Using Task Features for Zero-Shot Knowledge Transfer in Lifelong Learning

David Isele† and **Mohammad Rostami**† and **Eric Eaton**
University of Pennsylvania, Philadelphia, PA, USA
{isele, mrostami, eeaton}@seas.upenn.edu

Abstract

Knowledge transfer between tasks can improve the performance of learned models, but requires an accurate estimate of the inter-task relationships to identify the relevant knowledge to transfer. These inter-task relationships are typically estimated based on training data for each task, which is inefficient in lifelong learning settings where the goal is to learn each consecutive task rapidly from as little data as possible. To reduce this burden, we develop a lifelong reinforcement learning method based on coupled dictionary learning that incorporates high-level task descriptors to model the inter-task relationships. We show that using task descriptors improves the performance of the learned task policies, providing both theoretical justification for the benefit and empirical demonstration of the improvement across a variety of dynamical control problems. Given only the descriptor for a new task, the lifelong learner is also able to accurately predict the task policy through zero-shot learning using the coupled dictionary, eliminating the need to pause to gather training data before addressing the task.

1 Introduction

Transfer and multi-task learning (MTL) methods reduce the amount of experience needed to train individual task models by reusing knowledge from other related tasks. These techniques typically select the relevant knowledge to transfer by modeling inter-task relationships, based on training data for each task [Baxter, 2000; Ando & Zhang, 2005; Bickel *et al.*, 2009; Maurer *et al.*, 2013]. However, this process requires sufficient training data for each task to identify these relationships before knowledge transfer can succeed.

Consider instead the human ability to rapidly bootstrap a model for a new task, given *only* a *high-level task description*, before obtaining experience on the actual task. For example, viewing only the image on the box of a new Ikea chair, we can immediately identify previous related assembly tasks and begin formulating a plan to assemble the chair. In the same manner, an experienced inverted pole balancing agent may be able to predict the controller for a new pole given its mass and length, prior to interacting with the physical system.

Inspired by this idea, we explore the use of high-level task descriptions to improve knowledge transfer between multiple machine learning tasks. We focus on lifelong learning scenarios [Thrun, 1996; Ruvolo & Eaton, 2013], in which multiple tasks arrive consecutively and the goal is to rapidly learn each new task by building upon previous knowledge. Although we focus on reinforcement learning (RL) tasks in this paper, our approach extends easily to regression and classification.

Our algorithm, Task Descriptors for Lifelong Learning (TaDeLL), encodes the task descriptions as feature vectors that identify each task, treating these descriptors as side information in addition to training data on the individual tasks. This idea of using task features for knowledge transfer has been explored previously by Bonilla et al. [2007] in an offline batch MTL setting, and more recently by Sinapov et al. [2015] in a computationally expensive method for estimating transfer relationships between pairs of tasks. In comparison, our approach operates online over consecutive tasks and is much more computationally efficient.

We use *coupled dictionary learning* to model the inter-task relationships between both the task descriptions and the individual task policies in lifelong learning. The coupled dictionary learning enforces the notion that tasks with similar descriptions should have similar policies, but still allows dictionary elements the freedom to accurately represent the different task policies. We connect the coupled dictionaries to the concept of mutual coherence in sparse coding, providing theoretical justification for why the task descriptors improve performance, and verify this improvement empirically.

In addition to improving the task policies, we show that the task descriptors enable the learner to accurately predict the policies for unseen tasks given only their description—this process of learning without data is known as *zero-shot learning*. This capability is particularly important in the online setting of lifelong learning, enabling the system to accurately predict policies for new tasks through transfer, without requiring it to pause to gather training data on each task.

2 Related Work

Batch MTL [Caruana, 1997] methods often model the relationships between tasks to determine the knowledge to transfer [Baxter, 2000; Ando & Zhang, 2005; Bickel *et al.*, 2009;

†These authors contributed equally to this work.

Lazaric & Ghavamzadeh, 2010; Maurer *et al.*, 2013]. These techniques include modeling a task distance metric [Ben-David *et al.*, 2007], using correlations to determine when transfer is appropriate [Wang *et al.*, 2014], or building models based on nearest neighbors [Parameswaran & Weinberger, 2010]. More recently, MTL has been extended to a lifelong learning setting, in which tasks arrive consecutively for regression and classification [Ruvolo & Eaton, 2013] and for reinforcement learning [Bou Ammar *et al.*, 2014; 2015]. However, all of these methods require training data for each task in order to assess their relationships and determine the knowledge to transfer.

Instead of relying solely on the tasks' training data, several works have explored the use of high level task descriptors to model the inter-task relationships in MTL and transfer learning settings. Task descriptors have been used in combination with neural networks [Bakker & Heskes, 2003] to define a task-specific prior or to control the gating network between individual task clusters. Bonilla et al. [2007] explore similar techniques for multi-task kernel machines, using task features in combination with the data for a gating network over individual task experts or to augment the original task training data. These papers focus on multi-task classification and regression in batch settings where the system has access to the data and features for all tasks, in contrast to our study of task descriptors for lifelong learning over consecutive RL tasks.

In the work most similar to ours, Sinapov et al. [2015] use task descriptors to estimate the transferability between each pair of tasks for transfer learning. Given the descriptor for a new task, they identify the source task with the highest predicted transferability, and use that source task for a warm start in RL. Though effective, their approach is computationally expensive, since they estimate the transferability for every task pair through repeated simulation. Their evaluation is also limited to a transfer learning setting, and they do not consider the effects of transfer over consecutive tasks or updates to the transferability model, as we do in the lifelong setting.

Our work is also related to the Simple Zero-Shot Learning (Simple ZSL) method by Romera-Paredes and Tor [2015], which learns a multi-class linear model, and factorizes the linear model parameters, assuming the descriptors are coefficients over a latent basis to reconstruct the models. Our approach assumes a more flexible relationship: that both the model parameters and task descriptors can be reconstructed from separate latent bases that are coupled together through their coefficients. In comparison to our lifelong learning approach, Simple ZSL operates in an offline multi-class setting.

3 Background

3.1 Reinforcement Learning

A reinforcement learning (RL) agent must select sequential actions in an environment to maximize its expected return. An RL task is typically formulated as a Markov Decision Process (MDP) $\langle \mathcal{X}, \mathcal{A}, P, R, \gamma \rangle$, where $\mathcal{X}$ is the set of states, and $\mathcal{A}$ is the set of actions that the agent may execute, $P : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \to [0, 1]$ is the state transition probability describing the systems dynamics, $R : \mathcal{X} \times \mathcal{A} \times \mathcal{X} \to \mathbb{R}$ is the reward function, and $\gamma \in [0, 1)$ is the discount assigned to rewards over time. At time step h, the agent is in state $\boldsymbol{x}_h \in \mathcal{X}$ and chooses an action $\boldsymbol{a} \in \mathcal{A}$ according to policy $\pi : \mathcal{X} \times \mathcal{A} \mapsto [0, 1]$, which is represented as a function defined by a vector of control parameters $\boldsymbol{\theta} \in \mathbb{R}^d$. The agents then receives reward r_h according to R and transitions to state $\boldsymbol{x}_{h+1}$ according to P. This sequence of states, actions, and rewards is given as a trajectory $\boldsymbol{\tau} = \{(\boldsymbol{x}_1, \boldsymbol{a}_1, r_1), \ldots, (\boldsymbol{x}_H, \boldsymbol{a}_H, r_H)\}$ over a horizon H. The goal of RL is to find the optimal policy π^* with parameters $\boldsymbol{\theta}^*$ that maximizes the expected reward. However, learning an individual task still requires numerous trajectories, motivating the use of transfer to reduce the number of interactions with the environment.

Policy Gradient (PG) methods [Sutton *et al.*, 1999], which we employ as our base learner, are a class of RL algorithms that are effective for solving high dimensional problems with continuous state and action spaces, such as robotic control [Peters & Schaal, 2008]. The goal of PG is to optimize the expected average return: $\mathcal{J}(\boldsymbol{\theta}) = E\left[\frac{1}{H}\sum_{h=1}^{H} r_h\right] = \int_{\mathbb{T}} p_{\boldsymbol{\theta}}(\boldsymbol{\tau})\mathfrak{R}(\boldsymbol{\tau})d\boldsymbol{\tau}$, where $\mathbb{T}$ is the set of all possible trajectories, the average reward on trajectory $\boldsymbol{\tau}$ is given by $\mathfrak{R}(\boldsymbol{\tau}) = \frac{1}{H}\sum_{h=1}^{H} r_h$, and $p_{\boldsymbol{\theta}}(\boldsymbol{\tau}) = P_0(\boldsymbol{x}_1)\prod_{h=1}^{H} p(\boldsymbol{x}_{h+1} \mid \boldsymbol{x}_h, \boldsymbol{a}_h)\, \pi(\boldsymbol{a}_h \mid \boldsymbol{x}_h)$ is the probability of $\boldsymbol{\tau}$ under an initial state distribution $P_0 : \mathcal{X} \mapsto [0, 1]$. Most PG methods (e.g., episodic REINFORCE [Williams, 1992], PoWER [Kober & Peters, 2009], and Natural Actor Critic [Peters & Schaal, 2008]) optimize the policy by maximizing a lower bound on $\mathcal{J}(\boldsymbol{\theta})$, comparing trajectories generated by $\pi_{\boldsymbol{\theta}}$ against those generated by a new candidate policy $\pi_{\tilde{\boldsymbol{\theta}}}$. For details, see Kober & Peters [2009].

3.2 Lifelong Machine Learning

In a lifelong learning setting [Thrun, 1996; Ruvolo & Eaton, 2013], the learner faces multiple, consecutive tasks and must rapidly learn each new task by building upon its previous experience. The learner may encounter a previous task at any time, and so must optimize performance across all tasks seen so far. A priori, the agent does not know the total number of tasks $T_{\max}$, the task distribution, or the task order.

At time t, the lifelong learner encounters task $\mathcal{Z}^{(t)}$. In this paper, each task $\mathcal{Z}^{(t)}$ is specified by an MDP $\langle \mathcal{X}^{(t)}, \mathcal{A}^{(t)}, P^{(t)}, R^{(t)}, \gamma^{(t)} \rangle$, but the lifelong learning setting and our approach can equivalently handle classification or regression tasks. The agent will learn each task consecutively, acquiring training data (i.e., trajectories) in each task before advancing to the next. The agent's goal is to learn the optimal policies $\{\pi^*_{\boldsymbol{\theta}^{(1)}}, \ldots, \pi^*_{\boldsymbol{\theta}^{(T)}}\}$ with corresponding parameters $\{\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(T)}\}$, where T is the number of unique tasks seen so far ($1 \leq T \leq T_{\max}$). Ideally, knowledge learned from previous tasks $\{\mathcal{Z}^{(1)}, \ldots, \mathcal{Z}^{(T-1)}\}$ should accelerate training and improve performance on each new task $\mathcal{Z}^{(T)}$. Also, the lifelong learner should scale effectively to large numbers of tasks, learning each new task rapidly from minimal data.

The Efficient Lifelong Learning Algorithm (ELLA) [Ruvolo & Eaton, 2013] and PG-ELLA [Bou Ammar *et al.*, 2014] were developed to operate in this lifelong learning setting for classification/regression and RL tasks, respectively. Both approaches assume the parameters for each task model can be factorized using a shared knowledge base $\boldsymbol{L}$, facilitating transfer between tasks. Specifically, the model parameters for task $\mathcal{Z}^{(t)}$ are given by $\boldsymbol{\theta}^{(t)} = \boldsymbol{L}\boldsymbol{s}^{(t)}$, where $\boldsymbol{L} \in \mathbb{R}^{d\times k}$ is the shared basis over the model space, and $\boldsymbol{s}^{(t)} \in \mathbb{R}^k$ are the sparse coefficients over the basis. This factorization has been effective for transfer in both lifelong and multi-task learning [Kumar & Daumé, 2012; Maurer *et al.*, 2013].

Under this assumption, the MTL objective for PG is:

$$\min_{\boldsymbol{L},\boldsymbol{S}} \frac{1}{T}\sum_t \left[-\mathcal{J}(\boldsymbol{\theta}^{(t)}) + \mu\|\boldsymbol{s}^{(t)}\|_1\right] + \lambda\|\boldsymbol{L}\|_{\mathsf{F}}^2 \ , \qquad (1)$$

where $\boldsymbol{S} = [\boldsymbol{s}^{(1)} \cdots \boldsymbol{s}^{(T)}]$ is the matrix of sparse vectors, $\mathcal{J}(\boldsymbol{\theta}^{(t)})$ is the PG objective for task $\mathcal{Z}^{(t)}$, $\|\cdot\|_{\mathsf{F}}$ is the Frobenius norm, the L_1 norm is used to approximate the true vector sparsity of $\boldsymbol{s}^{(t)}$, and μ and λ are regularization parameters. To solve this objective in a lifelong learning setting, Bou Ammar et al. [2014] approximate the multi-task objective by first substituting in the lower-bound to the PG objective, then taking a second-order Taylor expansion to approximate the objective around an estimate $\boldsymbol{\alpha}^{(t)} \in \mathbb{R}^d$ of the single-task policy parameters for each task $\mathcal{Z}^{(t)}$, and updating only the coefficients $\boldsymbol{s}^{(t)}$ for the current task at each time step. This process reduces the MTL objective to the problem of sparse coding the single-task policies in the shared basis $\boldsymbol{L}$, and enables $\boldsymbol{S}$ and $\boldsymbol{L}$ to be solved efficiently by the following online update rules that constitute PG-ELLA [Bou Ammar *et al.*, 2014]:

$$\boldsymbol{s}^{(t)} \leftarrow \arg\min_{\boldsymbol{s}} \|\boldsymbol{\alpha}^{(t)} - \boldsymbol{L}\boldsymbol{s}\|_{\boldsymbol{\Gamma}^{(t)}}^2 + \mu\|\boldsymbol{s}\|_1 \qquad (2)$$

$$\boldsymbol{A} \leftarrow \boldsymbol{A} + (\boldsymbol{s}^{(t)}\boldsymbol{s}^{(t)\top}) \otimes \boldsymbol{\Gamma}^{(t)} \qquad (3)$$

$$\boldsymbol{b} \leftarrow \boldsymbol{b} + \mathrm{vec}\left(\boldsymbol{s}^{(t)\top} \otimes \left(\boldsymbol{\alpha}^{(t)\top}\boldsymbol{\Gamma}^{(t)}\right)\right) \qquad (4)$$

$$\boldsymbol{L} \leftarrow \mathrm{mat}\left(\left(\frac{1}{T}\boldsymbol{A} + \lambda\boldsymbol{I}_{kd}\right)^{-1}\frac{1}{T}\boldsymbol{b}\right) \ , \qquad (5)$$

where $\|\boldsymbol{v}\|_{\boldsymbol{A}}^2 = \boldsymbol{v}^\top\boldsymbol{A}\boldsymbol{v}$, the symbol $\otimes$ denotes the Kronecker product, $\boldsymbol{\Gamma}^{(t)}$ is the Hessian of the PG lower bound on $\mathcal{J}(\boldsymbol{\alpha}^{(t)})$, $\boldsymbol{I}_m$ is the $m \times m$ identity matrix, $\boldsymbol{A}$ is initialized to a $kd \times kd$ zero matrix, and $\boldsymbol{b} \in \mathbb{R}^{kd}$ is initialized to zeros.

Though effective for lifelong learning, this approach requires significant training data to estimate the policy for each new task before the learner can solve it. We eliminate this restriction by incorporating task descriptors into lifelong learning, enabling zero-shot transfer to new tasks.

4 Task Descriptors

While most MTL and lifelong learning methods use task training data to model inter-task relationships, high level descriptions can describe task differences. For example, in multi-task medical domains, patients are often grouped into tasks by demographic data and disease presentation [Oyen & Lane, 2012]. In control problems, the dynamical system parameters (e.g., the spring, mass, and damper constants in a spring-mass-damper system) describe the task. Descriptors can also be derived from external sources, such as Wikipedia [Pennington *et al.*, 2014]. Such task descriptors have been used extensively for zero-shot learning [Palatucci *et al.*, 2009; Socher *et al.*, 2013].

Formally, we assume that each task $\mathcal{Z}^{(t)}$ has an associated descriptor $\boldsymbol{m}^{(t)}$ that is given to the learner upon first presentation of the task. The learner has no knowledge of future tasks, or the distribution of task descriptors. The descriptor is represented by a feature vector $\phi\big(\boldsymbol{m}^{(t)}\big) \in \mathbb{R}^{d_m}$, where $\phi(\cdot)$ performs feature extraction and (possibly) a non-linear basis transformation on the features. We make no assumptions on the uniqueness of $\phi\big(\boldsymbol{m}^{(t)}\big)$, although in general tasks will have different descriptors.[1] In addition, each task also has associated training data $\boldsymbol{X}^{(t)}$ to learn the model; in the case of RL tasks, the data consists of trajectories that are dynamically acquired by the agent through experience in the environment.

5 Lifelong Learning with Task Descriptors

We incorporate task descriptors into lifelong learning via sparse coding with a coupled dictionary, enabling the descriptors and learned policies to augment each other. Although we focus on RL tasks, our method can easily be adapted to classification or regression, as described in the Appendix.[2]

5.1 Coupled Dictionary Optimization

As described previously, many multi-task and lifelong learning approaches have found success with factorizing the policy parameters $\boldsymbol{\theta}^{(t)}$ for each task as a sparse linear combination over a shared basis: $\boldsymbol{\theta}^{(t)} = \boldsymbol{L}\boldsymbol{s}^{(t)}$. In effect, each column of the shared basis $\boldsymbol{L}$ serves as a reusable policy component representing a cohesive chunk of knowledge. In lifelong learning, the basis $\boldsymbol{L}$ is refined over time as the system learns more tasks. The coefficient vectors $\boldsymbol{S} = [\boldsymbol{s}^{(1)} \ldots \boldsymbol{s}^{(T)}]$ encode the task policies in this shared basis, providing an embedding of the tasks based on how their policies share knowledge.

We make a similar assumption about the task descriptors—that the descriptor features $\phi\big(\boldsymbol{m}^{(t)}\big)$ can be linearly factorized[3] using a latent basis $\boldsymbol{D} \in \mathbb{R}^{d_m\times k}$ over the descriptor space. This basis captures relationships among the descriptors, with coefficients that similarly embed tasks based on commonalities in their descriptions. From a co-view perspective [Yu *et al.*, 2014], both the policies and descriptors provide information about the task, and so each can augment the learning of the other. Each underlying task is common to both views, and so we seek to find task embeddings that are consistent for *both* the policies and their corresponding task descriptors. We can enforce this by coupling the two bases $\boldsymbol{L}$ and $\boldsymbol{D}$, sharing the same coefficient vectors $\boldsymbol{S}$ to reconstruct both the policies and descriptors. Therefore, for task $\mathcal{Z}^{(t)}$,

$$\boldsymbol{\theta}^{(t)} = \boldsymbol{L}\boldsymbol{s}^{(t)} \qquad\qquad \phi\big(\boldsymbol{m}^{(t)}\big) = \boldsymbol{D}\boldsymbol{s}^{(t)} \ . \qquad (6)$$

To optimize the coupled bases $\boldsymbol{L}$ and $\boldsymbol{D}$ during the lifelong learning process, we employ techniques for coupled dictionary optimization from the sparse coding literature [Yang

[1]This raises the question of what descriptive features to use, and how task performance will change if some descriptive features are unknown. We explore these issues empirically in the Appendix.

[2]The online appendix is available on the third author's website.

[3]This is potentially non-linear w.r.t $\boldsymbol{m}^{(t)}$, since ϕ can be non-linear.

Algorithm 1 TaDeLL (k, λ, μ)

```
1:  $T \leftarrow 0$
2:  $\boldsymbol{L} \leftarrow$ RandomMatrix$_{d,k}$, $\boldsymbol{D} \leftarrow$ RandomMatrix$_{m,k}$
3:  while some task $\left(\mathcal{Z}^{(t)}, \phi\left(\boldsymbol{m}^{(t)}\right)\right)$ is available do
4:      if isNewTask($\mathcal{Z}^{(t)}$) then
5:          $T \leftarrow T + 1$
6:          $\mathbb{T}^{(t)} \leftarrow$ sampleRandomTrajectories($\mathcal{Z}^{(t)}$)
7:      else
8:          $\mathbb{T}^{(t)} \leftarrow$ sampleTrajectories($\mathcal{Z}^{(t)}$, $\pi_{\boldsymbol{\alpha}^{(t)}}$)
9:      end if
10:     Compute $\boldsymbol{\alpha}^{(t)}$ and $\boldsymbol{\Gamma}^{(t)}$ from $\mathbb{T}^{(t)}$
11:     $\boldsymbol{s}^{(t)} \leftarrow \arg\min_{\boldsymbol{s}} \left\|\boldsymbol{\beta}^{(t)} - \boldsymbol{K}\boldsymbol{s}\right\|^2_{\boldsymbol{A}^{(t)}} + \mu\|\boldsymbol{s}\|_1$
12:     $\boldsymbol{L} \leftarrow$ update$L(\boldsymbol{L}, \boldsymbol{s}^{(t)}, \boldsymbol{\alpha}^{(t)}, \boldsymbol{\Gamma}^{(t)}, \lambda)$
13:     $\boldsymbol{D} \leftarrow$ update$D(\boldsymbol{D}, \boldsymbol{s}^{(t)}, \phi(\boldsymbol{m}^{(t)}), \rho\boldsymbol{I}_{d_m}, \lambda)$
14:     for $t \in \{1, \ldots, T\}$ do: $\boldsymbol{\theta}^{(t)} \leftarrow \boldsymbol{L}\boldsymbol{s}^{(t)}$
15: end while
```

et al., 2010], which optimizes the dictionaries for multiple feature spaces that share a joint sparse representation. This notion of coupled dictionary learning has led to high performance algorithms for image super-resolution [Yang *et al.*, 2010], allowing the reconstruction of high-res images from low-res samples, and for multi-modal retrieval [Zhuang *et al.*, 2013] and cross-domain retrieval [Yu *et al.*, 2014].

Given the factorization in Eq. 6, we can re-formulate the multi-task objective (Eq. 1) for the coupled dictionaries as

$$\min_{\boldsymbol{L},\boldsymbol{D},\boldsymbol{S}} \frac{1}{T}\sum_t \left[-\mathcal{J}\left(\boldsymbol{\theta}^{(t)}\right) + \rho\left\|\phi\left(\boldsymbol{m}^{(t)}\right) - \boldsymbol{D}\boldsymbol{s}^{(t)}\right\|_2^2 + \mu\left\|\boldsymbol{s}^{(t)}\right\|_1\right] + \lambda(\|\boldsymbol{L}\|_{\mathsf{F}}^2 + \|\boldsymbol{D}\|_{\mathsf{F}}^2) \ , \quad (7)$$

where ρ balances the policy's fit to the task descriptor's fit.

To solve Eq. 7 online, we approximate $\mathcal{J}(\cdot)$ by a second-order Taylor expansion around $\boldsymbol{\alpha}^{(t)}$, the minimizer for the PG lower bound of $\mathcal{J}(\cdot)$ (i.e., $\pi_{\boldsymbol{\alpha}^{(t)}}$ is the single-task policy for $\mathcal{Z}^{(t)}$ based on the observed trajectories), following Bou Ammar et al. [2014]. This simplifies Eq. 7 to

$$\min_{\boldsymbol{L},\boldsymbol{D},\boldsymbol{S}} \frac{1}{T}\sum_t \left[\left\|\boldsymbol{\alpha}^{(t)} - \boldsymbol{L}\boldsymbol{s}^{(t)}\right\|^2_{\boldsymbol{\Gamma}^{(t)}} + \rho\left\|\phi\left(\boldsymbol{m}^{(t)}\right) - \boldsymbol{D}\boldsymbol{s}^{(t)}\right\|_2^2 + \mu\left\|\boldsymbol{s}^{(t)}\right\|_1\right] + \lambda(\|\boldsymbol{L}\|_{\mathsf{F}}^2 + \|\boldsymbol{D}\|_{\mathsf{F}}^2) \ . \quad (8)$$

We can merge pairs of terms in Eq. 8 by choosing:

$$\boldsymbol{\beta}^{(t)} = \begin{bmatrix} \boldsymbol{\alpha}^{(t)} \\ \phi\left(\boldsymbol{m}^{(t)}\right)\end{bmatrix} \quad \boldsymbol{K} = \begin{bmatrix}\boldsymbol{L} \\ \boldsymbol{D}\end{bmatrix} \quad \boldsymbol{A}^{(t)} = \begin{bmatrix}\boldsymbol{\Gamma}^{(t)} & \boldsymbol{0} \\ \boldsymbol{0} & \rho\boldsymbol{I}_{d_m}\end{bmatrix} \ ,$$

where $\boldsymbol{0}$ is the zero matrix, letting us rewrite (8) concisely as

$$\min_{\boldsymbol{K},\boldsymbol{S}} \frac{1}{T}\sum_t \left[\left\|\boldsymbol{\beta}^{(t)} - \boldsymbol{K}\boldsymbol{s}^{(t)}\right\|^2_{\boldsymbol{A}^{(t)}} + \mu\left\|\boldsymbol{s}^{(t)}\right\|_1\right] + \lambda\|\boldsymbol{K}\|_{\mathsf{F}}^2 \ . \quad (9)$$

This objective can now be solved efficiently online, as a series of per-task update rules given in Algorithm 1. $\boldsymbol{L}$ and $\boldsymbol{D}$ are updated independently using Equations 3–5, following a recursive construction based on an eigenvalue decomposition.

The complete implementation of our approach is available on the third author's website.

Algorithm 2 Zero-Shot Transfer to a New Task $\mathcal{Z}^{(t_{new})}$

```
1: Inputs: task descriptor $\boldsymbol{m}^{(t_{new})}$, learned bases $\boldsymbol{L}$ and $\boldsymbol{D}$
2: $\tilde{\boldsymbol{s}}^{(t_{new})} \leftarrow \arg\min_{\boldsymbol{s}} \left\|\phi\left(\boldsymbol{m}^{(t_{new})}\right) - \boldsymbol{D}\boldsymbol{s}\right\|_2^2 + \mu\|\boldsymbol{s}\|_1$
3: $\tilde{\boldsymbol{\theta}}^{(t_{new})} \leftarrow \boldsymbol{L}\tilde{\boldsymbol{s}}^{(t_{new})}$
4: Return: $\pi_{\tilde{\boldsymbol{\theta}}^{(t_{new})}}$
```

5.2 Zero-Shot Transfer Learning

In a lifelong setting, when faced with a new task, the agent's goal is to learn an effective policy for that task as quickly as possible. At this stage, previous multi-task and lifelong learners incurred a delay before they could produce a decent policy, since they needed to acquire data from the new task in order to identify related knowledge and train the new policy.

Incorporating task descriptors enables our approach to predict a policy for the new task immediately, given *only* the descriptor. This ability to perform zero-shot transfer is enabled by the use of coupled dictionary learning, which allows us to observe a data instance in one feature space (i.e., the task descriptor), and then recover its underlying latent signal in the other feature spaces (i.e., the policy parameters) using the dictionaries and sparse coding [Yang *et al.*, 2010].

Given only the descriptor $\boldsymbol{m}^{(t_{new})}$ for a new task $\mathcal{Z}^{(t_{new})}$, we can estimate the embedding of the task in the latent descriptor space via LASSO on the learned dictionary $\boldsymbol{D}$:

$$\tilde{\boldsymbol{s}}^{(t_{new})} \leftarrow \arg\min_{\boldsymbol{s}} \left\|\phi\left(\boldsymbol{m}^{(t_{new})}\right) - \boldsymbol{D}\boldsymbol{s}\right\|_2^2 + \mu\|\boldsymbol{s}\|_1 \ . \quad (10)$$

Since the estimate given by $\tilde{\boldsymbol{s}}^{(t_{new})}$ also serves as the coefficients over the latent policy space $\boldsymbol{L}$, we can immediately predict a policy for the new task as: $\tilde{\boldsymbol{\theta}}^{(t_{new})} = \boldsymbol{L}\tilde{\boldsymbol{s}}^{(t_{new})}$. This zero-shot transfer learning procedure is given as Algorithm 2.

5.3 Theoretical Analysis

This section discusses why incorporating task descriptors through coupled dictionaries can improve performance of the learned policies and enable zero-shot transfer to new tasks. In the Appendix[2], we also prove the convergence of TaDeLL. A full sample complexity analysis is beyond the scope of this paper, and, indeed, remains an open problem for zero-shot learning [Romera-Paredes & Torr, 2015].

To analyze the policy improvement, since the policy parameters are factored as $\boldsymbol{\theta}^{(t)} = \boldsymbol{L}\boldsymbol{s}^{(t)}$, we proceed by showing that incorporating the descriptors through coupled dictionaries can improve both $\boldsymbol{L}$ and $\boldsymbol{S}$. In this analysis, we employ the concept of *mutual coherence*, which has been studied extensively in the sparse recovery literature. Mutual coherence measures the similarity of a dictionary's elements as $M(\boldsymbol{Q}) = \max_{1\leq i \neq j \leq n}\left(\frac{|\boldsymbol{q}_i^\top \boldsymbol{q}_j|}{\|\boldsymbol{q}_i\|_2\|\boldsymbol{q}_j\|_2}\right) \in [0,1]$, where $\boldsymbol{q}_i$ is the i^{th} column of a dictionary $\boldsymbol{Q} \in \mathbb{R}^{d\times k}$. If $M(\boldsymbol{Q}) = 0$, then $\boldsymbol{Q}$ is an invertible orthogonal matrix and so sparse recovery can be solved directly by inversion; $M(\boldsymbol{Q}) = 1$ implies that $\boldsymbol{Q}$ is not full rank and a poor dictionary. Intuitively, low mutual coherence indicates that the dictionary's columns are considerably different, and thus such a "good" dictionary can represent many different policies, potentially yielding more knowledge transfer. This intuition is shown in the following:

Theorem 5.1. *[Donoho* et al.*, 2006] Suppose we have noisy observations $\hat{\boldsymbol{\theta}}$ of the linear system $\boldsymbol{\theta} = \boldsymbol{Q}\boldsymbol{s}$, such that $\|\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}\|_2 \leq \epsilon$. Let $\boldsymbol{s}^*$ be a solution to the system, and let $K = \|\boldsymbol{s}\|_0$. If $K < 0.5(1 + M(\boldsymbol{Q})^{-1})$, then $\boldsymbol{s}^*$ is the unique sparsest solution of the system. Moreover, if $\boldsymbol{s}^+$ is the LASSO solution for the system from the noisy observations, then: $\|\boldsymbol{s}^* - \boldsymbol{s}^+\|_2^2 \leq \frac{4\epsilon^2}{1-M(\boldsymbol{Q})(4K-1)}$.*

Therefore, an $\boldsymbol{L}$ with low mutual coherence would lead to more stable solutions of the $\boldsymbol{\theta}^{(t)}$'s against inaccurate single-task estimates of the policies (the $\boldsymbol{\alpha}^{(t)}$'s). We next show that our approach likely lowers the mutual coherence of $\boldsymbol{L}$.

TaDeLL alters the problem from training $\boldsymbol{L}$ to training the coupled dictionaries $\boldsymbol{L}$ and $\boldsymbol{D}$ (contained in $\boldsymbol{K}$). Let $\boldsymbol{s}^{*(t)}$ be the solution to Eq. 1 for task $\mathcal{Z}^{(t)}$, which is unique under sparse recovery theory, so $\left\|\boldsymbol{s}^{*(t)}\right\|_0$ remains unchanged for all tasks. Theorem 5.1 implies that, if $M(\boldsymbol{K}) < M(\boldsymbol{L})$, coupled dictionary learning can help with a more accurate recovery of the $\boldsymbol{s}^{(t)}$'s. To show this, we note that Eq. 7 can also be derived as a result of an MAP estimate from a Bayesian perspective, enforcing a Laplacian distribution on the $\boldsymbol{s}^{(t)}$'s and assuming $\boldsymbol{L}$ to be a Gaussian matrix with elements drawn i.i.d: $l_{ij} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2)$. When considering such a random matrix $\boldsymbol{Q} \in \mathbb{R}^{d \times k}$, Donoho & Huo [2001] proved that asymptotically $M(\boldsymbol{Q}) \propto \sqrt{\frac{\log(dk)}{d}}$ as $d \to \infty$. Using this as an estimate for $M(\boldsymbol{L})$ and $M(\boldsymbol{K})$, since incorporating task descriptors increases d, most likely $M(\boldsymbol{K}) < M(\boldsymbol{L})$, implying that TaDeLL learns a superior dictionary. Moreover, if $M(\boldsymbol{D}) \leq M(\boldsymbol{L})$, the theorem implies we can use $\boldsymbol{D}$ alone to recover the task policies through zero-shot transfer.

To show that task features can also improve the sparse recovery, we rely on the following theorem about LASSO:

Theorem 5.2. *[Negahban* et al.*, 2009] Let $\boldsymbol{s}^*$ be a unique solution to the system $\boldsymbol{\theta} = \boldsymbol{Q}\boldsymbol{s}$ with $\|\boldsymbol{s}\|_0 = K$ and $\boldsymbol{Q} \in \mathbb{R}^{d \times k}$. If $\boldsymbol{s}^+$ is the LASSO solution for the system from noisy observations, then with high probability: $\|\boldsymbol{s}^* - \boldsymbol{s}^+\|_2 \leq c'\sqrt{K\frac{\log k}{d}}$, where the constant $c' \in \mathbb{R}^+$ depends on properties of the linear system and observations.*

This theorem shows that the error reconstruction for LASSO is proportional to $\frac{1}{\sqrt{d}}$. When we incorporate the descriptor through $\boldsymbol{\beta}^{(t)}$, the RHS denominator increases from d to $(d + d_m)$ while K and k remain constant, yielding a tighter fit. Therefore, task descriptors can improve learned dictionary quality and sparse recovery accuracy. To ensure an equivalently tight fit for $\boldsymbol{s}^{(t)}$ using either policies or descriptors, Theorem 5.2 suggests it should be that $d_m \geq d$ to ensure that zero-shot learning yields similarly tight estimates of $\boldsymbol{s}^{(t)}$.

Computational Complexity Each update begins with one PG step to update $\boldsymbol{\alpha}^{(t)}$ and $\boldsymbol{\Gamma}^{(t)}$ at a cost of $O(\xi(d, n_t))$, where $\xi()$ depends on the base PG learner and n_t is the number of trajectories obtained for task $\mathcal{Z}^{(t)}$. The cost of updating $\boldsymbol{L}$ and $\boldsymbol{s}^{(t)}$ alone is $O(k^2 d^3)$ [Ruvolo & Eaton, 2013], and so the cost of updating $\boldsymbol{K}$ through coupled dictionary learning is $O(k^2(d + d_m)^3)$. This yields an overall per-update cost of $O(k^2(d + d_m)^3 + \xi(d, n_t))$, which is independent of T.

6 Experiments

We evaluated our method on learning control policies for three benchmark systems and an application to quadrotors.

6.1 Benchmark Dynamical Systems

Spring Mass Damper (SM) The SM system is described by three parameters: the spring constant, mass, and damping constant. The system's state is given by the position and velocity of the mass. The controller applies a force to the mass, attempting to stabilize it to a given position.

Cart Pole (CP) The CP system involves balancing an inverted pendulum by applying a force to the cart. The system is characterized by the cart and pole masses, pole length, and a damping parameter. The states are the position and velocity of the cart and the angle and rotational velocity of the pole.

Bicycle (BK) This system focuses on keeping a bicycle balanced upright as it rolls along a horizontal plane at constant velocity. The system is characterized by the bicycle mass, x- and z-coordinates of the center of mass, and parameters relating to the shape of the bike (the wheelbase, trail, and head angle). The state is the bike's tilt and its derivative; the actions are the torque applied to the handlebar and its derivative.

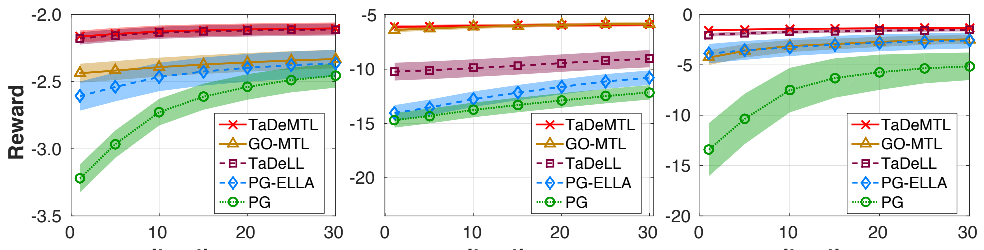
6.2 Methodology

In each domain we generated 40 tasks, each with different dynamics, by varying the system parameters. The reward for each task was taken to be the distance between the current state and the goal. For lifelong learning, tasks were encountered consecutively with repetition, and learning proceeded until each task had been seen at least once. We used the same random task order between methods to ensure fair comparison. The learners sampled trajectories of 100 steps, and the learning session during each task presentation was limited to 30 iterations. For MTL, all tasks weres presented simultaneously. We used Natural Actor Critic [Peters & Schaal, 2008] as the base learner for the benchmark systems and episodic REINFORCE [Williams, 1992] for quadrotor control. We chose k and the regularization parameters independently for each domain to optimize the combined performance of all methods on 20 held-out tasks, and set $\rho = mean(diag(\rho^{(t)}))$ to balance the fit to the descriptors and the policies. We measured learning curves based on the final policies for each of the 40 tasks, averaging results over seven trials. The system parameters for each task were used as the task descriptor features $\phi(\boldsymbol{m})$; we also tried several non-linear transformations as $\phi(\cdot)$, but found the linear features worked well.

6.3 Results on Benchmark Systems

Figure 1 compares our TaDeLL approach for lifelong learning with task descriptors to 1.) PG-ELLA [Bou Ammar *et al.*, 2014], which does not use task features, 2.) GO-MTL [Kumar & Daumé, 2012], the MTL optimization of Eq. 1, and 3.) single-task learning using PG. For comparison, we also performed an offline MTL optimization of Eq. 7 via alternating optimization, and plot the results as TaDeMTL. The shaded regions on the plots denote standard error bars.

We see that task descriptors improve lifelong learning on every system, even driving performance to a level that is unachievable from training the policies from experience alone



(a) Simple Mass (b) Cart Pole (c) Bicycle

Figure 1: Performance of multi-task (solid lines), lifelong (dashed), and single-task learning (dotted) on benchmark dynamical systems. (Best viewed in color.)

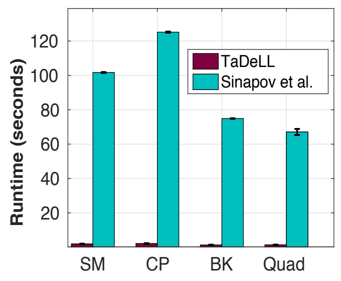
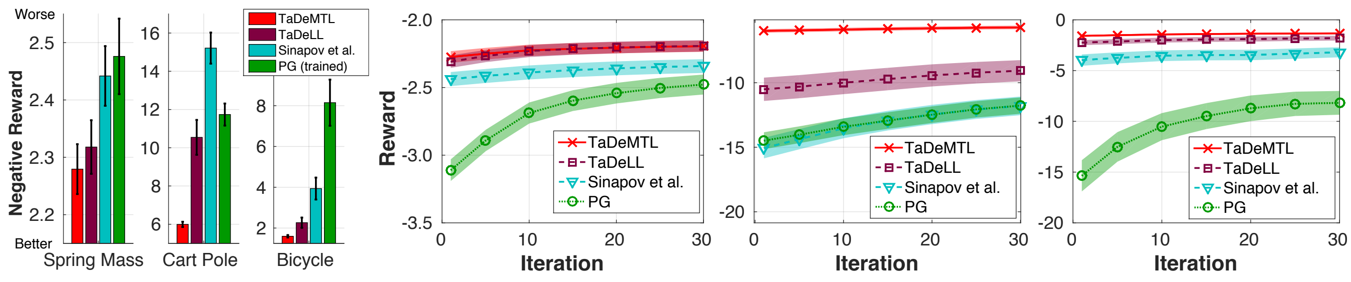


Figure 2: Runtime comparison.



(a) Jumpstart (b) Warm Start: Simple Mass (c) Warm Start: Cart Pole (d) Warm Start: Bicycle

Figure 3: Zero-shot transfer to new tasks. Figure (a) shows the initial "jumpstart" improvement on each task domain; Figures (b)–(d) depict the result of using zero-shot policies as warm start initializations for PG. (Best viewed in color.)

via GO-MTL in the SM and BK domains. The difference between TaDeLL and TaDeMTL is also negligible for all domains except CP (which had very diverse tasks), demonstrating the effectiveness of our online optimization.

Figure 3 shows that task descriptors are effective for zero-shot transfer to new tasks. We generated an additional 40 tasks for each domain to measure zero-shot performance, averaging results over these new tasks. Figure 3a shows that our approach improves the initial performance (i.e., the "jumpstart" [Taylor & Stone, 2009]) on new tasks, outperforming Sinapov et al. [2015]'s method and single-task PG, which was allowed to train on the task. We attribute the especially poor performance of Sinapov et al. on CP to the fact that the CP policies differ substantially; in domains where the source policies are vastly different from the target policies, Sinapov et al.'s algorithm does not have an appropriate source to transfer. Their approach is also much more computationally expensive (quadratic in the number of tasks) than our approach (linear in the number of tasks), as shown in Figure 2; details of the runtime experiments are included in the Appendix[2]. Figures 3b–3d show that the zero-shot policies can be used effectively as a warm start initialization for a PG learner, which is then allowed to improve the policy.

6.4 Application to Quadrotor Control

We also applied our approach to the more challenging domain of quadrotor control, focusing on zero-shot transfer to new stability tasks. To ensure realistic dynamics, we use the model of Bouabdallah and Siegwart [2005], which has been verified on physical systems. The quadrotors are characterized by three inertial constants and the arm length, with their state consisting of roll/pitch/yaw and their derivatives.

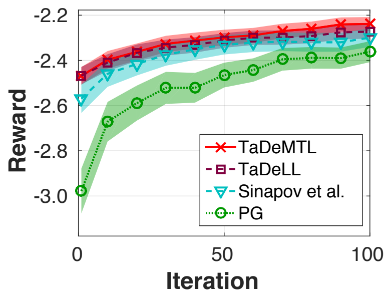


Figure 4: Warm start learning on quadrotor control.

Figure 4 shows the results of our application, demonstrating that TaDeLL can predict a controller for new quadrotors through zero-shot learning that has equivalent accuracy to PG, which had to train on the system. As with the benchmarks, TaDeLL is effective for warm start learning with PG.

7 Conclusion

We proposed a coupled dictionary method for incorporating task descriptors into lifelong learning, showing that descriptors improve learned policy performance, and enable us to predict policies for new tasks before observing training data. Experiments demonstrate that our method outperforms other approaches on dynamical control problems, and requires substantially less computational time than similar methods.

Acknowledgments

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

References

[Ando & Zhang, 2005] Rie Kubota Ando & Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, 2005.

[Bakker & Heskes, 2003] Bart Bakker & Tom Heskes. Task clustering and gating for Bayesian multi-task learning. *Journal of Machine Learning Research*, 4:83–99, 2003.

[Baxter, 2000] Jonathan Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

[Ben-David *et al.*, 2007] Shai Ben-David, John Blitzer, Koby Crammer, & Fernando Pereira. Analysis of representations for domain adaptation. *Neural Information Processing Systems*, 19:137–144, 2007.

[Bickel *et al.*, 2009] Steffen Bickel, Christoph Sawade, & Tobias Scheffer. Transfer learning by distribution matching for targeted advertising. *Neural Information Processing Systems*, 2009.

[Bonilla *et al.*, 2007] Edwin Bonilla, Felix Agakov, & Christopher Williams. Kernel multi-task learning using task-specific features. *Int. Conference on Artificial Intelligence and Statistics*, 2007.

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

[Bou Ammar *et al.*, 2015] Haitham Bou Ammar, Eric Eaton, Jose Marcio Luna, & Paul Ruvolo. Autonomous cross-domain knowledge transfer in lifelong policy gradient reinforcement learning. *International Joint Conference on Artificial Intelligence*, 2015.

[Bouabdallah & Siegwart, 2005] Samir Bouabdallah & Roland Siegwart. Backstepping and sliding-mode techniques applied to an indoor micro quadrotor. *IEEE International Conference on Robotics and Automation.*, 2005.

[Caruana, 1997] Rich Caruana. Multitask Learning. *Machine Learning*, 28:41–75, 1997.

[Donoho & Huo, 2001] David Donoho & Xiaoming Huo. Uncertainty principles and ideal atomic decomposition. *IEEE Transactions on Information Theory*, 47(7):2845–2862, 2001.

[Donoho *et al.*, 2006] David Donoho, Michael Elad, & Vladimir Temlyakov. Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Transactions on Information Theory*, 52(1):6–18, 2006.

[Kober & Peters, 2009] Jens Kober & Jan Peters. Policy search for motor primitives in robotics. *Neural Information Processing Systems*, pp. 849–856, 2009.

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

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

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

[Negahban *et al.*, 2009] Sahand Negahban, Bin Yu, Martin Wainwright, & Pradeep Ravikumar. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. *Neural Information Processing Systems*, 2009.

[Oyen & Lane, 2012] Diane Oyen & Terran Lane. Leveraging domain knowledge in multitask Bayesian network structure learning. *AAAI Conference on Artificial Intelligence*, 2012.

[Palatucci *et al.*, 2009] Mark Palatucci, Geoffrey Hinton, Dean Pomerleau, & Tom Mitchell. Zero-shot learning with semantic output codes. *Neural Information Processing Systems*, 2009.

[Parameswaran & Weinberger, 2010] Shibin Parameswaran & Kilian Q. Weinberger. Large margin multi-task metric learning. *Neural Information Processing Systems*, pp. 1867–1875, 2010.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, & Christopher Manning. GloVe: Global vectors for word representation. *Empiricial Methods in Natural Language Processing*, 12:1532–1543, 2014.

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

[Romera-Paredes & Torr, 2015] Bernardino Romera-Paredes & Philip Torr. An embarrassingly simple approach to zero-shot learning. *International Conference on Machine Learning*, 2015.

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

[Sinapov *et al.*, 2015] Jivko Sinapov, Sanmit Narvekar, Matteo Leonetti, & Peter Stone. Learning inter-task transferability in the absence of target task samples. *International Conference on Autonomous Agents and Multiagent Systems*, 2015.

[Socher *et al.*, 2013] Richard Socher, Milind Ganjoo, Christopher Manning, & Andrew Ng. Zero-shot learning through cross-modal transfer. *Neural Information Processing Systems*, 2013.

[Sutton *et al.*, 1999] Richard Sutton, David McAllester, Satinder Singh, & Yishay Mansour. 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.

[Thrun, 1996] Sebastian Thrun. Is learning the n-th thing any easier than learning the first? *Neural Information Processing Systems*, pp. 640–646, 1996.

[Wang *et al.*, 2014] Qifan Wang, Lingyun Ruan, & Luo Si. Adaptive knowledge transfer for multiple instance learning in image classification. *AAAI Conference on Artificial Intelligence*, 2014.

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

[Yang *et al.*, 2010] Jianchao Yang, John Wright, Thomas Huang, & Yi Ma. Image super-resolution via sparse representation. *IEEE Transactions on Image Processing*, 19(11):2861–2873, 2010.

[Yu *et al.*, 2014] Zhou Yu, Fei Wu, Yi Yang, Qi Tian, Jiebo Luo, & Yueting Zhuang. Discriminative coupled dictionary hashing for fast cross-media retrieval. *International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2014.

[Zhuang *et al.*, 2013] Yue Ting Zhuang, Yan Fei Wang, Fei Wu, Yin Zhang, & Wei Ming Lu. Supervised coupled dictionary learning with group structures for multi-modal retrieval. *AAAI Conference on Artificial Intelligence*, 2013.

A Appendix

This appendix provides further details on the approach, theory, and experiments described in the main paper.

A.1 Extension to Classification and Regression

Although we derived our approach for RL tasks, TaDeLL can easily be adapted to classification or regression by the following modifications to Algorithm 1:

1. Instead of sampling trajectories from the environment, each time step TaDeLL would observe a labeled training set $(\boldsymbol{X}^{(t)}, \boldsymbol{y}^{(t)})$ for task $\mathcal{Z}^{(t)}$, where $\boldsymbol{X}^{(t)} \subseteq \mathbb{R}^{n_t \times d}$. For classification tasks, $\boldsymbol{y}^{(t)} \in \{+1, -1\}^{n_t}$, and for regression tasks, $\boldsymbol{y}^{(t)} \in \mathbb{R}^{n_t}$.
2. Set $\boldsymbol{\alpha}^{(t)}$ to be the parameters of a single-task model trained via classification or regression (e.g., logistic or linear regression) on that data set, as described by Ruvolo and Eaton [2013].
3. Set $\boldsymbol{\Gamma}^{(t)}$ to be the Hessian of the corresponding loss function around the single-task solution $\boldsymbol{\alpha}^{(t)}$ (available in [Ruvolo & Eaton, 2013]), and
4. Using the reconstructed $\boldsymbol{\theta}^{(t)}$ as the model parameters for the corresponding classification or regression model for that task.

This modification procedure would support any parametric classification or regression base learner, such as logistic or linear regression, where the loss function is second-order differentiable in order to compute the Hessian.

A.2 Theoretical Convergence of TaDeLL

In this section, we prove the convergence of TaDeLL, showing that the learned dictionaries become increasingly stable as it learns more tasks. We build upon the theoretical results from Bou Ammar et al. [2014] and Ruvolo & Eaton [2013], demonstrating that these results apply to coupled dictionary learning with task descriptors, and use them to prove convergence.

Let $\hat{g}_T(\mathrm{L})$ represent the sparse coded approximation to the MTL objective, which can be defined as:

$$\hat{g}_T(\boldsymbol{L}) = \frac{1}{T}\sum_{t=1}^{T} \|\boldsymbol{\alpha}^{(t)} - \boldsymbol{L}\boldsymbol{s}^{(t)}\|^2_{\boldsymbol{\Gamma}^{(t)}} + \mu\|\boldsymbol{s}^{(t)}\|_1 + \lambda\|\boldsymbol{L}\|^2_{\mathsf{F}} \ .$$

This equation can be viewed as the cost for $\boldsymbol{L}$ when the sparse coefficients are kept constant. Let $\boldsymbol{L}_T$ be the version of the dictionary $\boldsymbol{L}$ obtained after observing T tasks. Given these definitions, we consider the following theorem:

Theorem A.1. *[Ruvolo & Eaton, 2013]*

1. *The trained dictionary $\boldsymbol{L}$ is stabilized over learning with rate: $\boldsymbol{L}_T - \boldsymbol{L}_{T-1} = O(\frac{1}{T})$*
2. *$\hat{g}_T(\boldsymbol{L}_T)$ converges almost surely.*
3. *$\hat{g}_T(\boldsymbol{L}_T) - \hat{g}_T(\boldsymbol{L}_{T-1})$ converges almost surely to zero.*

This theorem requires two conditions:

1. The tuples $\boldsymbol{\Gamma}^{(t)}$, $\boldsymbol{\alpha}^{(t)}$ are drawn i.i.d from a distribution with compact support to bound the norms of $\boldsymbol{L}$ and $\boldsymbol{s}^{(t)}$.
2. For all t, let $\boldsymbol{L}_\kappa$ be the subset of the dictionary $\boldsymbol{L}_t$, where only columns corresponding to non-zero element of $\boldsymbol{s}^{(t)}$ are included. Then, all eigenvalues of the matrix $\boldsymbol{L}_\kappa^\mathsf{T}\boldsymbol{\Gamma}^{(t)}\boldsymbol{L}_\kappa$ need to be strictly positive.

Bou Ammar et al. [2014] show that both of these conditions are met for the lifelong learning framework given in Eqs. 2–5. When we incorporate the task descriptors into this framework, we alter $\boldsymbol{\alpha}^{(t)} \to \boldsymbol{\beta}^{(t)}$, $\boldsymbol{L} \to \boldsymbol{K}$, and $\boldsymbol{\Gamma}^{(t)} \to \boldsymbol{A}^{(t)}$. Note both $\boldsymbol{\beta}^{(t)}$ and $\boldsymbol{A}^{(t)}$ are formed by adding deterministic entries and thus can be considered to be drawn i.i.d (because $\boldsymbol{\Gamma}^{(t)}$ and $\boldsymbol{\alpha}^{(t)}$ are assumed to be drawn i.i.d). Therefore, incorporating task descriptors does not violate Condition 1.

To show that Condition 2 holds, if we analogously form $\boldsymbol{K}_\kappa$, then the eigenvalues of $\boldsymbol{K}_\kappa$ are strictly positive because they are either eigenvalues of $\boldsymbol{L}$ (which are strictly positive according to [Bou Ammar *et al.*, 2014]) or the regularizing parameter ρ by definition. Thus, both conditions are met and convergence follows directly from Theorem A.1.

A.3 Additional Experiments

This section describes several additional experiments that were omitted from the main paper due to space limitations, exploring 1.) the choice of task descriptor features, 2.) zero-shot performance as the number of tasks varies, and 3.) diversity of the policies in each task domain.

Choice of Task Descriptor Features

In the main paper, we used the system parameters as the task descriptor features. However, it is unclear exactly how the choice of task descriptor features might affect the resulting performance. In other cases, we may have only partial knowledge of the system parameters. To address these questions, we conducted additional experiments on the Spring-Mass (SM) system, using various subsets of the task descriptor features when learning the coupled dictionaries.

Figure 5 shows how the number and selection of parameters affects performance on the SM domain. We evaluated jumpstart performance when using all possible subsets of the system parameters as the task descriptor features. These subsets of the SM system parameters (mass M, damping constant D, and spring constant K) are shown along the horizontal axis for the task descriptors. Overall, the results show that the learner performs better when using larger subsets of the system parameters (i.e., a better characterization of the system) as the task descriptors.

Performance for Various Numbers of Tasks

Although we have shown in Section A.2 that the learned dictionaries become more stable as the system learns more tasks, we cannot currently guarantee that this will improve the performance of zero-shot transfer. To evaluate the effect of the number of tasks on zero-shot performance, we conducted an additional set of experiments on the Simple-Mass domain. Our results, shown in Figure 6, reveal that zero-shot performance does indeed improve as the dictionaries are trained over more tasks. This improvement is most stable and rapid in an MTL setting, since the optimization over all dictionaries and task policies is run to convergence, but TaDeLL also shows clear improvement in zero-shot performance as

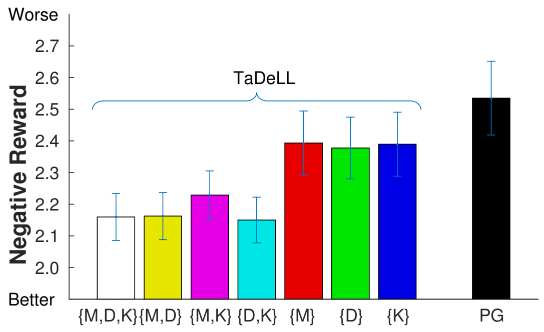


Figure 5: Spring-mass system performance using various subsets of the SM system parameters (mass M, damping constant D, and spring constant K) as the task descriptors.

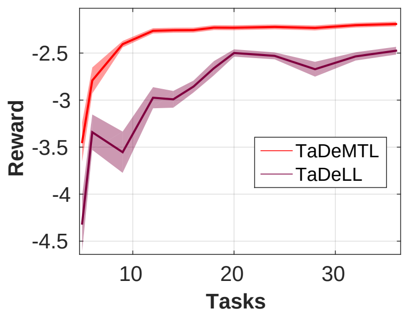


Figure 6: Zero-shot performance as a function of the number of SM tasks used to train the dictionary. As more tasks are used, the performance of the zero-shot transfer improves.

T_{max} increases. Since zero-shot transfer involves only the learned coupled dictionaries, we can conclude that the quality of these dictionaries for zero-shot transfer improves as the system learns more tasks.

Diversity of Policies in the Cart-Pole Domain

Figures 1 and 3 revealed that the performance difference between TaDeLL and TaDeMTL was greater in the Cart-Pole domain than the other domains, in which their performance difference was negligible. Sinapov et al.'s method also had lower performance on the Cart-Pole tasks. Based on this, we hypothesized that the Cart-Pole tasks were more diverse than in the other domains, requiring substantially different policies. To empirically verify this hypothesis, we plotted the first two parameters of the learned policies in each domain, as shown in Figure 7. As we see, the learned Cart-Pole policies are much more diverse than for the other tasks, explaining the greater challenge provided by the CP domain.

A.4 Computational Efficiency

This section provides details on the runtime experiments, which were included as Figure 2 in the main paper.

We compared the average per-task runtime of our approach to that of Sinapov et al. [2015], the most closely related method to our approach. Since Sinapov et al.'s method requires training transferability predictors between all pairs of

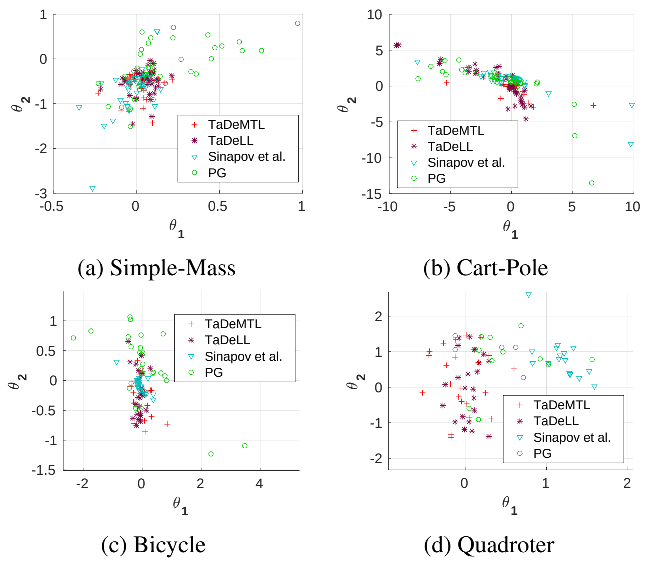


Figure 7: Visualization of the learned policies, revealing that the CP policies are more diverse than the other domains, and the different methods often learn very different policies.

tasks, its total runtime grows quadratically with the number of tasks. In comparison, our online algorithm is highly efficient. As shown in Section 5.3, the per-update cost of TaDeLL is $O\left(k^2(d+m)^3 + \xi(d, n_t)\right)$. Note that this per-update cost is independent of the number of tasks T, giving TaDeLL a total runtime that scales linearly in the number of tasks.

Figure 2 shows the per-task runtime for each algorithm based on a set of 40 tasks, as evaluated on an Intel Core I7-4700HQ CPU. TaDeLL samples tasks randomly with replacement and terminates once every task has been seen. For Sinapov et al., we used 10 PG iterations for calculating the warm start, ensuring fair comparison between the methods. These results show a substantial reduction in computational time for TaDeLL: two orders of magnitude over the 40 tasks.

A.5 Errata

The version of this paper that appeared in the IJCAI'16 proceedings omitted the t_{new} superscript in several places in Section 5.2. This version corrects those minor notational errors.