Representation as a Service: Discriminative Representations for Transfer Learning

Ouais Alsharif
McGill University
Montreal, QC, Canada
ouais.alsharif@gmail.com

Philip Bachman
McGill University
Montreal, QC, Canada
phil.bachman@gmail.com

Joelle Pineau
McGill University
Montreal, QC, Canada
jpineau@cs.mcgill.ca

Abstract

Consider a Machine Learning Service Provider (MLSP) designed to rapidly create highly accurate learners for a never-ending stream of new tasks. The challenge is to produce task-specific learners that can be trained from few labeled samples, even if tasks are not uniquely identified, and the number of tasks and input dimensionality are large. In this paper, we argue that the MLSP should exploit knowledge from previous tasks to build a good representation of the environment it is in, and more precisely, that useful representations for such a service are ones that minimize generalization error for a new hypothesis trained on a new task. We formalize this intuition with a novel method that minimizes an empirical proxy of the intra-task small-sample generalization error. We present several empirical results showing state-of-the art performance on single-task transfer, multitask learning, and the full lifelong learning problem.

1 Introduction

Consider a Machine Learning Service Provider (MLSP) designed to rapidly create highly accurate learners for a never-ending stream of new tasks. Different clients, for example a social networking site or video surveillance company, could ask the MLSP to design a stream of different face recognition agents, each achieving recognition of a different set of target individuals. In such a setting, it is necessary to quickly produce a task-specific learner that can be trained from very few labeled samples (e.g. examples of the target face). Learning from few labeled samples has been known to arise in many tasks for which data is expensive to label (e.g., medical images) or slow to collect (e.g., human computer interaction). In general, there are three paths towards acceptable performance for learning from few samples:

1. Using domain knowledge.
2. Using unlabeled samples.
3. Using labeled samples from a different, but related task.

Bayesian methods [1] have typically focused on the first option, using knowledge of structure in the target task to bias search towards better hypotheses. Other methods, like manifold regularization [2], blend the first and second options by combining domain knowledge, through an engineered distance metric/kernel, and unlabeled samples. Meanwhile, the representation learning community has pursued the second option, producing a number of methods like Deep Boltzmann Machines

[3], Stacked Denoising Autoencoders [4, 5], and Sparse Coding [6] that have proven effective for transfer from unlabeled data to tasks with few labeled samples.

While methods based on options (1) and (2) work well for a variety of tasks, they typically ignore the existence of large amounts of labeled data from different, but possibly related tasks, that may provide significant information about the task-of-interest (called *target task*). The transfer learning community has explored this idea through a variety of methods, often termed "supervised transfer". The most common tools for supervised transfer come from multitask learning [7], in which a fixed set of tasks is given a priori and the learner seeks a model that can *generalize well to new samples* from the given tasks. While multitask learning has been effective in many situations, it falls short in environments where new tasks are constantly arriving and one seeks to *generalize well to new tasks*, as is the case for our MLSP. This latter type of transfer is generally referred to as inductive bias learning [8], lifelong learning [9], learning to learn [10], or never ending learning [11], among other names. Our work focuses on this setting.

In this paper, we present a method to learn in an environment of streaming tasks, like that faced by our MLSP. Our method operates on two components: a parametric representation (shared between tasks) and a collection of parametric function approximators (one per task). We aim to learn the parameters of the representation such that its output would be a more effective input to new function approximators. Our method, called **LeaDR** (Learning Discriminative Representations), builds on the intuition that a good representation is one that allows transfer to new tasks with few labelled samples. We formalize this intuition through a novel objective function that minimizes an empirical proxy of the intra-task small-sample generalization error. This particular objective proves useful in forcing the representation to focus on small sample transfer to new tasks.

In developing our method, we tackle several challenges. First, since tasks faced by our MLSP are not fixed, but rather dynamically defined implicitly through their labellings, new tasks cannot be mapped to previously seen tasks. Moreover, label information will be inconsistent among clients (i.e. one client's person of interest $\neq$ another client's person of interest). This aspect precludes transfer of label information among tasks. We will refer to this problem as *task correspondence*. Another issue is scalability. Since we assume our learner operates in an environment where the number of streaming tasks is very large, memory and computational requirements should be sub-linear in the number of previously-presented tasks. Finally, as many problems of interest are inherently high dimensional, our learner must be efficient when dealing with high dimensional inputs. While some work has sought to address problems similar to our described MLSP [12, 13], these algorithms do not scale well with respect to the number of tasks and/or input dimensionality, and thus experimental results have been limited to low-dimensional multitask problems.

While our primary goal is to tackle the MLSP problem outlined above, our approach proves effective in a wider range of tasks. In our experimental results, we show that LeaDR can be used in three settings: (1) to learn a good representation for achieving single task transfer (Sec. 5.1), where we outperform a state-of-the-art deep learner (Spike-and-Slab Sparse Coding) using only half the labelled samples on the target task, (2) to tackle standard multitask learning (Sec. 5.2), where we match or exceed performance of state-of-the-art multitask learning approaches despite not using task correspondence information, and (3) to solve the MLSP problem in high-dimensional input spaces (Sec. 5.3), with better scalability than previous lifelong learning approaches.

2 A Machine Learning Service Provider: Problem Definition

Let our MLSP operate in an environment $\mathcal{E} = (\mathcal{Q}, \mathcal{X})$, with an input domain $\mathcal{X}$ and a task distribution $\mathcal{Q}$. $\mathcal{Q}$ is a distribution over tasks $\mathcal{T}_i$, where each task $\mathcal{T}_i = (\mathcal{Y}_i, \mathcal{D}_i, \mathcal{L}_i, \mathcal{G}_i)$ comprises:

1. An output domain $\mathcal{Y}_i$.
2. A distribution $\mathcal{D}_i$ over $\mathcal{X} \times \mathcal{Y}_i$.
3. A non-negative loss function $\mathcal{L}_i : \mathcal{Y}_i \times \mathcal{Y}_i \rightarrow \mathbb{R}^+$.
4. A generalization functional $\mathcal{G}_i(h) = \mathbb{E}_{(x,y)\sim\mathcal{D}_i}\mathcal{L}_i(h(x), y)$.

For the environments we consider, $\mathcal{X} = \mathbb{R}^d$. For classification tasks, $\mathcal{Y}_i$ is the discrete space of relevant labels and for regression tasks, $\mathcal{Y}_i = \mathbb{R}$. The generalization functional $\mathcal{G}_i(h)$ measures the

true loss of the learned hypothesis h. For classification tasks, $\mathcal{G}_i(h)$ is the expected misclassification rate of the classifier h w.r.t. $\mathcal{D}_i$. For regression tasks, $\mathcal{G}_i(h)$ is the expected error of the regressor h, w.r.t. $\mathcal{D}_i$. We do not restrict our definitions to specific loss functions as our model is able to accommodate different loss functions (e.g., logistic, ranking, RMSE, etc).

Loosely speaking, our MLSP is a persistent machine learning agent faced with an environment $\mathcal{E}$ and tasked with producing hypotheses $\hat{h}_i : \mathcal{X} \rightarrow \mathcal{Y}_i$ for any tasks $\mathcal{T}_i$ it encounters, so as to minimize $\mathbb{E}_{\mathcal{T}_i \sim \mathcal{Q}} \mathcal{G}_i(\hat{h}_i)$. To produce a hypothesis $\hat{h}_i$ for task $\mathcal{T}_i$, the agent first receives an m-sample $(X_i, Y_i) = \{(x_j, y_j)\}_{j=1}^m$ of training observations drawn from $\mathcal{D}_i$. The agent then applies an algorithm $\mathcal{A}_i$ to the m-sample to produce a hypothesis $\hat{h}_i$ that minimizes a structured risk. We denote this process as $\hat{h}_i = \mathcal{A}_i(X_i, Y_i)$, with:

$$\mathcal{A}_i(X_i, Y_i) = \underset{h \in \mathcal{H}_i^{\mathcal{A}}}{\arg\min} \frac{1}{m} \sum_{(x_j, y_j) \in (X_i, Y_i)} \mathcal{L}_i(h(x_j), y_j) + \mathcal{R}_i(h),$$

where $\mathcal{H}_i^{\mathcal{A}}$ gives the hypothesis space searched by $\mathcal{A}_i$, $\mathcal{L}_i$ gives the loss function minimized by $\mathcal{A}_i$, and $\mathcal{R}_i$ gives any regularization terms used by $\mathcal{A}_i$ to bias the search over $\mathcal{H}_i^{\mathcal{A}}$. We let $\mathcal{A}_i$ minimize a surrogate loss $\mathcal{L}_i^{\mathcal{A}}$, as the natural task loss $\mathcal{L}_i$ may provide an intractable optimization objective.

Using the definitions presented thus far, our MLSP learning objective can be written as:

$$\underset{\theta \in \Theta}{\text{minimize}}\, \mathbb{E}_{\mathcal{T}_i \sim \mathcal{Q}} \mathbb{E}_{(X_i, Y_i) \sim \mathcal{D}_i} \mathcal{G}_i(\mathcal{A}_i(X_i, Y_i)).^1 \tag{1}$$

The objective in (1) measures the ability of the per-task algorithms $\mathcal{A}_i$ to find hypotheses $\hat{h}_i = \mathcal{A}_i(X_i, Y_i)$ that generalize well w.r.t. $\mathcal{G}_i$, in a small-sample setting. As outlined in the next section, in our framework, $\mathcal{A}_i$ is in fact structurally-biased by a common representation learned over all tasks.

3 A Method for Learning Discriminative Representations (LeaDR)

We now present a method for addressing the objective in (1). The basic idea is to combine a single parametric feature extractor f (shared between tasks), with an unbounded collection of trainable function approximators $\{.., , h_i, ...\}$ (one per task). Our approach aims to make the output of f more effective as input to the algorithm $\mathcal{A}_i$ when training h_i for task $\mathcal{T}_i$. While the collection of function approximators may be unbounded, we do select each algorithm/approximator pair $\mathcal{A}_i / h_i$ from a finite set of methods, e.g., linear logistic regression for classification tasks, linear least-squares regression for regression tasks[2].

Using the notation defined in Section 2, our objective is given by:

$$\underset{\theta \in \Theta}{\text{minimize}}\, \mathbb{E}_{\mathcal{T}_i \sim \mathcal{Q}} \mathbb{E}_{(X_i, Y_i) \sim \mathcal{D}_i} \mathcal{G}_i \left(\mathcal{A}_i(f_\theta(X_i), Y_i) \right), \tag{2}$$

where f_θ represents setting the parameters of the parametric function approximator f to θ. This objective is an instance of the objective in (1) in which structure sharing among the per-task algorithms—i.e., the $\mathcal{A}_i$ in (1)—is accomplished by using the same parameterized feature extractor f_θ to preprocess inputs to each $\mathcal{A}_i$.

The particular novelty in our method is that we explicitly minimize an empirical proxy for the expected per-task small-sample generalization errors given by $\mathcal{G}_i \left(\mathcal{A}_i(f(X_i), Y_i) \right)$. In contrast, typical approaches to multitask learning simultaneously learn a feature extractor f_θ and a collection of task-adapted functions $\hat{h}_i$ for a fixed set of tasks $\{\mathcal{T}_1, ..., \mathcal{T}_n\}$. This approach seeks an f such that there exist functions $\hat{h}_i$ with small error, w.r.t. $\mathcal{L}_i$, on the training sets available for each $\mathcal{T}_i$.

Given T sets of m-samples $\{(X_1, Y_1), ..., (X_T, Y_T)\}$ drawn from the environment, our method for Learning Discriminative Representations (called LeaDR) optimizes the following empirical approximation of (2):

$$\underset{\theta \in \Theta}{\text{minimize}}\, \frac{1}{T} \sum_{i=1}^{T} \hat{\mathcal{G}}_i(\mathcal{A}_i, (f_\theta(X_i), Y_i)), \tag{3}$$

[1] While we use m for the number of samples used as input to each $\mathcal{A}_i$, in practice m may differ across tasks.

[2] While we focus in this paper on parametric function approximators, possible extensions to non-parametric function approximators as in [**?**] are also possible.

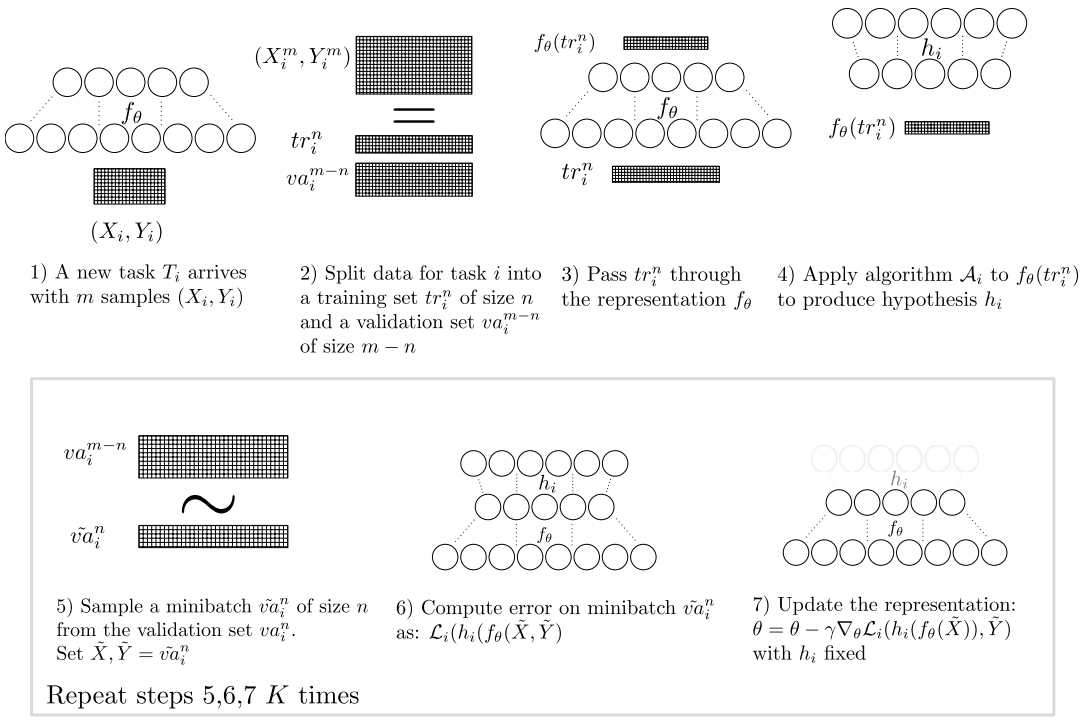


Figure 1: Key Steps for Learning Discriminative Representations (LeaDR) for Transfer

where $\hat{\mathcal{G}}_i$ is an empirical estimate of generalization functional on task i. To define $\hat{\mathcal{G}}_i$, we first define a process for sampling a pseudo-training/validation set pair (tr_i^n, va_i^{m-n}) from the m-sample (X_i, Y_i) available for $\mathcal{T}_i$. We sample the pseudo-train/validate split (tr_i^n, va_i^{m-n}) by randomly sampling $n < m$ observations (x_i, y_i) for tr_i^n, from (X_i, Y_i), and then let va_i^{m-n} contain the remaining $m - n$ observations in (X_i, Y_i). Now, we compute $\hat{\mathcal{G}}_i$ as follows:

$$\hat{\mathcal{G}}_i(\mathcal{A}_i, (f_\theta(X_i, Y_i))) = \mathbb{E}_{(tr_i^n, va_i) \sim (X_i, Y_i)} \left[\sum_{(x_j, y_j) \in va_i} \mathcal{L}_i(\hat{h}_i^n(f_\theta(x_j)), y_j) \right],$$

in which $\hat{h}_i^n = \mathcal{A}_i(f_\theta(tr_i^n))$ indicates the hypothesis produced by applying algorithm $\mathcal{A}_i$ to the pseudo-training set tr_i^n. While tr_i^n is used to train the task-specific hypothesis, va_i^n is used to train the intra-task representation f_θ.

Algorithm 1 describes concretely the steps of our algorithm. Figure 1 presents a visual depiction of each step to further elucidate the algorithm. Note that in the loop on line 6, after altering the representation once, the optimal function approximator given the representation changes. For computational reasons, we approximate the new optimal function with the one acquired before updating the representation.

Our method has the following key properties:

1. **Scalability**: LeaDR is scalable with respect to the number of tasks as computation per-task and memory requirements for all tasks are both $O(1)$. It is also scalable with respect to the dimensionality of the input space and the extracted feature space.
2. **Flexibility**: LeaDR is modular, and is essentially a meta-algorithm that can easily accommodating different feature extractors, f_θ (e.g., boosted stumps, convolutional nets), as well as different function approximators, $\hat{h}_i$ (e.g., regressors, classifiers).
3. **Streaming**: Tasks can be presented to our method sequentially, and need not be remembered explicitly (only through the parameters of the shared representation).

Algorithm 1 Learning Discriminative Representations (LeaDR) for Transfer

Inputs: Source environment $\mathcal{E} = (\mathcal{X}, \mathcal{Q})$, pseudo-training batch size n, updates per task sample K, representation learner f_θ with initial parameters $\theta = \theta_0$, learning rate γ.
while $\mathcal{T}_i \sim \mathcal{Q}$ is requested **do**
  Observe m training examples, $(X_i, Y_i) \sim \mathcal{D}_i$
  Sample train/validate split $(tr_i^n, va_i^{m-n}) \sim (X_i, Y_i)$
  Create an appropriate algorithm/approximator pair $\mathcal{A}_i / h_i$
  Apply $\mathcal{A}_i$ to tr_i^n to get $\hat{h}_i^n$.
  for $k = 1$ **to** K **do**
    Sample a minibatch $\tilde{va}_i^n$ from va_i^n.
    Define $(\tilde{X}, \tilde{Y})$ such that $\tilde{va}_i^n = (\tilde{X}, \tilde{Y})$.
    Let $\theta := \theta - \gamma \nabla_\theta \mathcal{L}_i(\hat{h}_i^n(f_\theta(\tilde{X})), \tilde{Y})$
  end for
end while
Output: Learned representation f_θ.

4 Related Works

Our work relates to several sub-fields. Since our approach focuses primarily on transfer between tasks using labelled data, we focus here on literature related to supervised transfer. For a more extensive survey of transfer learning, see [14]. Ideas grouped under supervised transfer can be categorized into three sub-fields: *multitask learning*, *transductive learning* and *lifelong learning*.

In *multitask learning*, the goal is to solve a number of fixed problems simultaneously, with the hope that by sharing information between tasks, we can achieve better solutions for the problems considered [7]. Many structures have been investigated to share information across tasks, including neural networks [15, 7], distance metrics [10], Bayesian priors [16], sparse code dictionaries [17], and others. Overall, multitask learning assumes the marginal distributions of samples seen during testing to be the same as those during training, and that with each sample, a specific task id is given. The Online MultiTask Learning (OMTL) framework [12] can be used to tackle a stream of related tasks[3]. However OMTL does not scale to environments with large number of tasks as it requires maintaining a $T \times T$ task-relatedness matrix (T =number of tasks). In *transductive learning* [18, 19], the source and target tasks are also the same, however the marginal distributions differ between the training and testing samples. The focus of transductive learning has been on correcting for this difference, to improve performance in target domains.

Lifelong learning differs from the previous two sub-fields in that target problems are assumed to be distinct from training problems [8, 9, 15]. In general, methods used for lifelong learning can be applied in situations where multitask or transductive learning are used, but not the reverse, since lifelong learning relaxes the task correspondence assumption. Theoretical foundations for lifelong learning were established by Baxter [8], wherein PAC bounds for a setting similar to the one in Sec. 2 were presented. The algorithm in [8] uses a two-part model to optimize a multitask learning objective. This model however, consumes an amount of memory linear in the number of tasks. Baxter's method extends to learning deep representations [20].

The best method we are aware of to tackle lifelong learning over large number of tasks is ELLA [13], which builds on a class of methods in multitask learning where multiple tasks share a common basis. The main idea is to represent the parameters of the predictive function f_w for a new task using a sparse code over dictionary elements, i.e. $f_w(x) = (Ls)^\top x$, where $||s||_1$ is small and $w = Ls$. While ELLA is a lifelong learner, consuming $O(T)$ time and $O(1)$ space for T tasks, it is inefficient in domains with high dimensional input spaces, requiring $O(d^3k^2)$ computation time, where d is the dimensionality of the data and k is the number of bases in the sparse code.

While our MLSP is primarily focused on learning representations for inter-task transfer from a large stream of tasks, the LeaDR approach we propose can also improve transfer to new tasks when learning from just a single encountered task. As such, our work is connected to recent literature in computer vision where deep neural networks are trained on a single highly multiclass task with large

[3]The word "online" in OMTL refers to samples from fixed tasks arriving in an online manner. This is distinct from the stream of *tasks* faced by our MLSP.

amounts of labeled data, after which the features learned by the network are applied to tasks other than the training task, for which few labeled samples are available [21, 22, 23]. Using these deeply and supervisedly learned features has led to a rapid advance in the state-of-the-art for a number of domains [21]. Our method can be seen as a new approach to training such networks, aimed at improving transfer to new tasks with very small training sets.

5 Empirical Evaluation

5.1 Representation Learning for Single Task Transfer

First we consider the single task transfer case, in which we compare the representation learned by LeaDR to that learned by vanilla backpropagation, where the goal is to use that representation to transfer knowledge to a new domain. This experiment targets the NIPS 2011 transfer learning challenge [24] which was proposed during the NIPS workshop on Learning Hierarchical Representations. Data is available as follows: a first dataset of 50,000 $32 \times 32 \times 3$ images labelled into 100 classes is given (namely, the CIFAR-100 dataset [25]), along with a second dataset of 100,000 unlabelled $32 \times 32 \times 3$ images. The target domain is a 10-way classification problem, for which only 120 labelled training examples are given; each of the classes in the target domain is distinct from those seen in the source domain. The goal is to create a method that performs well on the target domain's test set, which consists of 2542 samples. The best previously published result for this challenge was Spike and Slab Sparse Coding (S3C) [26] which ignores the labels on CIFAR-100 and uses it just as unlabelled data. After training, the representation learned by S3C is used to extract features, on which a linear classifier is trained using the 120 samples for the target task.

To instantiate the LeaDR framework, we used a convolutional neural network for the representation, and logistic regression (trained by gradient descent) for the function approximator. A stream of training tasks was simulated by sampling different 20-way classification problems from the CIFAR-100 dataset. After training the representation in this way, we used the learned f_θ to extract features on the target task, after which, we trained $\hat{h}_i$ for the target task using the 120 samples (or less). The test set for the target task (2542 samples) was projected through the learned representation and then through the task-specific hypothesis to evaluate performance. We compared our method to learning a representation using a Convolutional Neural Network trained by Backpropagation on the full CIFAR-100 dataset, where we use the representation in the penultimate layer of the ConvNet to project training data of the new domain into a new subspace on which a logistic regression is trained. It is important to note that during testing, both representations were held fixed. Architecture wise, we used the same convolutional architecture for both methods. Note that both methods ignored the unlabelled data. Time wise, LeaDR took about the same time to converge as did backpropagation. More implementation details are in the supplementary material.

Figure 2 shows the performance of LeaDR compared to the standard ConvNet with Backprop, as the number of samples per class increases. We see that LeaDR has an advantage when new tasks have less than 3 samples per class (30 in total). In the "one shot" case, LeaDR presents a gain of 8% compared to a ConvNet. At 5 labelled samples per class (50 in total), we see that representations trained in a supervised way start to outperform unsupervised pre-training (S3C, which uses all 120 labels for the target task). Surprisingly, for the ConvNet with Backprop, performance decreases when we train on the full 120 samples. This may be due to an uneven distribution of 120 samples among the 10 classes on the test problem. The performance of LeaDR in small sample cases showcases the advantage of minimizing empirical generalization, as opposed to standard loss.

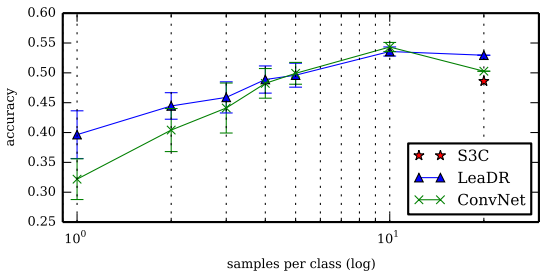


Figure 2: NIPS 2011 transfer learning challenge

5.2 Multitask Learning Experiments

Second we focus on the standard multitask learning case, where training tasks are fixed and testing tasks are constrained to be from the same set of tasks. We show that even though LeaDR was

Table 1: Performance of LeaDR against state-of-the-art multitask learning algorithms. Let d be the dimensionality of the input space, $|T|$ the number of tasks in the dataset, N the number of available examples in the dataset.

Dataset	Error Type	OMTL	GOMTL	ELLA	STL	LeaDR
Landmine						
$d = 9, \|T\| = 29$	AUC	0.63	0.78	0.776	0.76	0.78
London Schools						
$d = 27, \|T\| = 139$	RMSE	N/A	10.10	10.20	11.06	10.08

not designed for this setting, and does not exploit task correspondence, it performs comparably to state-of-the-art multitask learners on two benchmark domains: a classification task, *Landmine* [16], and a regression task, *London schools* [17]. Both datasets have a few dozen tasks, and relatively low-dimensional input space.

We compare our method to three algorithms designed for multitask and lifelong learning: GOMTL [17], OMTL [12] and ELLA [13]. We also include comparison to standard single task learning (STL) as a baseline: logistic regression for the Landmine problem and random forests for the London schools problem. We apply the experimental methodology used in [13], where data for each task (within a domain) is split 50/50 between training/testing sets. We measure error on Landmine (classification) in terms the AUC to be consistent with [13][4]. For the London schools (regression), we measure error in RMSE. More details on datasets, training process and experimental setup are in supplemental material.

Results in Table 1 show that LeaDR performs comparably to other state-of-the-art algorithms on these standard multitask datasets. We note that due to the simplicity of these domains, in terms of task specification and input dimension, most methods perform similarly, and there may not be much room for performance gain relative to single task learning.

5.3 Lifelong Learning Experiments

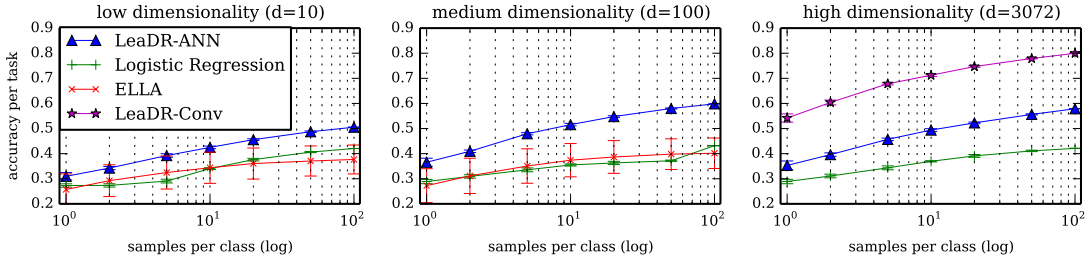
In our final set of experiments, we tackle the full lifelong learning problem, where we investigate how LeaDR performs when presented with a large set of streaming (unknown, non-repeating) tasks with high-dimensional input spaces, as in the MLSP setting. We consider variations on two standard ML domains: the 20 Newsgroups text classification domain [27], and the above-mentioned CIFAR-100 image classification domain [25].

We simulate the online streaming of tasks as follows. We sequentially sample 1000 random (training) 5-way classification problems from the respective set of classes, and apply both stages of LeaDR (representation + fn approx.) to these tasks. We then fix the representation, f_θ, and present 100 new (testing) 5-way classification tasks. We optimize a task-specific hypothesis, $\hat{h}_i$, for each of these using a small set of labelled samples; f_θ is held fixed during testing (so no need for additional va_i samples, Alg.1, line 4). We test the accuracy of each task-specific hypothesis on a test set for that particular task. Samples used for training the representation, training the hypotheses for the test tasks, and for measuring the accuracy on each test task, come from three disjoint sets.

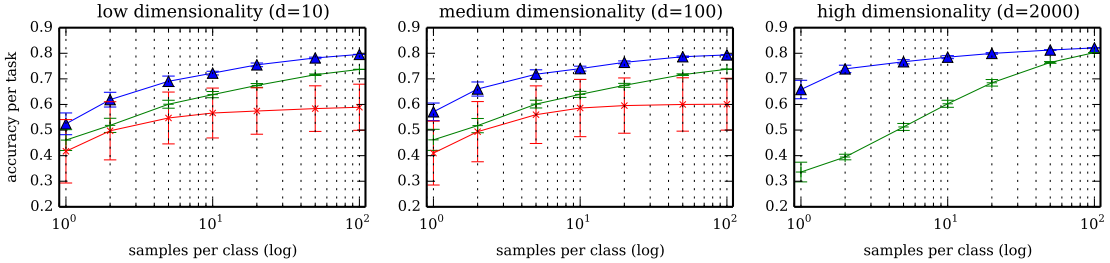
We instantiate LeaDR with a two layer neural net for the representation, and logistic regression for the task-specific function approximators. We compare our method to ELLA [13] and a single-task multinomial logistic regression (training data from all tasks mixed in a single batch); both of which scale independently of the number of tasks. Other algorithms [12, 17] were not included because their per-task cost scales linearly with the number of previously seen tasks[5]. Since ELLA cannot efficiently handle problems with high dimensional input spaces, we consider low and medium dimensional projections of the input spaces, in addition to the (natural) high dimensional input space (we cannot include ELLA in this last case). For 20 Newsgroups, the high dim. input space assumes each document is represented by a vector of counts for the 2000 most-common non-stop words; low (d=10) and medium (d=100) input spaces are created using LDA [28] over these word vectors.

[4] In our experiments, misclassification error was about the same for all the methods, including STL. In our opinion, this suggests the Landmine dataset may not be a great domain for multitask learning.

[5] While we could run other methods on 1000 tasks at the cost of simply waiting longer. We believe this would be misleading, as these methods are inherently not scalable as $T \to \infty$.



(a) Results on CIFAR-100



(b) Results on 20-newsgroups

Figure 3: Results for the lifelong learning experiments. d is the number of dimensions in the input. Each point is averaged over the 100 different test problems, and over 10 different sample sets for each test task.

For CIFAR-100, the high dim. input space uses raw pixel values, whereas the low and medium input spaces are obtained by projecting this down using PCA. We also include results for LeaDR on CIFAR-100 with high dim. inputs where the representation learner is replaced by a ConvNet (rather than a two layer NN).

Figure 3 presents the accuracy of each method as the number of samples for testing tasks increases (we did not find the order of training tasks to have a significant effect.) We observe that LeaDR consistently outperforms other methods for all available sample sizes, and also has lower variance than ELLA. Even single-task regression sometimes outperforms ELLA, usually for tasks where the classes are transpositions of other tasks' classes (e.g, predict (0,1,2) vs. (2,1,0)), as ELLA does not exploit shared features among tasks, but rather approximates the parameter vector directly. LeaDR overcomes this challenge by using the representation to "guide" the function approximator training, and thus can discover and exploit such transpositions.

6 Discussion

In this paper, we formalized an interesting and challenging learning problem, termed the Machine Learning Service Provider, that deals with the problem of rapidly creating accurate learners in environments with streaming (related but non-identical) tasks. We then presented an algorithmic framework (LeaDR) that tackles this problem in a flexible and scalable way. A particular novelty of our method, is to learn the feature representation by minimizing an empirical proxy of the inter-task generalization error. This particular objective proves useful in many scenarios when the goal is to use the learned representation for transfer. Our method presents several desirable properties, including flexibility, scalability. The flexibility of our approach is in its modularity, as both the representation and task-specific function approximators can be changed to suit the input domain and tasks at hand. In terms of scalability, our framework allows scaling multitask and lifelong learning to high dimensional, streaming tasks regimes. Empirically, we verified these claims in three relevant contexts: single-task transfer, where our method was shown to be better at learning convolutional supervised representations than standard backpropagation; Multitask learning: where our algorithm matches our outperforms state-of-the-art multitask learners and finally, lifelong learning, where our algorithm outperforms other methods when facing a large stream of tasks with high dimensional inputs.

References

[1] Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. Computer Vision and Image Understanding (2007)

[2] Belkin, M., Niyogi, P., Sindhwani, V.: Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. The Journal of Machine Learning Research (2006)

[3] Salakhutdinov, R., Hinton, G.E.: Deep boltzmann machines. In: Artificial Intelligence and Statistics (AISTATS). (2009)

[4] Vincent, P., Larochelle, H., Lajoie, I., Benjaminngio, Y., Manzagol, P.A.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. The Journal of Machine Learning Research (2010)

[5] Bengio, Y.: Deep learning of representations for unsupervised and transfer learning. Unsupervised and Transfer Learning Challenges in Machine Learning, Volume 7 (2012)

[6] Raina, R., Battle, A., Lee, H., Packer, B., Ng, A.Y.: Self-taught learning: transfer learning from unlabeled data. In: International Conference on Machine Learning (ICML), ACM (2007)

[7] Caruana, R.: Multitask learning. Machine learning (1997)

[8] Baxter, J.: A model of inductive bias learning. Journal of Artificial Intelligence Research (JAIR) (2000)

[9] Thrun, S.: Lifelong learning: A case study. Technical report (1995)

[10] Thrun, S.: Learning to learn: Introduction. In: In Learning To Learn. (1996)

[11] Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka Jr, E.R., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: AAAI. (2010)

[12] Saha, A., Rai, P., Iii, H.D., Venkatasubramanian, S.: Online learning of multiple tasks and their relationships. In: Artificial Intelligence and Statistics (AISTATS). (2011)

[13] Ruvolo, P., Eaton, E.: Ella: An efficient lifelong learning algorithm. In: International Conference on Machine Learning (ICML). (2013)

[14] Pan, S.J., Yang, Q.: A survey on transfer learning. Knowledge and Data Engineering, IEEE Transactions on (2010)

[15] Baxter, J.: Learning internal representations. In: COLT, ACM (1995)

[16] Xue, Y., Liao, X., Carin, L., Krishnapuram, B.: Multi-task learning for classification with dirichlet process priors. The Journal of Machine Learning Research (2007)

[17] Kumar, A., Daume III, H.: Learning task grouping and overlap in multi-task learning. (2012)

[18] Daumé III, H., Marcu, D.: Domain adaptation for statistical classifiers. (2006)

[19] Zadrozny, B.: Learning and evaluating classifiers under sample selection bias. In: International Conference on Machine Learning (ICML), ACM (2004)

[20] Collobert, R., Weston, J.: A unified architecture for natural language processing: Deep neural networks with multitask learning. In: International Conference on Machine Learning (ICML), ACM (2008)

[21] Razavian, A.S., Azizpour, H., Sullivan, J., Carlsson, S.: Cnn features off-the-shelf: an astounding baseline for recognition. arXiv preprint arXiv:1403.6382 (2014)

[22] Jia, Y.: Caffe: An open source convolutional architecture for fast feature embedding. `http://caffe.berkeleyvision.org/` (2013)

[23] Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: Decaf: A deep convolutional activation feature for generic visual recognition. arXiv preprint arXiv:1310.1531 (2013)

[24] 2011: Nips transfer learning challenge. `https://sites.google.com/site/nips2011workshop/transfer-learning-challenge`

[25] Krizhevsky, A.: Learning multiple layers of features from tiny images. MSc. Thesis (2009)

[26] Goodfellow, I., Courville, A., Bengio, Y.: Large-scale feature learning with spike-and-slab sparse coding. International Conference on Machine Learning (ICML) (2012)

[27] Lang, K.: 20 newsgroups dataset. `http://qwone.com/~jason/20Newsgroups/`

[28] Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. The Journal of Machine Learning Research (2003)