Active Task Selection for Lifelong Machine Learning

Paul Ruvolo and **Eric Eaton**
Bryn Mawr College
Computer Science Department
101 North Merion Avenue, Bryn Mawr, PA 19010
{pruvolo, eeaton}@cs.brynmawr.edu

Abstract

In a lifelong learning framework, an agent acquires knowledge incrementally over consecutive learning tasks, continually building upon its experience. Recent lifelong learning algorithms have achieved nearly identical performance to batch multi-task learning methods while reducing learning time by three orders of magnitude. In this paper, we further improve the scalability of lifelong learning by developing curriculum selection methods that enable an agent to actively select the next task to learn in order to maximize performance on future learning tasks. We demonstrate that active task selection is highly reliable and effective, allowing an agent to learn high performance models using up to 50% fewer tasks than when the agent has no control over the task order. We also explore a variant of transfer learning in the lifelong learning setting in which the agent can focus knowledge acquisition toward a particular target task.

Introduction

Current multi-task learning (MTL) algorithms improve performance by sharing learned knowledge between multiple task models. With few exceptions, MTL has focused primarily on learning in a batch framework, where all tasks are given at once. However, versatile learning agents need to be able to learn tasks consecutively, continually building upon their knowledge over a lifetime of experience.

In such a *lifelong learning* framework, tasks arrive sequentially and the learning agent's goal is to maximize its performance across all tasks. Recent lifelong learning methods (Ruvolo and Eaton 2013; Saha et al. 2011) have demonstrated the ability to learn multiple tasks online, yielding nearly identical model performance to batch MTL with over 1,000x speedup in learning. These methods do not control the order in which they learn the tasks; in the basic lifelong learning setting the task order is outside the agent's control.

However, consider a versatile personal assistant agent that supports lifelong learning. In some situations the agent may not have control over the task order (which could be specified by its boss), but in other situations it may have knowledge of the next several tasks it needs to learn. In these cases, it can intelligently choose the next task to learn in order to maximize its overall performance. Previous research on curriculum selection (Bengio et al. 2009) has affirmed that proper task ordering can improve learning performance.

In this paper, we explore the use of *active curriculum selection* to improve the scalability of lifelong learning over a long sequence of tasks. Specifically, we focus on a setting in which a lifelong learner can choose the next task to learn from a pool of candidate tasks in order to maximize future learning performance using as few tasks as possible. Learning in a task-efficient manner is crucial not only for interactive settings where learning each task requires extensive input from a human, but also in autonomous settings where learning each task may be time consuming. We develop two general approaches to active task selection, one based on information maximization and the other on model performance, and apply them to the Efficient Lifelong Learning Algorithm (ELLA) (Ruvolo and Eaton 2013). We show that active task selection enables ELLA to achieve large gains in learning efficiency compared to when the task order is outside its control, thereby reducing the number of tasks required to achieve a particular level of performance and improving scalability. We also demonstrate a form of transfer learning in the lifelong learning setting, in which the agent actively seeks knowledge to optimize performance on a specific target task. We show that this targeted active task selection enables ELLA to focus its learning and achieve better performance on the target task than other methods.

Related Work

The model framework used in ELLA is related to a number of current MTL algorithms (Rai and Daumé III 2010; Zhang et al. 2008; Kumar and Daumé III 2012) that represent each individual task's model as a linear combination of a common shared basis $\mathbf{L}$. In most cases, the basis is learned in tandem with all task models in an expensive batch training process. Adding even a single task may require re-optimization of all models, making these batch MTL approaches unsuitable for lifelong learning. Saha et al. (2011) proposed an algorithm for online multi-task classification, OMTL, that is based on perceptron learning and supports incremental learning. However, OMTL exhibits worse performance than ELLA, both in terms of accuracy and speed.

Task selection is closely related to both active learning and curriculum learning methods, which order training instances in an attempt to maximize model performance. Ac-

tive learning criteria typically focus on the most uncertain (Tong and Koller 2001; Cohn et al. 1996) or the most informative instances (MacKay 1992). In contrast, curriculum learning (Bengio et al. 2009; Kumar et al. 2010; Lee and Grauman 2011) advocates learning the easiest instances first and progressing toward increasingly harder instances, effectively the opposite of active learning criteria.

Active learning has been evaluated in a number of MTL algorithms (Saha et al. 2010; Zhang 2010; Saha et al. 2011), and has been shown to generally reduce the amount of training data necessary to achieve a particular level of performance. This paper builds on this earlier work by extending these ideas to lifelong learning, and introducing an efficient and highly effective mechanism for selecting the next task to learn: the Diversity heuristic. In contrast to these active MTL methods, which focus primarily on optimizing model performance, we focus on learning the basis $\mathbf{L}$ efficiently, which will provide maximal benefit to learning future tasks.

The Lifelong Learning Problem

This paper uses the following notational conventions: matrices are denoted by bold uppercase letters, vectors are denoted by bold lowercase letters, scalars are denoted by normal lowercase letters, and sets are denoted using script typeface (e.g., $\mathcal{A}$). Quantities related to a particular task are denoted using parenthetical superscripts (e.g., matrix $\mathbf{A}^{(t)}$ and vector $\mathbf{v}^{(t)}$ are each related to task t). We use the shorthand $\mathbf{A}^{(1:t)}$ to refer to the list of variables $\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(t)}$.

We employ a lifelong learning framework in which the agent faces a series of supervised learning tasks $\mathcal{Z}^{(1)}, \mathcal{Z}^{(2)}, \ldots, \mathcal{Z}^{(T_{\max})}$. Each learning task $\mathcal{Z}^{(t)} = \left(\hat{f}^{(t)}, \mathbf{X}^{(t)}, \mathbf{y}^{(t)}\right)$ is specified by a (hidden) function $\hat{f}^{(t)} : \mathcal{X}^{(t)} \mapsto \mathcal{Y}^{(t)}$ from an instance space $\mathcal{X}^{(t)} \subseteq \mathbb{R}^d$ to a set of labels $\mathcal{Y}^{(t)}$ (typically $\mathcal{Y}^{(t)} = \{-1, +1\}$ for classification tasks and $\mathcal{Y}^{(t)} = \mathbb{R}$ for regression tasks). To learn $\hat{f}^{(t)}$, the agent is given n_t training instances $\mathbf{X}^{(t)} \in \mathbb{R}^{d \times n_t}$ with corresponding labels $\mathbf{y}^{(t)} \in {\mathcal{Y}^{(t)}}^{n_t}$ given by $\hat{f}^{(t)}$. For brevity, we use $(\mathbf{x}_i^{(t)}, y_i^{(t)})$ to represent the ith labeled training instance for task t. The agent does not know *a priori* the total number of tasks $T_{\max}$, their order, or their distribution.

Each time step, the agent receives a batch of labeled training data for some task t, either a new task or as additional data for a previous task. Let T denote the number of tasks the agent has encountered so far. At any time, the agent may be asked to make predictions on data from any previous task. Its goal is to construct task models $f^{(1)}, \ldots, f^{(T)}$, where each $f^{(t)} : \mathbb{R}^d \mapsto \mathcal{Y}^{(t)}$, such that each $f^{(t)}$ will approximate $\hat{f}^{(t)}$ to enable the accurate labeling of new data. Additionally, the agent must be able to rapidly update each model $f^{(t)}$ in response to new training data for a previously learned task, and efficiently add $f^{(t)}$'s to model new tasks.

Background on ELLA

This section provides an overview of the Efficient Lifelong Learning Algorithm (ELLA) (Ruvolo and Eaton 2013) that forms the foundation for our proposed enhancements. For further details on ELLA, please see the original paper.

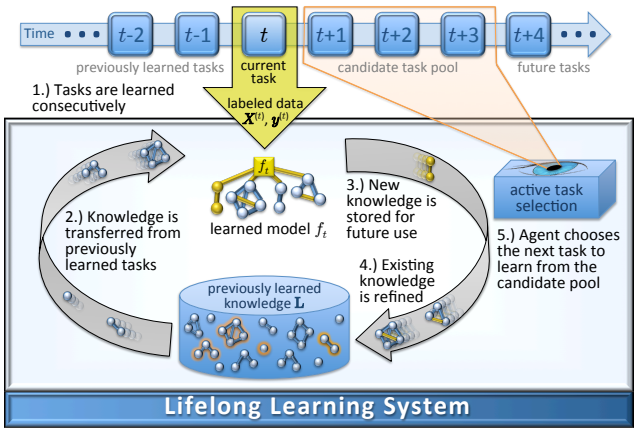


Figure 1: An illustration of the lifelong learning process.

ELLA learns and maintains a repository of k latent model components $\mathbf{L} \in \mathbb{R}^{d \times k}$, which forms a basis for all task models and serves as the mechanism for knowledge transfer between tasks. For each task t, ELLA learns a model $f^{(t)}(\mathbf{x}) = f(\mathbf{x}; \boldsymbol{\theta}^{(t)})$ that is parameterized by a d-dimensional task-specific vector $\boldsymbol{\theta}^{(t)} = \mathbf{L}\mathbf{s}^{(t)}$ represented as a sparse linear combination of the columns of $\mathbf{L}$ using the weight vector $\mathbf{s}^{(t)} \in \mathbb{R}^k$. The specific form of $f^{(t)}(\mathbf{x})$ is dependent on the base learning algorithm, as described below.

Given the labeled training data for each task, ELLA optimizes the predictive loss of each model while encouraging shared structure through $\mathbf{L}$ by minimizing:

$$e_T(\mathbf{L}) = \frac{1}{T}\sum_{t=1}^{T} \min_{\mathbf{s}^{(t)}} \left\{ \frac{1}{n_t}\sum_{i=1}^{n_t} \mathcal{L}\left(f\left(\mathbf{x}_i^{(t)}; \mathbf{L}\mathbf{s}^{(t)}\right), y_i^{(t)}\right) + \mu\|\mathbf{s}^{(t)}\|_1 \right\} + \lambda\|\mathbf{L}\|_{\mathrm{F}}^2 \ , \quad (1)$$

where $\mathcal{L}$ is a known loss function for fitting the model (e.g., squared loss, etc.). This optimization problem is closely related to a number of batch MTL algorithms, such as GO-MTL (Kumar and Daumé III 2012). Equation 1 is not jointly convex in $\mathbf{L}$ and the $\mathbf{s}^{(t)}$'s, and so most batch MTL methods yield a local optimum using an alternating optimization procedure that is prohibitively expensive for lifelong learning.

To enable its efficient solution, ELLA approximates Equation 1 using two simplifications:

1. To eliminate the explicit dependence on all previous training data through the inner summation, ELLA approximates Equation 1 using the second-order Taylor expansion of $\frac{1}{n_t}\sum_{i=1}^{n_t} \mathcal{L}\left(f\left(\mathbf{x}_i^{(t)}; \boldsymbol{\theta}\right), y_i^{(t)}\right)$ around $\boldsymbol{\theta} = \boldsymbol{\theta}^{(t)}$, where $\boldsymbol{\theta}^{(t)} = \arg\min_{\boldsymbol{\theta}} \frac{1}{n_t}\sum_{i=1}^{n_t} \mathcal{L}\left(f\left(\mathbf{x}_i^{(t)}; \boldsymbol{\theta}\right), y_i^{(t)}\right)$ is an optimal predictor learned on only the training data for task t. Crucially, this step removes the optimization's dependence on the number of data instances $n_1 \ldots n_T$.
2. ELLA computes each $\mathbf{s}^{(t)}$ only when training on task t, and does not update them when training on other tasks. This choice improves computational efficiency without significantly affecting performance—as T grows large,

the performance penalty for this simplification becomes vanishingly small (Ruvolo and Eaton 2013).

These simplifications yield the following efficient update equations that approximate the result of Equation 1:

$$\mathbf{s}^{(t)} \leftarrow \arg\min_{\mathbf{s}^{(t)}} \ell\left(\mathbf{L}_m, \mathbf{s}^{(t)}, \boldsymbol{\theta}^{(t)}, \mathbf{D}^{(t)}\right) \quad (2)$$

$$\mathbf{L}_{m+1} \leftarrow \arg\min_{\mathbf{L}} \hat{g}_m(\mathbf{L}) \quad (3)$$

$$\hat{g}_m(\mathbf{L}) = \lambda\|\mathbf{L}\|_{\mathsf{F}}^2 + \frac{1}{T}\sum_{t=1}^{T} \ell\left(\mathbf{L}, \mathbf{s}^{(t)}, \boldsymbol{\theta}^{(t)}, \mathbf{D}^{(t)}\right) \quad (4)$$

where $\ell\left(\mathbf{L}, \mathbf{s}, \boldsymbol{\theta}, \mathbf{D}\right) = \mu\left\|\mathbf{s}\right\|_1 + \left\|\boldsymbol{\theta} - \mathbf{L}\mathbf{s}\right\|_{\mathbf{D}}^2$, $\mathbf{L}_m$ refers to the value of the latent model components at the start of the mth iteration, $\mathbf{D}^{(t)}$ is the Hessian of the loss function $\mathcal{L}$ evaluated at $\boldsymbol{\theta}^{(t)}$ multiplied by $1/2$, and t is the current task.

Each iteration m, ELLA receives training data $(\mathbf{X}^{(t)}, \mathbf{y}^{(t)})$ for some task t. First, ELLA computes an optimal model $\boldsymbol{\theta}^{(t)}$ and Hessian $\mathbf{D}^{(t)}$ from only this training data for task t using a single-task learner:

$$(\boldsymbol{\theta}^{(t)}, \mathbf{D}^{(t)}) = \text{singleTaskLearner}(\mathbf{X}^{(t)}, \mathbf{y}^{(t)}) \ . \quad (5)$$

The form of this step depends on the base learning algorithm; Ruvolo and Eaton (2013) provide details for learning linear and logistic regression models using ELLA. Once the task has been modeled, ELLA then updates the basis $\mathbf{L}$ to incorporate new knowledge by nulling the gradient of Equation 4 and solving for $\mathbf{L}$ via an efficient incremental update.

Active Curriculum Selection

So far, we have framed lifelong learning as a passive process, i.e., one in which the agent has no control over the order in which learning tasks are presented. We now consider a lifelong learning agent that may *actively* choose the next task to learn from a set of candidate tasks (see Figure 1).

We formalize the problem of active curriculum selection in the following manner. Consider a learning agent that has access to training data from a pool of unlearned tasks $\mathcal{Z}^{(T+1)}, \ldots, \mathcal{Z}^{(T_{\text{pool}})}$, where $T+1 \leq T_{\text{pool}} \leq T_{\max}$ and only a subset of the training instances for each of these tasks is labeled. Let $\left(\hat{\mathbf{X}}^{(t)}, \hat{\mathbf{y}}^{(t)}\right)$ denote this subset of initially labeled data for task t. Based on these small labeled subsets, the agent must select the index of the next task to learn $t_{\text{next}} \in \{T+1, \ldots, T_{\text{pool}}\}$.[1] The value of T_{pool} could be a fixed value or it could be set dynamically during learning (e.g., we could set $T_{\text{pool}} = T+z$ so that the agent has knowledge of the next z upcoming tasks). Once the agent decides to learn a particular task, all of the training labels $\mathbf{y}^{(t_{\text{next}})}$ are revealed to the agent, allowing it to model the new task.

In this section, we develop task selection methods that allow agents to maximize their ability to learn across a wide range of future tasks. Therefore, we will focus on encouraging the repository of latent components to be able to flexibly and rapidly adapt to any future task the agent might be required to solve. Later we will develop an approach that focuses the selection of tasks toward learning a *specific* target task. We consider three approaches for choosing the next task to learn that encourage general knowledge acquisition:

- choose t_{next} to maximize expected information gain about the basis $\mathbf{L}$ (Information Maximization),
- choose t_{next} to minimize the worse case fit of $\mathbf{L}$ to each task in the pool (Diversity methods), and
- choose t_{next} randomly (this corresponds to the original lifelong learning setting in which the curriculum order is not under the learner's control).

Next, we describe both the Information Maximization and Diversity methods for selecting the next task to learn. These methods are inspired by active learning criteria (see Related Work). We also evaluated an approach based on curriculum learning (Bengio et al. 2009) that chooses to learn the task with the best performance given the current $\mathbf{L}$, but found that this method performed significantly worse than random task selection in all cases and so do not consider it further.

Information Maximization

Suppose a learning agent's goal is to infer the value of a random vector $\mathbf{h}$ from noisy observations. For lifelong learning, the random variable $\mathbf{h}$ represents the set of latent model components $\mathbf{L}$ used in ELLA. Let $\mathcal{X}_m$ be a set of random vectors that the learner can choose to observe at iteration m. Here, we use a myopic information maximization criterion (*InfoMax* for short) where the learner chooses to observe the random vector from the set $\mathcal{X}_m$ that gives the most expected information gain about the random vector $\mathbf{h}$:

$$\begin{aligned}\mathbf{x}_{\text{next}} &= \arg\min_{\mathbf{x}\in\mathcal{X}_m} H\left[\mathbf{h}|\mathbf{x}, \mathcal{I}_m\right] \\ &= \arg\min_{\mathbf{x}\in\mathcal{X}_m} \int p(\mathbf{x}=\mathbf{v}|\mathcal{I}_m) H[\mathbf{h}|\mathbf{x}=\mathbf{v}, \mathcal{I}_m]\, d\mathbf{v}\,, \end{aligned} \quad (6)$$

where H is the differential entropy, $\mathcal{I}_m$ represents all information available at iteration m, $\mathbf{x}$ is a random vector, and $\mathbf{x} = \mathbf{v}$ is the event that the random vector $\mathbf{x}$ has value $\mathbf{v}$.

Our goal is to learn a flexible and complete set of latent model components for ELLA, and so we maximize information about the basis $\mathbf{L}$ where the set of random variables that the agent may observe at the mth iteration, $\mathcal{X}_m$, corresponds to the tuples $(\boldsymbol{\theta}^{(t)}, \mathbf{D}^{(t)})$ returned by the single task learner (Equation 5) for $t \in \{T+1, \ldots, T_{\text{pool}}\}$. This formulation yields the following objective function:

$$\begin{aligned} t_{\text{next}} = \arg\min_{t} \iint p(\boldsymbol{\theta}^{(t)}=\mathbf{u}, \mathbf{D}^{(t)}=\mathbf{V}|\mathcal{I}_m) \times \\ H\left[\mathbf{L}|\boldsymbol{\theta}^{(t)}=\mathbf{u}, \mathbf{D}^{(t)}=\mathbf{V}, \mathcal{I}_m\right] d\mathbf{u}\, d\mathbf{V}\,. \end{aligned} \quad (7)$$

Unfortunately, this objective is very computationally expensive in the general case, and so it does not have the efficiency or scalability necessary for lifelong learning in its current form. To overcome this issue, we describe several modifications that allow us to efficiently approximate Equation 7.

The first challenge is that the probability density $p(\boldsymbol{\theta}^{(\mathbf{t})}=\mathbf{u}, \mathbf{D}^{(\mathbf{t})}=\mathbf{V}|\mathcal{I}_m)$ will depend on the particular base learner used by ELLA. To provide a general solution, we approximate the density using a Dirac delta function at $(\hat{\boldsymbol{\theta}}^{(t)}, \hat{\mathbf{D}}^{(t)}) = \text{singleTaskLearner}(\hat{\mathbf{X}}^{(t)}, \hat{\mathbf{y}}^{(t)})$ (that is we use the tuple returned by running the single task learner on the subset of labeled training data for task t).

[1] Note that after the index is chosen, $T \leftarrow T+1$ and the tasks are reordered such that the remaining unlearned tasks in the pool are given as $\mathcal{Z}^{(T+1)}, \ldots, \mathcal{Z}^{(T_{\text{pool}})}$.

Next, we approximate the entropy term using a Laplace approximation of the density of $\mathbf{L}$ by constructing a Gaussian density for $\mathbf{L}$ about $\mathbf{L}^\star$, which is a mode of Equation 4. Specifically, given $\mathbf{L}^\star$, we can approximate the density of $\text{vec}(\mathbf{L})$ as a multivariate Gaussian:

$$\text{E}\left[\text{vec}(\mathbf{L})|\boldsymbol{\theta}^{(t)}=\hat{\boldsymbol{\theta}}^{(t)},\mathbf{D}^{(t)}=\hat{\mathbf{D}}^{(t)},\mathcal{I}_m\right] = \text{vec}(\mathbf{L}^\star)$$

$$\text{Cov}\left[\text{vec}(\mathbf{L})|\boldsymbol{\theta}^{(t)}=\hat{\boldsymbol{\theta}}^{(t)},\mathbf{D}^{(t)}=\hat{\mathbf{D}}^{(t)},\mathcal{I}_m\right] = \frac{1}{2}\left(\lambda\mathbf{I}_{k\times d,k\times d}+\frac{1}{T+1}\left(\frac{1}{\hat{n}_t}\left(\hat{\mathbf{s}}^{(t)}\hat{\mathbf{s}}^{(t)\top}\right)\otimes\hat{\mathbf{D}}^{(t)}+\sum_{t'=1}^{T}\frac{1}{n_{t'}}\left(\mathbf{s}^{(t')}\mathbf{s}^{(t')\top}\right)\otimes\mathbf{D}^{(t')}\right)\right)^{-1}.$$

The previous equation does not incorporate uncertainty over the $\mathbf{s}^{(t)}$'s in computing the entropy. This simplification avoids additional computational overhead, which would preclude the use of this active task selection method for lifelong learning, where efficiency is paramount. We leave further exploration of this simplification and its effect on quality of the selected tasks to future work. To this end we approximate the densities of $\mathbf{s}^{(t)}$ for $t \in \{T+1,\ldots,T_{\text{pool}}\}$ given $\hat{\boldsymbol{\theta}}^{(t)},\hat{\mathbf{D}}^{(t)}$, and $\mathcal{I}_j$ using a Dirac delta function at $\hat{\mathbf{s}}^{(t)} = \arg\min_{\mathbf{s}} \ell(\mathbf{L}_m,\mathbf{s},\hat{\boldsymbol{\theta}}^{(t)},\hat{\mathbf{D}}^{(t)})$. Similarly, the densities for $\mathbf{s}^{(t')}$ with $t' \in \{1,\ldots,T\}$ given $\hat{\boldsymbol{\theta}}^{(t)},\hat{\mathbf{D}}^{(t)}$, and $\mathcal{I}_m$ are approximated using Dirac delta functions at $\mathbf{s}^{(t')} = \arg\min_{\mathbf{s}} \ell(\mathbf{L}_{\text{iter}(t')},\mathbf{s},\boldsymbol{\theta}^{(t')},\mathbf{D}^{(t')})$, where $\text{iter}(t')$ is the iteration that ELLA last received training data for task t'.

To arrive at a final task selection criterion, we combine our approximations with the well-known differential entropy $H[\mathbf{x}] = \frac{1}{2}\ln\left((2\pi e)^d|\boldsymbol{\Sigma}|\right)$ of a multivariate Gaussian-distributed random variable $\mathbf{x} \in \mathbb{R}^d$ with covariance $\boldsymbol{\Sigma}$, yielding the following InfoMax task selection rule:

$$t_{\text{next}} = \underset{t\in\{T+1,\ldots,T_{\text{pool}}\}}{\arg\min} \ln\left|\boldsymbol{\Sigma}^{(t)}\right| , \tag{8}$$

where

$$\boldsymbol{\Sigma}^{(t)} = \text{Cov}\left[\text{vec}(\mathbf{L})|\boldsymbol{\theta}^{(t)}=\hat{\boldsymbol{\theta}}^{(t)},\mathbf{D}^{(t)}=\hat{\mathbf{D}}^{(t)},\mathcal{I}_m\right] .$$

Diversity Methods

The previous section approached task selection from an information theoretic perspective. In this section, we propose a simple and efficient *Diversity heuristic* that encourages $\mathbf{L}$ to serve as an effective basis for a wide variety of tasks, enabling ELLA to be able to rapidly learn any new task. To encourage ELLA to be able to solve the widest range of tasks, we select the next task as the one that the current basis $\mathbf{L}$ is doing the *worst* job solving, relative to how well that particular task could be solved using a single task learning model:

$$t_{\text{next}} = \underset{t\in\{T+1,\ldots,T_{\text{pool}}\}}{\arg\max} \min_{\mathbf{s}} \ell\left(\mathbf{L}_m,\mathbf{s},\hat{\boldsymbol{\theta}}^{(t)},\hat{\mathbf{D}}^{(t)}\right) , \tag{9}$$

where, as stated in the previous section, $(\hat{\boldsymbol{\theta}}^{(t)},\hat{\mathbf{D}}^{(t)}) = \text{singleTaskLearner}(\hat{\mathbf{X}}^{(t)},\hat{\mathbf{y}}^{(t)}))$. Effectively, Equation 9 focuses on selecting tasks that are encoded poorly with the current basis set (i.e., tasks that are unlike anything that the learner has seen so far), thereby encouraging diversity.

The Diversity heuristic has connections to the work of Arthur and Vassilvitskii (2007) on choosing initial centroids for k-means. Their algorithm, k-means++, stochastically chooses to add a point to a set of initial centroids with probability proportional to the squared distance between the candidate point and the closest centroid. This selection rule encourages points to be selected as centroids that are poorly represented by the current set. This strategy is analogous to the Diversity heuristic in the sense that we measure squared distance based on the error of the current basis on coding the candidate task model. Inspired by k-means++, we test a stochastic version of the Diversity heuristic that we call *Diversity++*, in which the probability of selecting task t is:

$$p(t_{\text{next}}=t) \propto \left(\min_{\mathbf{s}} \ell\left(\mathbf{L}_m,\mathbf{s},\hat{\boldsymbol{\theta}}^{(t)},\hat{\mathbf{D}}^{(t)}\right)\right)^2 . \tag{10}$$

Note that Diversity is equivalent to $\arg_t \max p(t_{\text{next}}=t)$.

Computational Complexity

All of the active curriculum selection methods first estimate $\mathbf{s}^{(t)}$ for each candidate task by invoking a single task learner (with some complexity $O(\xi(d,n_t))$) and solving a Lasso problem (which takes $O(d^3+d^2k+dk^2)$). For the Diversity methods, we next compute the error of the shared basis model fit to the single task model fit (which takes $O(kd+d^2)$), yielding a total complexity of $O(d^3+d^2k+dk^2+\xi(n_t,d))$ per candidate task. In contrast, the InfoMax method requires inverting a $d\times k$ by $d\times k$ matrix, which can be done in $O(d^3k^2)$ (see Ruvolo and Eaton (2013)), for a total complexity of $O(d^3k^2+\xi(d,n_t))$ per candidate task.

Targeted Curriculum Selection

So far, we have focused on active task selection to support the learning of an unknown set of future tasks. In this section, we explore a form of transfer learning (Pan and Yang 2010) in which the lifelong learner focuses on acquiring knowledge to support a specific *target task*, for which it has access to training data $\left(\mathbf{X}^{(\text{target})},\mathbf{y}^{(\text{target})}\right)$. Its goal is to select learning tasks in order to quickly and effectively optimize those model components needed for the target task, thereby maximizing its performance on the target task (rather than on any arbitrary future task as before).

We derive the Targeted InfoMax method in a nearly identical fashion to general InfoMax, with the exception that instead of setting $\mathbf{h} = \mathbf{L}$, we let $\mathbf{h} = \mathbf{L}\boldsymbol{s}^{(\text{target})}$, where $\mathbf{s}^{(\text{target})}$ is learned each iteration m via Equations 2 and 5. This formulation corresponds to maximizing the information about the model parameters for the target task. The objective for selecting the next task becomes:

$$t_{\text{next}} = \arg\min_t \iint p(\boldsymbol{\theta}^{(t)}=\mathbf{u},\mathbf{D}^{(t)}=\mathbf{V}|\mathcal{I}_m)\times H\left[\mathbf{L}\mathbf{s}^{(\text{target})}|\boldsymbol{\theta}^{(t)}=\mathbf{u},\mathbf{D}^{(t)}=\mathbf{V},\mathcal{I}_m\right]d\mathbf{u}\,d\mathbf{V} . \tag{11}$$

Using similar approximations as for Equation 7, we arrive at the following rule for Targeted InfoMax task selection:

$$t_{\text{next}} = \underset{t\in\{T+1,\ldots,T_{\text{pool}}\}}{\arg\min} \ln\left|\boldsymbol{\Psi}^\top\boldsymbol{\Sigma}^{(t)}\boldsymbol{\Psi}\right| , \tag{12}$$

where $\boldsymbol{\Psi} = \mathbf{s}^{(\text{target})}\otimes\mathbf{I}_d$.

Evaluation

We evaluated the following methods of active task selection against random task selection: InfoMax, Targeted InfoMax, Diversity, and Diversity++. We assessed performance in two problem settings: (1) *general knowledge acquisition*, in which performance was evaluated against a pool of unknown tasks, and (2) *targeted knowledge acquisition* in which performance was evaluated against a known target task. Each method of task selection was tested on four data sets:

Synthetic Regression Tasks We created a set of $T_{\max} = 100$ random tasks with $d = 13$ features and $n_t = 100$ instances per task. The task parameters $\boldsymbol{\theta}^{(t)}$ were generated using a linear combination of $k = 6$ random latent components in $\mathbb{R}^{12}$. The $\mathbf{s}^{(t)}$ vectors had a sparsity of 0.5 (i.e., each $\boldsymbol{\theta}^{(t)}$ was constructed using only three latent components). Each instance in $\mathbf{X}^{(t)}$ was generated from a standard normal distribution, with labels $\mathbf{y}^{(t)} = {\mathbf{X}^{(t)}}^{\top}\boldsymbol{\theta}^{(t)} + \epsilon$, where each element of ϵ is independent univariate Gaussian noise. A bias term was added as the 13th feature prior to learning.

London School Data The London Schools data set consists of exam scores from 15,362 students in 139 schools. We treat the data from each school as a separate task. The goal is to predict the exam score of each student. We use the same feature encoding as used by Kumar and Daumé III (2012), where four school-specific and three student-specific categorical variables are encoded as a collection of binary features. We use the exam year and a bias term as additional features, giving each data instance $d = 27$ features.

Land Mine Detection In the Land Mine data set (Xue et al. 2007), the goal is to detect whether or not a land mine is present in an area based on radar images. The 10 input features (plus a bias term) are extracted from radar data; see Xue et al. (2007) for details. The data set consists of 14,820 data instances in total, divided into 29 different geographical regions. We treat each region as a different task.

Facial Expression Recognition This data set is from a recent facial expression recognition challenge (Valstar et al. 2011). Each task involves recognizing one of three different facial action units (#5: upper lid raiser, #10: upper lip raiser, and #12: lip corner pull) from an image of one of seven subjects' faces. There are a total of 21 tasks, each with 450–999 images. We use the same feature encoding as Ruvolo and Eaton (2013), using a multi-scale Gabor pyramid to extract 2,880 Gabor features for each image, then reducing them to 100 dimensions using PCA, and adding a bias term.

Evaluation Procedure

For each task, we created random $50\%/50\%$ splits between training and hold-out test data. Additionally, we divided the set of tasks into two subsets: one set of *training tasks* that serves as the pool for active task selection and is used to learn $\mathbf{L}$, and one set of *evaluation tasks* on which we measure the performance of the learned $\mathbf{L}$. Therefore, $T_{\text{pool}} \approx \frac{1}{2}T_{\max}$ in these experiments. The agent never has access to the evaluation tasks, and so must select the curriculum order from the training tasks alone. For each training task, 25% of its training data was revealed as the initial sample of labeled data (recall that each task selection method requires a small amount of labeled data to measure the utility of learning a particular task). Each experiment was repeated 1,000 times to smooth out variability due to the factors discussed above.

The *general knowledge acquisition* experiments followed the procedure below to evaluate each task selection method:

1. Begin with no tasks learned; $\mathbf{L}$ is initialized randomly.
2. Select the next training task to learn using some method.
3. Learn the new task using Equations 2–4 to yield a new $\mathbf{L}$.
4. Compute models for each evaluation task using Equation 2 for the current $\mathbf{L}$ (no modification of $\mathbf{L}$ is allowed).
5. Record the performance of the evaluation task models on the hold-out test data, yielding an estimate of how well the current $\mathbf{L}$ captures knowledge for new tasks.
6. Go to step 2 while additional training tasks remain.

The experiments on *targeted knowledge acquisition* first chose a single target task from the set of evaluation tasks and then followed the same procedure, but instead measured performance only against that target task in Step 5.

The values of the parameters k and a ridge term, Γ, for the single task regressors for ELLA were selected using a grid-search over the values $k \in \{1, \ldots, 10\}$ and $\Gamma \in \{e^{-5}, e^{-3}, e^{-1}, e^{1}\}$ to maximize performance on the evaluation tasks averaged over all of the task selection methods. The value of μ was set to 1 and the value of λ was set to e^{-5}.

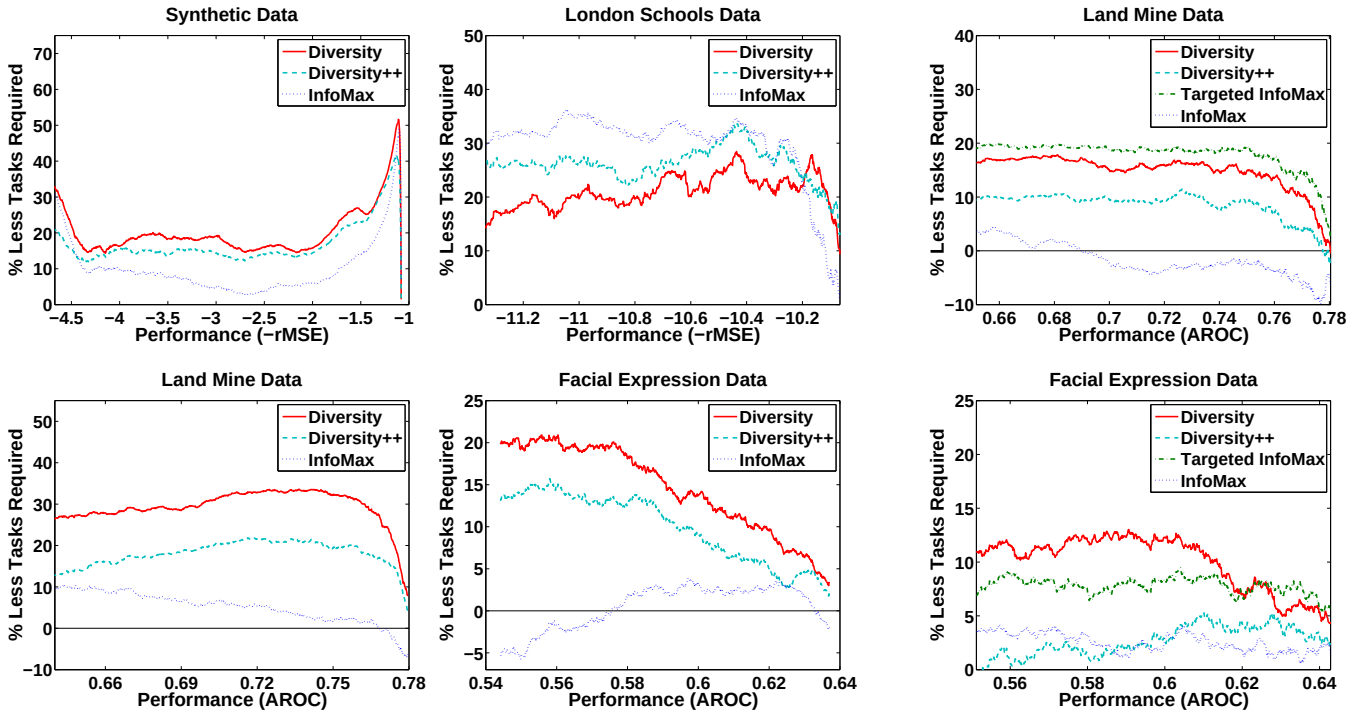
We are primarily concerned with the performance level achieved by each task selection method as a function of the number of tasks learned. Specifically, we measure how many less tasks (expressed as a percentage) are required for active task selection to achieve and maintain a particular performance level as compared to the number of tasks required for random task selection to achieve and maintain that same performance level.[2] We measured performance using the area under the ROC curve for classification tasks (chosen due to the highly skewed class distributions for the data sets) and -rMSE (negative root mean squared error) for regression tasks. A score of 0 on *% Less Tasks Required* indicates that the active task selection method has equal efficiency to random selection, a negative score indicates that it is *less* efficient, and a positive score indicates that it is *more* efficient.

Results

Figure 2a and Table 1a depict our results on general knowledge acquisition, showing that active task selection is almost always more efficient than random selection. Most encouragingly, Diversity achieves large gains in efficiency over random selection on *all* data sets. In some cases, Diversity achieves equivalent performance to random selection using approximately one-half the number of tasks.

InfoMax also shows strong gains in efficiency over random selection. However, the results for this method are not as consistent as those for Diversity, since we occasionally observe inefficient task selection (e.g., on the two classification data sets). However, InfoMax achieves the best efficiency of all approaches on the London Schools data set.

[2] If a particular run did not achieve a specific level of performance, then the number of tasks required to achieve that level of performance was set to one plus the total number of training tasks.



(a) Active Task Selection for General Knowledge Acquisition

(b) Targeted Knowledge Acquisition

Figure 2: The results of active task selection for (a) general, and (b) targeted knowledge acquisition. Each plot shows the accuracy achieved by each method versus the relative efficiency (in number of tasks) as compared to random task selection.

Table 1: The results for (a) general and (b) targeted knowledge acquisition, as measured by the percent less tasks required by active task selection, averaged across all performance levels. The mean and standard deviation are reported.

(a) General Knowledge Acquisition

Data Set	InfoMax	Diversity	Diversity++
Land Mine	5.1±3.7	**29.4±4.1**	18.1±3.0
Facial Expr.	0.5±2.6	**14.6±5.1**	9.9±4.0
Syn. Data	10.2±7.9	**20.2±6.7**	17.0±5.9
London Sch.	**29.8±6.8**	21.0±3.1	26.2±3.1

(b) Targeted Knowledge Acquisition

Data Set	Targeted InfoMax	InfoMax	Diversity	Diversity++
Land Mine	**17.9±2.7**	-1.7±3.0	14.9±3.2	8.5±2.5
Facial Expr.	7.8±0.7	2.6±0.8	**10.0±2.5**	2.7±1.3
Syn. Data	**38.4±7.5**	11.4±5.6	19.9±4.9	16.6±5.0
London Sch.	**26.9±1.8**	20.1±2.8	22.3±1.1	16.4±2.7

Recall that Diversity++ is a stochastic version of Diversity, and therefore does not always select the task with the worst performance to learn next. The results show that Diversity++ was dominated in three of the four data sets by Diversity, affirming the benefits of selecting the task with the worst performance to learn next. However, both Diversity++ and InfoMax perform better than Diversity on the London Schools data set. Further investigation is needed to determine whether particular characteristics of the data set, such as the degree of sparsity of its input features, are important for determining when each method should be used.

Table 1b shows our results on *targeted* knowledge acquisition, with Figure 2b depicting extended results for selected data sets. These results show that targeted task selection is highly effective, revealing that Targeted InfoMax is the most efficient method for three out of the four datasets. Therefore, it appears that incorporating knowledge of the target task into InfoMax can lead to large gains in performance, not only over general InfoMax but also over Diversity, the best general task selection method.

Conclusion

We have considered the setting of active curriculum selection in which a lifelong learner can select which task to learn next for either general and targeted knowledge acquisition. Our proposed Diversity heuristic is efficient and effective for general knowledge acquisition, achieving significant reductions in the number of tasks required to obtained a particular performance level as compared to random selection. Although InfoMax did not work as well as Diversity for general knowledge acquisition, it achieved the best performance on targeted knowledge acquisition for modeling a specific target task. In future work, we intend to connect active task selection to higher level learning goals and incorporate external guidance from a teacher.

Acknowledgements

This research was supported by ONR grant N00014-11-1-0139. We thank Terran Lane, Diane Oyen, and the anonymous reviewers for their helpful feedback on this paper.

References

Arthur, D., and Vassilvitskii, S. 2007. k-means++: The advantages of careful seeding. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1027–1035. SIAM.

Bengio, Y.; Louradour, J.; Collobert, R.; and Weston, J. 2009. Curriculum learning. In *Proceedings of the 26th International Conference on Machine Learning*, 41–48. ACM Press.

Cohn, D. A.; Ghahramani, Z.; and Jordan, M. I. 1996. Active learning with statistical models. *Journal of Artificial Intelligence Research* 4:129–145.

Kumar, A., and Daumé III, H. 2012. Learning task grouping and overlap in multi-task learning. In *Proceedings of the 29th International Conference on Machine Learning*, 1383–1390. Omnipress.

Kumar, M. P.; Packer, B.; and Koller, D. 2010. Self-paced learning for latent variable models. In *Advances in Neural Information Processing Systems 23*, 1189–1197.

Lee, Y. J., and Grauman, K. 2011. Learning the easy things first: Self-paced visual category discovery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1721–1728. IEEE Press.

MacKay, D. J. 1992. Information-based objective functions for active data selection. *Neural Computation* 4:590–604.

Pan, S., and Yang, Q. 2010. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering* 22(10):1345–1359.

Rai, P., and Daumé III, H. 2010. Infinite predictor subspace models for multitask learning. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, 613–620.

Ruvolo, P., and Eaton, E. 2013. ELLA: An efficient lifelong learning algorithm. In *Proceedings of the 30th International Conference on Machine Learning*.

Saha, A.; Rai, P.; Daumé III, H.; and Venkatasubramanian, S. 2010. Active online multitask learning. In *ICML 2010 Workshop on Budget Learning*.

Saha, A.; Rai, P.; Daumé III, H.; and Venkatasubramanian, S. 2011. Online learning of multiple tasks and their relationships. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, 643–651.

Tong, S., and Koller, D. 2001. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research* 2:45–66.

Valstar, M.; Jiang, B.; Mehu, M.; Pantic, M.; and Scherer, K. 2011. The first facial expression recognition and analysis challenge. In *Proceedings of the IEEE International Conference on Automatic Face & Gesture Recognition*, 921–926. IEEE Press.

Xue, Y.; Liao, X.; Carin, L.; and Krishnapuram, B. 2007. Multi-task learning for classification with Dirichlet process priors. *The Journal of Machine Learning Research* 8:35–63.

Zhang, J.; Ghahramani, Z.; and Yang, Y. 2008. Flexible latent variable models for multi-task learning. *Machine Learning* 73(3):221–242.

Zhang, Y. 2010. Multi-task active learning with output constraints. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*.