1)})}, \ldots, \boldsymbol{\Psi}^{(\mathcal{G}^{(G)})}\right) \tag{3}$$
$$= \sum_{g=1}^{G} \left\{ \frac{1}{|\mathcal{G}^{(g)}|} \sum_{t \in \mathcal{G}^{(g)}} \min_{\boldsymbol{s}^{(t)}} \left[-\mathcal{J}_{\mathcal{L},\boldsymbol{\theta}}\left(\tilde{\boldsymbol{\theta}}^{(t)}\right) + \mu_1 \left|\left| \boldsymbol{s}^{(t)} \right|\right|_1 \right] + \mu_2 \left|\left| \boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \right|\right|_{\mathsf{F}}^2 \right\} + \mu_3 \left|\left| \boldsymbol{L} \right|\right|_{\mathsf{F}}^2 \ ,$$

with $\tilde{\boldsymbol{\theta}}^{(t)} = \boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \boldsymbol{L} \boldsymbol{s}^{(t)}$. However, we can note that

$$\mathcal{J}_{\mathcal{L},\boldsymbol{\theta}}\left(\tilde{\boldsymbol{\theta}}^{(t)}\right) \propto -\int_{\boldsymbol{\tau} \in \mathbb{T}^{(t)}} p_{\boldsymbol{\theta}^{(t)}}(\boldsymbol{\tau})\, \mathfrak{R}^{(t)}(\boldsymbol{\tau}) \log \left[\frac{p_{\boldsymbol{\theta}^{(t)}}(\boldsymbol{\tau})\, \mathfrak{R}^{(t)}(\boldsymbol{\tau})}{p_{\tilde{\boldsymbol{\theta}}^{(t)}}(\boldsymbol{\tau})} \right] d\boldsymbol{\tau} \ .$$

Therefore, maximizing the lower bound of $\mathcal{J}_{\mathcal{L},\boldsymbol{\theta}}\left(\tilde{\boldsymbol{\theta}}^{(t)}\right)$ is equivalent to the following *minimization* problem:

$$\min_{\tilde{\boldsymbol{\theta}}^{(t)}} \int_{\boldsymbol{\tau} \in \mathbb{T}^{(t)}} p_{\boldsymbol{\theta}^{(t)}}(\boldsymbol{\tau})\, \mathfrak{R}^{(t)}(\boldsymbol{\tau}) \log \left[\frac{p_{\boldsymbol{\theta}^{(t)}}(\boldsymbol{\tau})\, \mathfrak{R}^{(t)}(\boldsymbol{\tau})}{p_{\tilde{\boldsymbol{\theta}}^{(t)}}(\boldsymbol{\tau})} \right] d\boldsymbol{\tau} \ , \tag{4}$$

which can be plugged into Equation 3 in place of $\mathcal{J}_{\mathcal{L},\boldsymbol{\theta}}\left(\tilde{\boldsymbol{\theta}}^{(t)}\right)$ to obtain the MTL policy gradients objective.

Approximate Learning Objective

To eliminate the dependence of the objective function on all available trajectories, we can approximate $e_{\mathsf{T}}(\cdot)$ by performing a second-order Taylor expansion of $\mathcal{J}_{\mathcal{L},\boldsymbol{\theta}}\left(\tilde{\boldsymbol{\theta}}^{(t)}\right)$ around the optimal solution $\boldsymbol{\alpha}^{(t)}$ to Equation 4, following the technique used in PG-ELLA [Bou Ammar *et al.*, 2014]. Note that attaining $\boldsymbol{\alpha}^{(t)}$ corresponds to solving the policy gradient problem of task $t \in \mathcal{G}^{(g)}$, which might be computationally expensive. Therefore, rather than using the above, we use an approximation acquired by performing a gradient step in task t: $\boldsymbol{\alpha}^{(t)} = \boldsymbol{\theta} + \eta \boldsymbol{\mathcal{I}}^{-1} \nabla_{\tilde{\boldsymbol{\theta}}^{(t)}} \mathcal{J}_{\mathcal{L},\boldsymbol{\theta}}\left(\tilde{\boldsymbol{\theta}}^{(t)}\right)$, where $\boldsymbol{\mathcal{I}}$ is the Fisher information matrix. The first derivative, needed for the second-order Taylor expansion, is given by:

$$\nabla_{\tilde{\boldsymbol{\theta}}^{(t)}} \mathcal{J}_{\mathcal{L},\boldsymbol{\theta}}\left(\tilde{\boldsymbol{\theta}}^{(t)}\right) = -\int_{\boldsymbol{\tau} \in \mathbb{T}^{(t)}} p_{\boldsymbol{\theta}^{(t)}}(\boldsymbol{\tau})\, \mathfrak{R}^{(t)}(\boldsymbol{\tau})\, \nabla_{\tilde{\boldsymbol{\theta}}^{(t)}} \log p_{\tilde{\boldsymbol{\theta}}^{(t)}}(\boldsymbol{\tau})\, d\boldsymbol{\tau}$$

$$\log p_{\tilde{\boldsymbol{\theta}}^{(t)}}(\boldsymbol{\tau}) = \log p^{(t)}\left(\boldsymbol{x}_1^{(t)}\right) + \sum_{m=1}^{M^{(t)}} p^{(t)}\left(\boldsymbol{x}_{m+1}^{(t)} \mid \boldsymbol{x}_m^{(t)}, \boldsymbol{a}_m^{(t)}\right) + \sum_{m=1}^{M^{(t)}} \log \pi_{\tilde{\boldsymbol{\theta}}^{(t)}}\left(\boldsymbol{a}_m^{(t)} \mid \boldsymbol{x}_m^{(t)}\right) \ .$$

Therefore, we have that

$$\nabla_{\tilde{\boldsymbol{\theta}}^{(t)}} \mathcal{J}_{\mathcal{L},\boldsymbol{\theta}}\left(\tilde{\boldsymbol{\theta}}^{(t)}\right)$$
$$= -\int_{\boldsymbol{\tau} \in \mathbb{T}^{(t)}} p_{\boldsymbol{\theta}^{(t)}}(\boldsymbol{\tau})\, \mathfrak{R}^{(t)} \left[\sum_{m=1}^{M^{(t)}} \nabla_{\tilde{\boldsymbol{\theta}}^{(t)}} \log \pi_{\tilde{\boldsymbol{\theta}}^{(t)}}\left(\boldsymbol{a}_m^{(t)} | \boldsymbol{x}_m^{(t)}\right) \right] d\boldsymbol{\tau}$$
$$= -\mathbb{E}\left[\left(\sum_{m=1}^{M^{(t)}} \nabla_{\tilde{\boldsymbol{\theta}}^{(t)}} \log \pi_{\tilde{\boldsymbol{\theta}}^{(t)}}\left(\boldsymbol{a}_m^{(t)} | \boldsymbol{x}_m^{(t)}\right) \right) \mathfrak{R}^{(t)}(\boldsymbol{\tau}) \right] \ .$$

The second derivative of $\mathcal{J}_{\mathcal{L},\boldsymbol{\theta}}\left(\tilde{\boldsymbol{\theta}}^{(t)}\right)$ is then:

$$\boldsymbol{\Gamma}^{(t)} = -\mathbb{E}\left[\mathfrak{R}^{(t)}(\boldsymbol{\tau}) \sum_{m=1}^{M^{(t)}} \nabla^2_{\tilde{\boldsymbol{\theta}}^{(t)}, \tilde{\boldsymbol{\theta}}^{(t)}} \log \pi_{\tilde{\boldsymbol{\theta}}^{(t)}}\left(\boldsymbol{a}_m^{(t)} | \boldsymbol{x}_m^{(t)}\right) \right]_{\tilde{\boldsymbol{\theta}}^{(t)} = \boldsymbol{\alpha}^{(t)}} \ .$$

Substituting the second-order Taylor expansion yields the following approximation of Equation 2:

$$\hat{e}_{\mathsf{T}}\left(\boldsymbol{L}, \boldsymbol{\Psi}^{(\mathcal{G}^{(1)})}, \ldots, \boldsymbol{\Psi}^{(\mathcal{G}^{(G)})}\right) \tag{5}$$
$$= \sum_{g=1}^{G} \frac{1}{|\mathcal{G}^{(g)}|} \sum_{t \in \mathcal{G}^{(g)}} \min_{\boldsymbol{s}^{(t)}} \left[\left|\left| \boldsymbol{\alpha}^{(t)} - \boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \boldsymbol{L} \boldsymbol{s}^{(t)} \right|\right|^2_{\boldsymbol{\Gamma}^{(t)}} + \mu_1 \left|\left| \boldsymbol{s}^{(t)} \right|\right|_1 + \mu_2 \left|\left| \boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \right|\right|_{\mathsf{F}}^2 \right] + \mu_3 \left|\left| \boldsymbol{L} \right|\right|_{\mathsf{F}}^2 \ ,$$

where $||\boldsymbol{v}||^2_{\boldsymbol{A}} = \boldsymbol{v}^{\mathsf{T}} \boldsymbol{A} \boldsymbol{v}$, the constant term was suppressed as it has no effect on the minimization, and the linear term was ignored by construction since $\boldsymbol{\alpha}^{(t)}$ is a minimizer. Critically, we have eliminated the dependence on all available trajectories, making the problem suitable for an online MTL setting.

Learning the Policy

We fit the policy parameters in two steps. Upon observing a task $t \in \mathcal{G}^{(g)}$, we use gradient descent to update the shared repository $\boldsymbol{L}$ and the group projections $\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})}$. The update rules, acquired by taking the derivative of Equation 5 with respect to $\boldsymbol{L}$ and $\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})}$, can be written as:

$$\begin{aligned}\Delta \boldsymbol{L} = \eta_{\boldsymbol{L}} \Bigg[\sum_{g} \frac{1}{|\mathcal{G}^{(g)}|} \sum_{t \in \mathcal{G}^{(g)}} -\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})^{\mathsf{T}}} \boldsymbol{\Gamma}^{(t)} \boldsymbol{\alpha}^{(t)} \boldsymbol{s}^{(t)^{\mathsf{T}}} \\ + \boldsymbol{\Psi}^{(\mathcal{G}^{(g)})^{\mathsf{T}}} \boldsymbol{\Gamma}^{(t)} \boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \boldsymbol{L} \boldsymbol{s}^{(t)} \boldsymbol{s}^{(t)^{\mathsf{T}}} + \mu_3 \boldsymbol{L} \Bigg]\end{aligned} \tag{6}$$

$$\begin{aligned}\Delta \boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} = \eta_{\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})}} \Bigg[\frac{1}{|\mathcal{G}^{(g)}|} \sum_{t \in \mathcal{G}^{(g)}} -\boldsymbol{\Gamma}^{(t)} \boldsymbol{\alpha}^{(t)} \left(\boldsymbol{L} \boldsymbol{s}^{(t)} \right)^{\mathsf{T}} \\ + \boldsymbol{\Gamma}^{(t)} \boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \left(\boldsymbol{L} \boldsymbol{s}^{(t)} \right) \left(\boldsymbol{L} \boldsymbol{s}^{(t)} \right)^{\mathsf{T}} + \mu_2 \boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \Bigg] ,\end{aligned} \tag{7}$$

where $\eta_{\boldsymbol{L}}$ and $\eta_{\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})}}$ are the learning rates for the shared repository $\boldsymbol{L}$ and group projections $\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})}$, respectively.

Having learned the shared representation $\boldsymbol{L}$ and group projections, task-specific coefficients $\boldsymbol{s}^{(t)}$ are then determined by solving an instance of Lasso [Tibshirani, 1996] to encode $\boldsymbol{\alpha}^{(t)}$ in the basis given by $\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \boldsymbol{L}$, leading to a task-specific policy $\pi^{(t)} = p_{\boldsymbol{\theta}^{(t)}}(\boldsymbol{a} \mid \boldsymbol{x})$ where $\boldsymbol{\theta}^{(t)} = \boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \boldsymbol{L} \boldsymbol{s}^{(t)}$. For computationally demanding domains, we can update $\boldsymbol{s}^{(t)}$ less frequently, instead allowing the policy to change by altering $\boldsymbol{L}$ and the $\boldsymbol{\Psi}$'s. The complete implementation of our approach is available on the authors' websites.

6 Theoretical Guarantees

This section shows that our approach becomes stable as the number of tasks and groups grow large. Detailed proofs and definitions are provided in an online appendix available on the authors' websites. First, we consider the one-group setting by defining the following expected loss:[1]

$$\begin{aligned}&\hat{\boldsymbol{h}}_{|\mathcal{G}^{(g)}|} \left(\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} | \mathcal{G}^{(g)} \right) \\ &= \mathbb{E}_{(\boldsymbol{\alpha}^{(t)}, \boldsymbol{\Gamma}^{(t)})} \left[\min_{\boldsymbol{s}} l \left(\boldsymbol{L}, \boldsymbol{\Psi}^{\mathcal{G}^{(g)}}, \boldsymbol{s}, \boldsymbol{\alpha}^{(t)}, \boldsymbol{\Gamma}^{(t)} \middle| \mathcal{G}^{(g)} \right) \right] ,\end{aligned} \tag{8}$$

where the expectation is over each task t in group $\mathcal{G}^{(g)}$ according to the task's parameters $\left(\boldsymbol{\alpha}^{(t)}, \boldsymbol{\Gamma}^{(t)}\right)$, and $l(\cdot)$ is the per-task loss of encoding $\boldsymbol{\alpha}^{(t)}$ in the basis given by $\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \boldsymbol{L}$.

Proposition 1.

$$\boldsymbol{\Psi}^{[|\mathcal{G}^{(g)}|, (\mathcal{G}^{(g)})]} - \boldsymbol{\Psi}^{[|\mathcal{G}^{(g)}|-1, (\mathcal{G}^{(g)})]} = \mathcal{O}\left(\frac{1}{|\mathcal{G}^{(g)}|} \right)$$

Proof. Here, we sketch the proof of Prop. 1. With $l \left(\boldsymbol{L}, \boldsymbol{\Psi}^{\mathcal{G}^{(g)}}, \boldsymbol{s}, \boldsymbol{\alpha}^{(t)}, \boldsymbol{\Gamma}^{(t)} \middle| \mathcal{G}^{(g)} \right)$, we can show that $\hat{\boldsymbol{h}}_{|\mathcal{G}^{(g)}|} \left(\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} | \mathcal{G}^{(g)} \right) - \hat{\boldsymbol{h}}_{|\mathcal{G}^{(g)}|-1} \left(\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} | \mathcal{G}^{(g)} \right)$ is Lipschitz with $\mathcal{O}\left(\frac{1}{|\mathcal{G}^{(g)}|} \right)$. Further, given enough gradient steps, for $\boldsymbol{\Psi}^{(|\mathcal{G}^{(g)}|-1), (\mathcal{G}^{(g)})}$ to minimize $\hat{\boldsymbol{h}}_{|\mathcal{G}^{(g)}|-1} \left(\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} | \mathcal{G}^{(g)} \right)$, it is clear that change in the loss can be upper bounded by $2\lambda \left|\left| \boldsymbol{\Psi}^{(|\mathcal{G}^{(g)}|), (\mathcal{G}^{(g)})} - \boldsymbol{\Psi}^{(|\mathcal{G}^{(g)}|-1), (\mathcal{G}^{(g)})} \right|\right|_{\mathsf{F}}^2$. Combining the Lipschitz bound with previous facts concludes the proof. □

Proposition 2. *With $\boldsymbol{h}(\cdot)$ as the actual loss in $\mathcal{G}^{(g)}$, we show:*
1. $\hat{\boldsymbol{h}}_{|\mathcal{G}^{(g)}|} \left(\boldsymbol{\Psi}^{\mathcal{G}^{(g)}} | \mathcal{G}^{(g)} \right)$ converges a.s.
2. $\hat{\boldsymbol{h}}_{|\mathcal{G}^{(g)}|} \left(\boldsymbol{\Psi}^{\mathcal{G}^{(g)}} | \mathcal{G}^{(g)} \right) - \boldsymbol{h}_{|\mathcal{G}^{(g)}|} \left(\boldsymbol{\Psi}^{\mathcal{G}^{(g)}} | \mathcal{G}^{(g)} \right)$ converges a.s. to 0
3. $\hat{\boldsymbol{h}}_{|\mathcal{G}^{(g)}|} \left(\boldsymbol{\Psi}^{\mathcal{G}^{(g)}} | \mathcal{G}^{(g)} \right) - \boldsymbol{h} \left(\boldsymbol{\Psi}^{\mathcal{G}^{(g)}} | \mathcal{G}^{(g)} \right)$ converges a.s. to 0
4. $\boldsymbol{h} \left(\boldsymbol{\Psi}^{\mathcal{G}^{(g)}} | \mathcal{G}^{(g)} \right)$ converges a.s.

Proof. First, we show that the sum of positive variations of the stochastic process $\boldsymbol{u}_{|\mathcal{G}^{(g)}|} = \hat{\boldsymbol{h}}_{|\mathcal{G}^{(g)}|} \left(\boldsymbol{\Psi}^{(|\mathcal{G}^{(g)}|), (\mathcal{G}^{(g)})} | \mathcal{G}^{(g)} \right)$ are bounded by invoking a corollary of the Donsker theorem [Van der Vaart, 2000]. This result in combination with a theorem from Fisk [1965], allows us to show that $u_{t\left(\mathcal{G}^{(g)}\right)}$ is a quasi-martingale that converges almost surely (a.s.). This fact along with a simple theorem of positive sequences allows us prove part 2 of the proposition. The final two parts (3 & 4) can be shown due to the equivalence of $\boldsymbol{h}$ and $\boldsymbol{h}_{|\mathcal{G}^{(g)}|}$ as $|\mathcal{G}^{(g)}| \to \infty$. □

Proposition 3. *The distance between $\boldsymbol{\Psi}^{(|\mathcal{G}^{(g)}|), (\mathcal{G}^{(g)})}$ and the set of $\boldsymbol{h}$'s stationary points converges a.s. to 0 as $|\mathcal{G}^{(g)}| \to \infty$.*

Proof. Both the surrogate $\hat{\boldsymbol{h}}_{|\mathcal{G}^{(g)}|}$ and the expected cost $\boldsymbol{h}$ have gradients that are Lipschitz with constant independent of $|\mathcal{G}^{(g)}|$. This fact, in combination with the fact that $\hat{\boldsymbol{h}}_{|\mathcal{G}^{(g)}|}$ and $\boldsymbol{g}$ converges a.s. as $|\mathcal{G}^{(g)}| \to \infty$, completes the proof. □

Next, we consider the loss of multiple groups:

$$\hat{\boldsymbol{g}}^{(\mathcal{G}^{(g)})}(\boldsymbol{L}) = \mathbb{E}_{\mathcal{G}^{(g)}} \left[\hat{\boldsymbol{h}}^{(|\mathcal{G}^{(g)}|)} \left(\boldsymbol{\Psi}^{\mathcal{G}^{(g)}} \right) \middle| \mathcal{G}^{(g)} \right] . \tag{9}$$

Proposition 4.

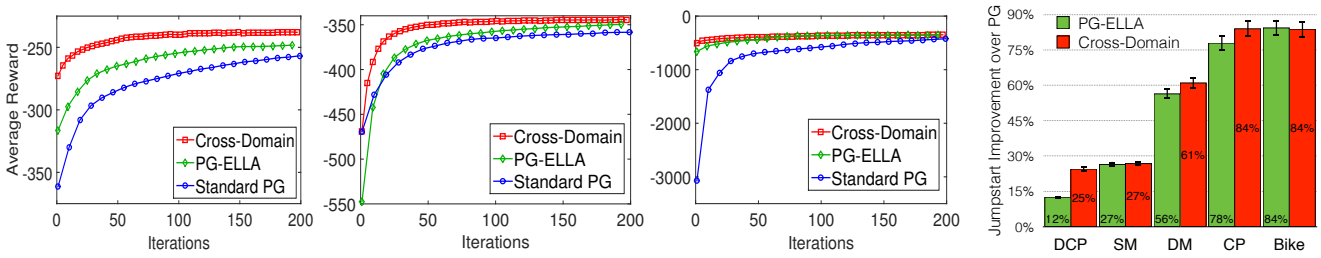
$$\boldsymbol{L}^{(\mathcal{G}^{(g)})} - \boldsymbol{L}^{(\mathcal{G}^{(g)})-1} = \mathcal{O}\left(\sum_{g=1}^{G} \frac{1}{|\mathcal{G}^{(g)}|} \right)$$

Proof. This can be easily shown as the upper bound of $\sum_g \left(\hat{\boldsymbol{h}} \left(\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \middle| \mathcal{G}^{(g)} \right) \cdot \right)$ is the sum of the bounds over $\hat{\boldsymbol{h}} \left(\boldsymbol{\Psi}^{(\mathcal{G}^{(g)})} \middle| \mathcal{G}^{(g)} \right)$. □

Proposition 5. *With $\boldsymbol{g}(\cdot)$ as the actual loss, we show:*
1. $\hat{\boldsymbol{g}}^{(\mathcal{G}^{(g)})}(\boldsymbol{L})$ converges a.s.
2. $\hat{\boldsymbol{g}}^{(\mathcal{G}^{(g)})}(\boldsymbol{L}) - \boldsymbol{g}^{(\mathcal{G}^{(g)})}(\boldsymbol{L})$ convergence a.s. to 0
3. $\hat{\boldsymbol{g}}^{(\mathcal{G}^{(g)})}(\boldsymbol{L}) - \boldsymbol{g}^{(\mathcal{G}^{(g)})}(\boldsymbol{L})$ convergence a.s. to 0
4. $\boldsymbol{g}^{(\mathcal{G}^{(g)})}(\boldsymbol{L})$ converges a.s.

Proof. This can be attained similarly to that of Prop. 2. □

[1] We super- or subscript variables with $(|\mathcal{G}^{(g)}|)$ to denote the version of the variable learned from $|\mathcal{G}^{(g)}|$ tasks in $\mathcal{G}^{(g)}$.



(a) Double Cart-Pole (b) Helicopter (c) Cart-Pole (d) Jumpstart over PG

Figure 2: Learning performance after interleaved training over multiple task domains. Figures (a) and (b) depict task domains where cross-domain transfer has a significant impact, showing that our approach outperforms standard PG and PG-ELLA. Figure (c) demonstrates that even when a domain benefits less from cross-domain transfer, our approach still achieves equivalent performance to PG-ELLA. Figure (d) depicts the average improvement in initial task performance over PG from transfer.

7 Empirical Results

We evaluated the ability of our approach to learn optimal control policies for multiple consecutive tasks from six different dynamical systems (each corresponding to one task domain):

Simple Mass (SM): The spring-mass-damper system is characterized by three parameters: spring and damping constants and the mass. The system's state is given by the position and velocity of the mass, which varies according to linear force. The goal is to control the mass to be in a specific state.

Double Mass (DM): The double spring-mass-damper has six parameters: two spring constants, damping constants, and masses. The state is given by the position and velocity of both masses. The goal is to control the first mass to a specific state, while only applying a linear force to the second.

Cart-Pole (CP): The dynamics of the inverted pendulum system are characterized by the cart's and pole's masses, the pole's length, and a damping parameter. The state is characterized by the cart's position and velocity, and the pole's angle and angular velocity. The goal is to balance the pole upright.

Double Cart-Pole (DCP): The DCP adds a second inverted pendulum to the CP system, with six parameters and six state features. The goal is to balance both poles upright.

Bicycle (Bike): The Bike model assumes a fixed rider, and is characterized by eight parameters. The goal is to keep the bike balanced as it rolls along the horizontal plane.

Helicopter (HC): This linearized model of a CH-47 tandem-rotor helicopter assumes horizontal motion at 40 knots. The main goal is to stabilize the helicopter by controlling the collective and differential rotor thrust.

For each of these systems, we created three different tasks by varying the system parameters to create systems with different dynamics, yielding 18 tasks total. These tasks used a reward function typical for optimal control, given by $-\sqrt{(x_m - \dot{x})^\intercal (x_m - \dot{x})} - \sqrt{a_m^\intercal a_m}$ where $\dot{x}$ is the goal state. Each round of the lifelong learning experiment, one task t was chosen randomly with replacement, and task t's model was trained from 100 sampled trajectories of length 50. This process continued until all tasks were seen at least once.

We then compared the performance of cross-domain lifelong learning with PG-ELLA and standard policy gradients (PG), averaging results over ∼94 trials per domain (each of which contained ∼60 interleaved training rounds). As the base PG learner in all algorithms, we used Natural Actor Critic [Peters & Schaal, 2008]. All regularization parameters (the μ's) were set to e^{-5}, and the learning rates and latent dimensions were set via cross-validation over a few tasks.

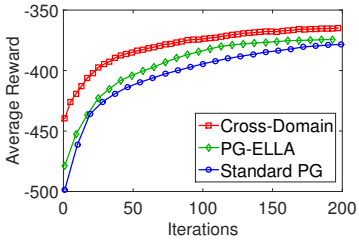


Figure 3: Average learning performance on a novel task domain (helicopter) after lifelong learning on other domains.

Figure 2 shows the average learning performance on individual domains after this process of interleaved lifelong learning, depicting domains in which cross-domain transfer shows clear advantages over PG-ELLA and PG (e.g., DCP, HC), and an example domain where cross-domain transfer is less effective (CP). Note that even in a domain where cross-domain transfer provides little benefit, our approach achieves equivalent performance to PG-ELLA, showing that cross-domain transfer does not interfere with learning effectiveness. On all task domains except HC, our approach provides a significant increase in initial performance due to transfer (Figure 2(d)).

Cross-domain transfer provides significant advantages when the lifelong learning agent faces a novel task domain. To evaluate this, we chose the most complex of the task domains (helicopter) and trained the lifelong learner on tasks from all other task domains to yield an effective shared knowledge base $\boldsymbol{L}$. Then, we evaluated the agent's ability to learn a new task from the helicopter domain, comparing the benefits of cross-domain transfer from $\boldsymbol{L}$ with PG-ELLA and PG (both of which learn from scratch on the new domain). Figure 3 depicts the result of learning on a novel domain, averaged over ten trials for all three HC tasks, showing the effectiveness of cross-domain lifelong learning in this scenario.

8 Conclusion

We have presented the first lifelong RL method that supports autonomous and efficient cross-domain transfer. This approach provides a variety of theoretical guarantees, and can learn effectively across multiple task domains, providing improved performance over single-domain methods.

Acknowledgements

This research was supported by ONR grant #N00014-11-1-0139 and AFRL grant #FA8750-14-1-0069. We thank the anonymous reviewers for their helpful feedback.

References

[Bou Ammar *et al.*, 2014] Haitham Bou Ammar, Eric Eaton, Paul Ruvolo, & Matthew Taylor. Online multi-task learning for policy gradient methods. In *International Conference on Machine Learning (ICML)*, 2014.

[Bou Ammar *et al.*, 2012] Haitham Bou Ammar, Mathew E. Taylor, Karl Tuyls, Kurt Driessens, & Gerhard Weiss. Reinforcement learning transfer via sparse coding. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2012.

[Bou Ammar *et al.*, 2013] Haitham Bou Ammar, Decebal Constantin Mocanu, Matthew Taylor, Kurt Driessens, Gerhard Weiss, & Karl Tuyls. Automatically mapped transfer between reinforcement learning tasks via three-way restricted Boltzmann machines. In *European Conference on Machine Learning (ECML)*, 2013.

[Bertsekas, 1995] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*, vol. 1. Athena Scientific, Belmont, MA, 1995.

[Busoniu *et al.*, 2008] Lucian Busoniu, Robert Babuska, & Bart De Schutter. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 38(2):156–172, 2008.

[Deisenroth *et al.*, 2014] Marc Peter Deisenroth, Peter Englert, Jan Peters, & Dieter Fox. Multi-task policy search for robotics. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3876–3881, 2014.

[Dempster & Leemans, 2006] Michael A.H. Dempster & Vasco Leemans. An automated FX trading system using adaptive reinforcement learning. *Expert Systems with Applications*, 30(3):543–552, 2006.

[Fisk, 1965] D.L. Fisk. Quasi-martingales. *Transactions of the American Mathematical Society*, 120(3):369–389, 1965.

[Han *et al.*, 2012] Shaobo Han, Xuejun Liao, & Lawrence Carin Cross-domain multi-task learning with latent probit models. In *International Conference on Machine Learning (ICML)*, 2012.

[Kang *et al.*, 2011] Zhuoliang Kang, Kristen Grauman, & Fei Sha. Learning with whom to share in multi-task feature learning. In *International Conference on Machine Learning (ICML)*, pages 521–528, 2011.

[Kober & Peters, 2011] Jens Kober & Jan Peters. Policy search for motor primitives in robotics. *Machine Learning*, 84(1-2), 2011.

[Konidaris & Doshi-Velez, 2014] George Konidaris & Finale Doshi-Velez. Hidden parameter Markov decision processes: an emerging paradigm for modeling families of related tasks. In the *AAAI Fall Symposium on Knowledge, Skill, and Behavior Transfer in Autonomous Robots*, 2014.

[Kumar & Daumé III, 2012] Abhishek Kumar & Hal Daumé III. Learning task grouping and overlap in multi-task learning. In *International Conference on Machine Learning (ICML)*, 2012.

[Lazaric & Ghavamzadeh, 2010] Alessandro Lazaric & Mohammad Ghavamzadeh. Bayesian multi-task reinforcement learning. In *International Conference on Machine Learning*, pages 599–606, 2010.

[Li *et al.*, 2009] Hui Li, Xuejun Liao, & Lawrence Carin. Multi-task reinforcement learning in partially observable stochastic environments. *Journal of Machine Learning Research*, 10:1131–1186, 2009.

[Maurer *et al.*, 2013] Andreas Maurer, Massimiliano Pontil, & Bernardino Romera-Paredes. Sparse coding for multitask and transfer learning. In *International Conference on Machine Learning*, pages 343–351, 2013.

[Peters & Schaal, 2008] Jan Peters & Stefan Schaal. Natural actor-critic. *Neurocomputing*, 71(7):1180–1190, 2008.

[Peters *et al.*, 2005] Jan Peters, Sethu Vijayakumar, & Stefan Schaal. Natural actor-critic. In *European Conference on Machine Learning (ECML)*, pages 280–291, 2005.

[Rückstieß *et al.*, 2008] Thomas Rückstieß, Martin Felder, & Jürgen Schmidhuber. State-dependent exploration for policy gradient methods. In *Machine Learning and Knowledge Discovery in Databases*, pages 234–249, 2008.

[Ruvolo & Eaton, 2013] Paul Ruvolo & Eric Eaton. ELLA: an efficient lifelong learning algorithm. In *International Conference on Machine Learning (ICML)*, 2013.

[Smart & Kaelbling, 2002] William D. Smart & Leslie Pack Kaelbling. Effective reinforcement learning for mobile robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 4, pages 3404–3410, 2002.

[Snel & Shimon Whiteson, 2014] Matthijs Snel & Shimon Whiteson. Learning potential functions and their representations for multi-task reinforcement learning. *Autonomous Agents and Multi-Agent Systems*, 28(4):637–681, 2014.

[Sutton *et al.*, 1999] Richard S. Sutton, David A. McAllester, Satinder P. Singh, Yishay Mansour, et al. Policy gradient methods for reinforcement learning with function approximation. *Neural Information Processing Systems* 99:1057–1063, 1999.

[Taylor & Stone, 2009] Matthew E. Taylor & Peter Stone. Transfer learning for reinforcement learning domains: a survey. *Journal of Machine Learning Research*, 10:1633–1685, 2009.

[Taylor *et al.*, 2007] Matthew E. Taylor, Shimon Whiteson, & Peter Stone. Transfer via inter-task mappings in policy search reinforcement learning. In *International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 156–163, 2007.

[Taylor *et al.*, 2008] Matthew E. Taylor, Gregory Kuhlmann, & Peter Stone. Autonomous transfer for reinforcement learning. In *International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 283–290, 2008.

[Thrun & O'Sullivan, 1996] Sebastian Thrun & Joseph O'Sullivan. Discovering structure in multiple learning tasks: the TC algorithm. In *International Conference on Machine Learning*, 1996.

[Tibshirani, 1996] Robert Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[Van der Vaart, 2000] A.W. Van der Vaart. *Asymptotic statistics*, volume 3. Cambridge University Press, 2000.

[Williams, 1992] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992.

[Wilson *et al.*, 2007] Aaron Wilson, Alan Fern, Soumya Ray, & Prasad Tadepalli. Multi-task reinforcement learning: a hierarchical Bayesian approach. In *International Conference on Machine Learning (ICML)*, 2007.