Regret Bounds for Lifelong Learning

Pierre Alquier, The Tien Mai
CREST, ENSAE, Université Paris Saclay

Massimiliano Pontil
Istituto Italiano di Tecnologia
and
University College London

Abstract

We consider the problem of transfer learning in an online setting. Different tasks are presented sequentially and processed by a within-task algorithm. We propose a lifelong learning strategy which refines the underlying data representation used by the within-task algorithm, thereby transferring information from one task to the next. We show that when the within-task algorithm comes with some regret bound, our strategy inherits this good property. Our bounds are in expectation for a general loss function, and uniform for a convex loss. We discuss applications to dictionary learning and finite set of predictors. In the latter case, we improve previous $O(1/\sqrt{m})$ bounds to $O(1/m)$, where m is the per task sample size.

1 INTRODUCTION

Most analyses of learning algorithms assume that the algorithm starts learning from scratch when presented with a new dataset. However, in real life, it is often the case that we will use the same algorithm on many different tasks, and that information should be transferred from one task to another. For example, a key problem in pattern recognition is to learn a dictionary of features helpful for image classification: it makes perfectly sense to assume that features learnt to classify dogs against other animals can be re-used to recognize cats. This idea is at the core of *transfer learning*, see (Thrun and Pratt, 1998; Balcan et al., 2015; Baxter, 1997, 2000; Cavallanti et al., 2010; Maurer, 2005; Maurer et al., 2013; Pentina and Lampert, 2014; Maurer et al., 2016) and references therein.

Proceedings of the 20^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54.

The setting in which the tasks are presented simultaneously is often referred to as *learning-to-learn* (Baxter, 2000), whereas when the tasks are presented sequentially, the term *lifelong learning* is often used (Thrun, 1996). In either case, a huge improvement over "learning in isolation" can be expected, especially when the sample size per task is relatively small. We will use the above terminology in the paper.

Although a substantial amount of work has been done on the theoretical study of learning-to-learn (see Baxter, 2000; Maurer, 2005; Pentina and Lampert, 2014; Maurer et al., 2016, and references therein), we are aware of only few papers which address the statistical analysis of lifelong learning (Balcan et al., 2015; Herbster et al., 2016; Pentina and Urner, 2016). Perhaps most related to our work is (Ruvolo and Eaton, 2013), which studied the convergence of certain optimization algorithms for lifelong learning, however, no statistical guarantees are provided. Furthermore, in all the aforementioned works, the authors propose a technique for transfer learning which constrains the within-task algorithm to be of a certain kind, e.g. empirical risk minimization.

The main goal of this paper is to show that it is possible to perform a theoretical analysis of lifelong learning with minimal assumptions on the form of the within-task algorithm. Given a learner with her/his own favourite algorithm(s) for learning within tasks, we propose a meta-algorithm for transferring information from one task to the next. The algorithm maintains a prior distribution on the set of representations, which is updated after the encounter of each new task using the exponentially weighted aggregation (EWA) procedure, hence we call it *EWA for lifelong learning* or EWA-LL.

A standard way to provide theoretical guarantees for online algorithms are regret bounds, which measure the discrepancy between the prediction error of the forecaster and the error of an ideal predictor. We prove that, as long as the within-task algorithms have good statistical properties, EWA-LL inherits these proper-

ties. Specifically in Theorem 3.1 we present regret bounds for EWA-LL, in which the regret bounds for the within-tasks algorithms are combined into a regret bound for the meta-algorithm.

We also show, using an online-to-batch analysis, that it is possible to derive a strategy for learning-to-learn, and provide risk bounds for this strategy. The bounds are generally in the order of $1/\sqrt{T}+1/\sqrt{m}$, where T is the number of tasks and m is the sample size per task, but we show that in some specific cases the bounds can be improved to the order of $1/\sqrt{T}+1/m$. These rates are novel up to our knowledge and justify the use of transfer learning with very small sample sizes.

The paper is organized as follows. In Section 2 we introduce the lifelong learning problem. In Section 3 we present the EWA-LL algorithm and provide a bound on its expected regret. This bound is very general, but might be uneasy to understand at first sight. So, in Section 4 we present more explicit versions of our bound in two classical examples: finite set of predictors and dictionary learning. We also provide a short simulation study for dictionary learning. At this point, we hope that the reader will have a clear overview of the problem under study. The rest of the paper is devoted to theoretical refinements: in online learning, uniform bounds are the norm rather than bounds in expectations (Cesa-Bianchi and Lugosi, 2006). In Section 5 we establish such bounds for EWA-LL. Section 6 provides an online-to-batch analysis that allows one to use a modification of EWA-LL for learning-to-learn. The supplementary material include proofs (Appendix A), improvements for dictionary learning (Appendix B) and extended results (Appendix C).

2 PROBLEM

In this section, we introduce our notation and present the lifelong learning problem.

Let $\mathcal{X}$ and $\mathcal{Y}$ be some sets. A predictor is a function $f : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{Y} = \mathbb{R}$ for regression and $\mathcal{Y} = \{-1, 1\}$ for binary classification. The loss of a predictor f on a pair (x, y) is a real number denoted by $\ell(f(x), y)$. As mentioned above, we want to transfer the information (a common data representation) gained from the previous tasks to a new one. Formally, we let $\mathcal{Z}$ be a set and prescribe a set $\mathcal{G}$ of feature maps (also called *representations*) $g : \mathcal{X} \to \mathcal{Z}$, and a set $\mathcal{H}$ of functions $h : \mathcal{Z} \to \mathbb{R}$. We shall design an algorithm that is useful when there is a function $g \in \mathcal{G}$, common to all the tasks, and task-specific functions $h_1, \dots, h_T$ such that $f_t = h_t \circ g$ is a good predictor for task t, in the sense that the corresponding prediction error (see below) is small.

We are now ready to describe the learning problem. We assume that tasks are dealt with sequentially. Furthermore, we assume that each task dataset is itself revealed sequentially and refer to this setting as *online-within-online* lifelong learning. Specifically, at each time step $t \in \{1, \dots, T\}$, the learner is challenged with a task, corresponding to a dataset

$$\mathcal{S}_t = \big((x_{t,1}, y_{t,1}), \dots, (x_{t,m_t}, y_{t,m_t})\big) \in (\mathcal{X} \times \mathcal{Y})^{m_t}$$

where $m_t \in \mathbb{N}$. The dataset $\mathcal{S}_t$ is itself revealed sequentially, that is, at each inner step $i \in \{1, \dots, m_t\}$:

- The object $x_{t,i}$ is revealed,
- The learner has to predict $y_{t,i}$, let $\hat{y}_{t,i}$ denote the prediction,
- The label $y_{t,i}$ is revealed, and the learner incurs the loss $\hat{\ell}_{t,i} := \ell(\hat{y}_{t,i}, y_{t,i})$.

The task t ends at time m_t, at which point the prediction error is

$$\frac{1}{m_t} \sum_{i=1}^{m_t} \hat{\ell}_{t,i}. \tag{2.1}$$

This process is repeated for each task t, so that at the end of all the tasks, the average error is

$$\frac{1}{T} \sum_{t=1}^{T} \frac{1}{m_t} \sum_{i=1}^{m_t} \hat{\ell}_{t,i}.$$

Ideally, if for a given representation g, the best predictor h_t for task t was known in advance, then an ideal learner using $h_t \circ g$ for prediction would incur the error

$$\inf_{h_t \in \mathcal{H}} \frac{1}{m_t} \sum_{i=1}^{m_t} \ell\big(h_t \circ g(x_{t,i}), y_{t,i}\big). \tag{2.2}$$

Hence, we define the within-task-regret of the representation g on task t as the difference between the prediction error (2.1) and the smallest prediction error (2.2),

$$\mathcal{R}_t(g) = \frac{1}{m_t} \sum_{i=1}^{m_t} \hat{\ell}_{t,i} - \inf_{h_t \in \mathcal{H}} \frac{1}{m_t} \sum_{i=1}^{m_t} \ell\big(h_t \circ g(x_{t,i}), y_{t,i}\big).$$

The above expression is slightly different from the usual notion of regret (Cesa-Bianchi and Lugosi, 2006), which does not contain the factor $1/m_t$. This normalization is important in that it allows us to give equal weights to different tasks.

Note that an oracle who would have known the best common representation g for all tasks in advance would have only suffered, on the entire sequence of datasets, the error

$$\inf_{g \in \mathcal{G}} \frac{1}{T} \sum_{t=1}^{T} \inf_{h_t \in \mathcal{H}} \frac{1}{m_t} \sum_{i=1}^{m_t} \ell\big(h_t \circ g(x_{t,i}), y_{t,i}\big).$$

We are now ready to state our principal objective: we wish to design a procedure (meta-algorithm) that, at the beginning of each task t, produces a function $\hat{g}_t$ so that, within each task, the learner can use its own favorite online learning algorithm to solve task t on the sequence $\big((\hat{g}_t(x_{t,1}), y_{t,1}), \ldots, (\hat{g}_t(x_{t,m_t}), y_{t,m_t})\big)$. We wish to control the *compound regret* of this procedure, which is defined as

$$\mathcal{R} := \frac{1}{T}\sum_{t=1}^{T}\frac{1}{m_t}\sum_{i=1}^{m_t}\hat{\ell}_{t,i} - \inf_{g\in\mathcal{G}}\frac{1}{T}\sum_{t=1}^{T}\inf_{h_t\in\mathcal{H}}\frac{1}{m_t}\sum_{i=1}^{m_t}\ell\big(h_t\circ g(x_{t,i}), y_{t,i}\big)$$

and can more concisely be written as the expression $\inf_{g\in\mathcal{G}}\big\{\frac{1}{T}\sum_{t=1}^{T}\mathcal{R}_t(g)\big\}$. The above objective is accomplished in Section 3 under the assumption that a regret bound for the within-task-algorithm is available.

We end this section with two examples included in the framework.

Example 2.1 (Dictionary learning)**.** Set $\mathcal{Z} = \mathbb{R}^K$, and call $g = (g_1, \ldots, g_K)$ a *dictionary*, where each g_k is a real-valued function on $\mathcal{X}$. Furthermore choose $\mathcal{H}$ to be a set of linear functions on $\mathbb{R}^K$, so that, for each task t

$$h_t \circ g(x) = \sum_{k=1}^{K}\theta_k^{(t)} g_k(x).$$

In practice depending on the value of K, we can use least square estimators or LASSO to learn $\theta^{(t)}$. In (Maurer et al., 2013; Ruvolo and Eaton, 2013), the authors consider $\mathcal{X} = \mathbb{R}^d$ and $g(x) = Dx$ for some $d \times K$ matrix D, and the goal is to learn jointly the predictors $\theta^{(t)}$ and the dictionary D.

Example 2.2 (Finite set $\mathcal{G}$)**.** We choose $\mathcal{G} = \{g_1, \ldots, g_K\}$ and $\mathcal{H}$ any set. While this example is interesting in its own right, it is also instrumental in studying the continuous case via a suitable discretization process. A similar choice has been considered by Crammer and Mansour (2012) in the multitask setting, in which the goal is to bound the average error on a prescribed set of tasks.

We notice that a slightly different learning setting is obtained when each dataset $\mathcal{S}_t$ is given all at once. We refer to this as *batch-within-online* lifelong learning; this setting is briefly considered in Appendix C. On the other hand when all datasets are revealed all at once, we are in the well-known setting of *learning-to-learn* (Baxter, 2000). In Section 6, we explain how our lifelong learning analysis can be adapted to this setting.

Algorithm 1 EWA-LL

Data A sequence of datasets $\mathcal{S}_t = \big((x_{t,1}, y_{t,1}), \ldots, (x_{t,m_t}, y_{t,m_t})\big)$, $1 \le t \le T$. associated with different learning tasks; the points within each dataset are also given sequentially.

Input A prior π_1, a learning parameter $\eta > 0$ and a learning algorithm for each task t which, for any representation g returns a sequence of predictions $\hat{y}_{t,i}^g$ and suffers a loss

$$\hat{L}_t(g) := \frac{1}{m_t}\sum_{i=1}^{m_t}\ell\left(\hat{y}_{t,i}^g, y_{t,i}\right).$$

Loop For $t = 1, \ldots, T$

i Draw $\hat{g}_t \sim \pi_t$.

ii Run the within-task learning algorithm on $\mathcal{S}_t$ and suffer loss $\hat{L}_t(\hat{g}_t)$.

iii Update

$$\pi_{t+1}(\mathrm{d}g) := \frac{\exp(-\eta\hat{L}_t(g))\pi_t(\mathrm{d}g)}{\int\exp(-\eta\hat{L}_t(\gamma))\pi_t(\mathrm{d}\gamma)}.$$

3 ALGORITHM

In this section, we present our lifelong learning algorithm, derive its regret bound and then specify it to two common within-task online algorithms.

3.1 EWA-LL Algorithm

Our EWA-LL algorithm is outlined in Algorithm 1. The algorithm is based on the exponentially weighted aggregation procedure (see e.g. Cesa-Bianchi and Lugosi, 2006, and references therein), and it updates a probability distribution π_t on the set of representations $\mathcal{G}$ before the encounter of each task t. The effect of Step **iii** is that any representation g which does not perform well on task t, is less likely to be reused on the next task. We insist on the fact that this procedure allows the user to freely choose the within-task algorithm, which does not even need to be the same for each task.

3.2 Bounding the Expected Regret

Since Algorithm 1 involves a randomization strategy, we can only get a bound on the expected regret, the expectation being with respect to the drawing of the function $\hat{g}_t$ at step **i** in the algorithm. Let $\mathbb{E}_{g\sim\pi}[F(g)]$ denote the expectation of $F(g)$ when $g \sim \pi$. Note that the expected overall-average loss that we want to

upper bound is then

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{\hat{g}_t\sim\pi_t}[\hat{L}_t(\hat{g}_t)].$$

Theorem 3.1. *If, for any $g\in\mathcal{G}$, $\hat{L}_t(g)\in[0,C]$ and the within-task algorithm has a regret bound $\mathcal{R}_t(g)\leq\beta(g,m_t)$, then*

$$\begin{aligned}&\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{\hat{g}_t\sim\pi_t}\left[\frac{1}{m_t}\sum_{i=1}^{m_t}\hat{\ell}_{t,i}\right]\\&\leq\inf_{\rho}\Bigg\{\mathbb{E}_{g\sim\rho}\left[\frac{1}{T}\sum_{t=1}^{T}\inf_{h_t\in\mathcal{H}}\frac{1}{m_t}\sum_{i=1}^{m_t}\ell\big(h_t\circ g(x_{t,i}),y_{t,i}\big)\right.\\&\qquad\left.+\frac{1}{T}\sum_{t=1}^{T}\beta(g,m_t)\right]+\frac{\eta C^2}{8}+\frac{\mathcal{K}(\rho,\pi_1)}{\eta T}\Bigg\},\end{aligned}$$

where the infimum is taken over all probability measures ρ and $\mathcal{K}(\rho,\pi_1)$ is the Kullback-Leibler divergence between ρ and π_1.

The proof is given in Appendix A. Some comments are in order as the bound in Theorem 3.1 might not be easy to read. First, similar to standard analyses in online learning, the parameter η is a decreasing function of T, hence the bound vanishes as T grows. Second, corollaries are derived in Section 4 that are easier to read, as they are more similar to usual regret inequalities (Cesa-Bianchi and Lugosi, 2006), that is, the right hand side of the bound is of the form

$$\inf_{g\in\mathcal{G}}\frac{1}{T}\sum_{t=1}^{T}\inf_{h_t\in\mathcal{H}}\frac{1}{m_t}\sum_{i=1}^{m_t}\ell\big(h_t\circ g(x_{t,i}),y_{t,i}\big)+\text{“rate”}.\quad(3.1)$$

The bound in Theorem 1 looks slightly different, but is quite similar in spirit. Indeed, instead of an infimum with respect to g we have an infimum over all possible aggregations with respect to g,

$$\inf_{\rho}\mathbb{E}_{g\sim\rho}\frac{1}{T}\sum_{t=1}^{T}\inf_{h_t\in\mathcal{H}}\frac{1}{m_t}\sum_{i=1}^{m_t}\ell\big(h_t\circ g(x_{t,i}),y_{t,i}\big)+\text{“remainder”}$$

where the remainder term depends on $\mathcal{K}(\rho,\pi_1)$. In order to look like (3.1), we could consider a measure ρ highly concentrated around the representation g minimizing (3.1). When $\mathcal{G}$ is finite, this is a reasonable strategy and the bound is given explicitly in Section 4.1 below. However, in some situations, this would cause the term $\mathcal{K}(\rho,\pi_1)$ to diverge. Studying accurately the minimizer in ρ usually leads to an interesting regret bound, and this is exactly what is done in Section 4.

Algorithm 2 OGA

Data A task $\mathcal{S}_t=\big((x_{t,1},y_{t,1}),\dots,(x_{t,m_t},y_{t,m_t})\big)$.

Input Stepsize $\zeta>0$, and $\theta_1=0$.

Loop For $i=1,\dots,m_t$,

i Predict $\hat{y}^g_{t,i}=h_{\theta_i}\circ g(x_{t,i})$,

ii $y_{t,i}$ is revealed, update $\theta_{i+1}=\theta_i-\zeta\nabla_\theta\ell\big(h_\theta\circ g(x_{t,i}),y_{t,i}\big)\big|_{\theta=\theta_i}$.

Finally note that the bound in Theorem 3.1 is given in expectation. In online learning, uniform bounds are usually prefered (Cesa-Bianchi and Lugosi, 2006). In Section 5 we show that it is possible to derive such bounds under additional assumptions.

3.3 Examples of Within Task Algorithms

We now specify the general bound in Theorem 1 to two common online algorithms which we use within tasks.

3.3.1 Online Gradient

The first algorithm assumes that $\mathcal{H}$ is a parametric family of functions $\mathcal{H}=\{h_\theta,\theta\in\mathbb{R}^p,\|\theta\|\leq B\}$, and for any (x,y,g), $\theta\mapsto\ell(h_\theta\circ g(x),y)$ is convex, L-Lipschitz, upper bounded by C and denote by ∇_θ a subgradient.

Corollary 3.2. *The EWA-LL algorithm using the OGA within task with step size $\zeta=\frac{B}{L\sqrt{2m_t}}$ satisfies*

$$\begin{aligned}&\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{\hat{g}_t\sim\pi_t}\left[\frac{1}{m_t}\sum_{i=1}^{m_t}\hat{\ell}_{t,i}\right]\\&\leq\inf_{\rho}\Bigg\{\mathbb{E}_{g\sim\rho}\left[\frac{1}{T}\sum_{t=1}^{T}\inf_{h_t\in\mathcal{H}}\frac{1}{m_t}\sum_{i=1}^{m_t}\ell(h_t\circ g(x_{t,i}),y_{t,i})\right.\\&\qquad\left.+\frac{BL}{T}\sum_{t=1}^{T}\sqrt{\frac{2}{m_t}}\right]+\frac{\eta C^2}{8}+\frac{\mathcal{K}(\rho,\pi_1)}{\eta T}\Bigg\}.\end{aligned}$$

Proof. Apply Theorem 3.1 and use the bound $\mathcal{R}_t(g)\leq\beta(g,m_t):=BL\sqrt{2/m_t}$ that can be found, for example, in (Shalev-Shwartz, 2011, Corollary 2.7). □

We note that under additional assumptions on loss functions, (Hazan et al., 2007, Theorem 1) provides bounds for $\beta(g,m_t)$ that are in $\log(m_t)/m_t$.

3.3.2 Exponentially Weighted Aggregation

The second algorithm is based on the EWA procedure on the space $\mathcal{H}\circ g$ for a prescribed representation $g\in\mathcal{G}$. Recall that a function $\varphi:\mathbb{R}\to\mathbb{R}$ is called ζ_0-exp-

Algorithm 3 EWA

Data A task $\mathcal{S}_t = \big((x_{t,1}, y_{t,1}), \dots, (x_{t,m_t}, y_{t,m_t})\big)$.

Input Learning rate $\zeta > 0$; a prior probability distribution μ_1 on $\mathcal{H}$.

Loop For $i = 1, \dots, m_t$,

i Predict $\hat{y}^g_{t,i} = \int_{\mathcal{H}} h \circ g(x_{t,i}) \mu_i(\mathrm{d}h)$,

ii $y_{t,i}$ is revealed, update

$$\mu_{i+1}(\mathrm{d}h) = \frac{\exp(-\zeta \ell(h \circ g(x_{t,i}), y_{t,i}))\mu_i(\mathrm{d}h)}{\int \exp(-\zeta \ell(u \circ g(x_{t,i}), y_{t,i}))\mu_i(\mathrm{d}u)}.$$

concave if $\exp(-\zeta_0 \varphi)$ is concave.

Corollary 3.3. *Assume that $\mathcal{H}$ is finite and that there exists $\zeta_0 > 0$ such that for any y, the function $\ell(\cdot, y)$ is ζ_0-exp-concave and upper bounded by a constant C. Then the EWA-LL algorithm using the EWA within task with $\zeta = \zeta_0$ satisfies*

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{\hat{g}_t \sim \pi_t}\left[\frac{1}{m_t}\sum_{i=1}^{m_t} \hat{\ell}_{t,i}\right] \le \inf_{\rho}\left\{\mathbb{E}_{g\sim\rho}\left[\frac{1}{T}\sum_{t=1}^{T}\inf_{h_t \in \mathcal{H}} \frac{1}{m_t}\sum_{i=1}^{m_t} \ell\big(h_t \circ g(x_{t,i}), y_{t,i}\big) + \frac{1}{T}\sum_{t=1}^{T}\frac{\zeta_0 \log|\mathcal{H}|}{m_t}\right] + \frac{\eta C^2}{8} + \frac{\mathcal{K}(\rho, \pi_1)}{\eta T}\right\}.$$

Proof. Apply Theorem 3.1 and use the bound $\mathcal{R}_t(g) \le \beta(g, m_t) := \zeta_0 \log|\mathcal{H}|/m_t$ that can be found, for example, in (Gerchinovitz, 2011, Theorem 2.2). □

A typical example is the quadratic loss function $\ell(y', y) = (y' - y)^2$. When there is some B such that $|y_{t,i}| \le B$ and $|h \circ g(x_{t,i})| \le B$, then the exp-concavity assumption is verified with $\zeta_0 = 1/(8B)$ and the boundedness assumption with $C = 4B^2$.

Note that when the exp-concavity assumption does not hold, Gerchinovitz (2011) derives a bound $\beta(g, m_t) = B\sqrt{\log(|\mathcal{H}|)/(2m_t)}$ with the choice $\zeta = (2/B)\sqrt{2\log(|\mathcal{H}|)/m_t}$. Moreover, PAC-Bayesian type bounds in various settings (including infinite $\mathcal{H}$) can be found in (Catoni, 2004; Audibert, 2006; Gerchinovitz, 2013). We refer the reader to (Gerchinovitz, 2011) for a comprehensive survey.

4 APPLICATIONS

In this section, we discuss two important applications. To ease our presentation, we assume that all the tasks have the same sample size, that is $m_t = m$ for all t.

4.1 Finite Subset of Relevant Predictors

We give details on Example 2.2, that is we assume that $\mathcal{G}$ is a set of K functions. Note that step **iii** in Algorithm 1 boils down to update K weights,

$$\pi_t(g_k) = \frac{\exp(-\eta \hat{L}_t(g_k))\pi_{t-1}(g_k)}{\sum_{j=1}^{K}\exp(-\eta \hat{L}_t(g_j))\pi_{t-1}(g_j)}.$$

Theorem 4.1. *Under the assumptions of Theorem 3.1, if we set $\eta = \frac{2}{C}\sqrt{\frac{2\log K}{T}}$ and π_1 uniform on $\mathcal{G}$,*

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{\hat{g}_t \sim \pi_t}\left[\frac{1}{m}\sum_{i=1}^{m} \hat{\ell}_{t,i}\right] \le \min_{1\le k\le K}\left\{\frac{1}{T}\sum_{t=1}^{T}\inf_{h_t\in\mathcal{H}}\frac{1}{m}\sum_{i=1}^{m}\ell(h_t \circ g_k(x_{t,i}), y_{t,i}) + \beta(g_k, m)\right\} + C\sqrt{\frac{\log K}{2T}}.$$

Proof. Fix $g \in \mathcal{G}$, ρ as the Dirac mass on g and note that $\mathcal{K}(\rho, \pi_1) = \log K$. □

We discussed in Sections 3.3.1 and 3.3.2 that typical orders for $\beta(g, m)$ are $\mathcal{O}(1/\sqrt{m})$, $\mathcal{O}(\log(m)/m)$ or $\mathcal{O}(1/m)$. We state a precise result in the finite case.

Corollary 4.2. *Assume that $\mathcal{H}$ is finite, that for some $\zeta_0 > 0$, for any y, the function $\ell(\cdot, y)$ is ζ_0-exp-concave and upper bounded by a constant C. Then the EWA-LL algorithm using the EWA within task with $\zeta = \zeta_0$ satisfies*

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{\hat{g}_t \sim \pi_t}\left[\frac{1}{m}\sum_{i=1}^{m} \hat{\ell}_{t,i}\right] \le \min_{1\le k\le K}\frac{1}{T}\sum_{t=1}^{T}\min_{h_t\in\mathcal{H}}\frac{1}{m}\sum_{i=1}^{m}\ell(h_t \circ g_k(x_{t,i}), y_{t,i}) + \frac{\zeta_0 \log|\mathcal{H}|}{m} + C\sqrt{\frac{\log K}{2T}}.$$

In Section 6, we derive from Theorem 3.1 a bound in the batch setting. As we shall see, in the finite case the bound is exactly the same as the bound on the compound regret. This allows us to compare our results to previous ones obtained in the learning-to-learn setting. In particular, our $\mathcal{O}(1/m)$ bound improves upon (Pentina and Lampert, 2014) who derived an $\mathcal{O}(1/\sqrt{m})$ bound.

4.2 Dictionary Learning

We now give details on Example 2.1 in the linear case. Specifically, we let $\mathcal{X} = \mathbb{R}^d$, we let $\mathcal{D}_K$ be the set

formed by all $d \times K$ matrices D, whose columns have euclidean norm equal to one, and we define $\mathcal{G} = \{x \mapsto Dx : D \in \mathcal{D}_K\}$. Within this subsection we assume that the loss ℓ is convex and Φ-Lipschitz with respect to its first argument, that is, for every $y \in \mathcal{Y}$ and $a_1, a_2 \in \mathbb{R}$, it holds $|\ell(a_1, y) - \ell(a_2, y)| \leq \Phi|a_1 - a_2|$. We also assume that for all $(t, i) \in \{1, \ldots, T\} \times \{1, \ldots, m\}$, $\|x_{t,i}\| \leq 1$, and $\beta(m) := \sup_{g \in \mathcal{G}} \beta(m, g) < +\infty$.

We define the prior π_1 as follows: the columns of D are i.i.d., uniformly distributed on the d-dimensional unit sphere.

Theorem 4.3. *Under the assumptions of Theorem 3.1, with* $\eta = \frac{2}{C}\sqrt{\frac{Kd}{T}}$,

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{\hat{g}_t \sim \pi_t}\left[\frac{1}{m}\sum_{i=1}^{m} \hat{\ell}_{t,i}\right] \leq \inf_{D \in \mathcal{D}_K}\left\{\frac{1}{T}\sum_{t=1}^{T}\inf_{h_t \in \mathcal{H}}\frac{1}{m}\sum_{i=1}^{m}\ell\big(\langle h_t, Dx_{t,i}\rangle, y_{t,i}\big) + \frac{C}{4}\sqrt{\frac{Kd}{T}}(\log(T)+7) + \beta(m)\right\} + \frac{B\Phi}{\sqrt{T}}\sqrt{\frac{1}{T}\sum_{t=1}^{T}\lambda_{\max}\left(\frac{1}{m}\sum_{i=1}^{m} x_{t,i}x_{t,i}^T\right)}.$$

The proof relies on an application of Theorem 3.1. The calculations being tedious, we postpone the proof to Appendix A.

When we use OGA within tasks, we can use Corollary 3.2 with $L = \Phi\sqrt{K}$ and so $\beta(m) \leq \Phi B\sqrt{2K/m}$ for any $D \in \mathcal{D}_K$. Moreover,

$$\lambda_{\max}\left(\frac{1}{m}\sum_{i=1}^{m} x_{t,i}x_{t,i}^T\right) \leq \mathrm{tr}\left(\frac{1}{m}\sum_{i=1}^{m} x_{t,i}x_{t,i}^T\right) \leq 1 \quad (4.1)$$

so Theorem 4.3 leads to the following corollary.

Corollary 4.4. *Algorithm EWA-LL for dictionary learning, with* $\eta = (2/C)\sqrt{Kd/T}$, *and using the OGA algorithm within tasks, with step* $\zeta = B/(\Phi\sqrt{2mK})$, *satisfies*

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}_{\hat{g}_t \sim \pi_t}\left[\frac{1}{m}\sum_{i=1}^{m} \hat{\ell}_{t,i}\right] \leq \inf_{D \in \mathcal{D}_K}\frac{1}{T}\sum_{t=1}^{T}\inf_{h_t \in \mathcal{H}}\frac{1}{m}\sum_{i=1}^{m}\ell\big(\langle h_t, Dx_{t,i}\rangle, y_{t,i}\big) + \frac{C}{4}\sqrt{\frac{Kd}{T}}(\log(T)+7) + \frac{B\Phi}{\sqrt{T}} + \frac{\Phi B\sqrt{2K}}{\sqrt{m}}.$$

Note that the upper bound (4.1) may be loose. For example, when the $x_{t,i}$ are i.i.d. on the unit sphere, $\lambda_{\max}\left(\sum_{i=1}^{m} x_{t,i}x_{t,i}^T/m\right)$ is close to $1/d$. In this case, it is possible to improve the term $\beta(m)$ employed in the calculation of the bound, we postpone the lengthy details to Appendix B.

Algorithm 4 EWA-LL for dictionary learning

Data As in Algorithm 1.

Input A learning rate η for EWA and a learning rate ζ for the online gradient. A number of steps N for the Metropolis-Hastings algorithm.

Start Draw $\hat{g}_1 \sim \pi_1$.

Loop For $t = 1, \ldots, T$

- **i** Run the within-task learning algorithm $\mathcal{S}_t$ and suffer loss $\hat{L}_t(\hat{g}_t)$.
- **ii** Set $\tilde{g} := \hat{g}_t$.
- **iii** Metropolis-Hastings algorithm. Repeat N times
  - **a** Draw $\tilde{g}' \sim \mathcal{N}(\tilde{g}, \sigma^2 I)$ and then set $\tilde{g}' := \tilde{g}'/\|\tilde{g}'\|$.
  - **b** Set $\tilde{g} := \tilde{g}'$ with probability
  $$\min\left\{1, \exp\left[\eta\sum_{h=1}^{t}\left(\hat{L}_h(\tilde{g}) - \hat{L}_h(\tilde{g}')\right)\right]\right\},$$
  $\tilde{g}$ remains unchanged otherwise.
- **iv** Set $\hat{g}_t := \tilde{g}$.

4.2.1 Algorithmic Details and Simulations

We implement our meta-algorithm Randomized EWA in this setting. The algorithm used within each task is the simple version of the online gradient algorithm outlined in Section 3.3.1. In order to draw $\hat{g}_t$ from π_t, we use N-steps of Metropolis-Hastings algorithm with a normalized Gaussian proposal (see, for example, Robert and Casella, 2013). In order to ensure a short burn-in period, we use the previous drawing $\hat{g}_{t-1}$ as a starting point. The procedure is given in Algorithm 4. Note the bottleneck of the algorithm: in step **b** we have to compare $\tilde{g}$ and $\tilde{g}'$ on the whole dataset so far.

We now present a short simulation study. We generate data in the following way: we let $K = 2$, $d = 5$, $T = 150$ and $m = 100$. The columns of D are drawn uniformly on the unit sphere, and task regression vectors θ_t are also independent and have i.i.d. coordinates in $\mathcal{U}[-1, 1]$. We generate the datasets $\mathcal{S}_t$ as follows: all the $x_{t,i}$ are i.i.d. from the same distribution as θ_t, and $y_{t,i} = \langle \theta_t, Dx_{t,i}\rangle + \varepsilon_{t,i}$ where the $\varepsilon_{t,i}$ are i.i.d. $\mathcal{N}(0, \sigma^2)$ and $\sigma = 0.1$.

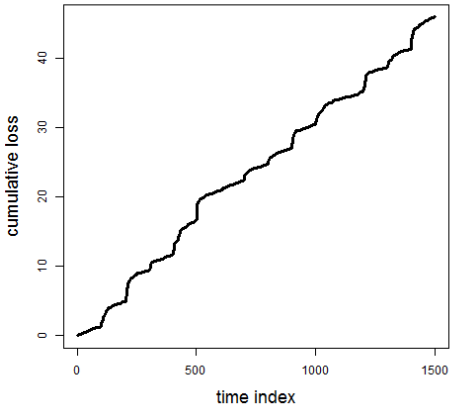

Figure 1: The cumulative loss of the oracle for the first 15 tasks.

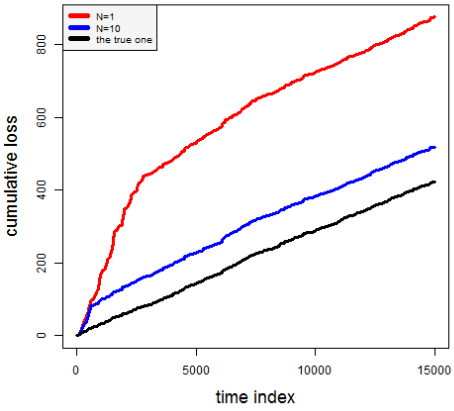

Figure 2: Cumulative loss of EWA-LL ($N = 1$ in red and $N = 10$ in blue) and cumulative loss of the oracle.

We compare Algorithm 4 with $N = 10$ to an oracle who knows the representation D, but not the task regression vectors θ_t, and learns them using the online gradient algorithm with step size $\zeta = 0.1$. Notice that after each chunk of 100 observations, a new task starts, so the parameter θ_t changes. Thus, the oracle incurs a large loss until it learns the new θ_t (usually within a few steps). This explains the "stair" shape of the cumulative loss of the oracle in Figure 1. Figure 2 indicates that after a few tasks, the dictionary D is learnt by EWA-LL: its cumulative loss becomes parallel to the one of the oracle. Due to the bottleneck mentioned above, the algorithm becomes quite slow to run when t grows. In order to improve the speed of the algorithm, we also tried Algorithm 4 with $N = 1$. There is absolutely no theoretical justification for this, however, obviously the algorithm is 10 times faster. As we can see on the red line in Figure 2, this version of the algorithm still learns D, but it takes more steps. Note that this is not completely unexpected: the Markov chain generated by this algorithm is no longer stationary, but it can still enjoy good mixing properties. It would be interesting to study the theoretical performance of Algorithm 4. However, this would require considerably technical tools from Markov chain theory which are beyond the scope of this paper.

5 UNIFORM BOUNDS

In this section, we show that it possible to obtain a uniform bound, as opposed to a bound in expectation as in Theorem 3.1. From a theoretical perspective, the price to pay is very low: we only have to assume that the loss function is convex with respect to its first argument. However, in practice, there is an aggregation step that might not be feasible. This is discussed at the end of the section. The algorithm is outlined in Algorithm 5.

Theorem 5.1. *Assume that for any $g, 0 \leq \hat{L}_t(g) \leq C$ and that the algorithm used within-task has a regret $\mathcal{R}_t(g) \leq \beta(g, m_t)$. Assume that ℓ is convex with respect to its first argument. Then it holds that*

$$\frac{1}{T}\sum_{t=1}^{T}\frac{1}{m_t}\sum_{i=1}^{m_t}\ell\left(\hat{y}_{t,i}, y_{t,i}\right)$$
$$\leq \inf_{\rho}\left\{\mathbb{E}_{g\sim\rho}\left[\frac{1}{T}\sum_{t=1}^{T}\inf_{h_t\in\mathcal{H}}\frac{1}{m_t}\sum_{i=1}^{m_t}\ell(h_t\circ g(x_{t,i}), y_{t,i}) + \frac{1}{T}\sum_{t=1}^{T}\beta(g, m_t)\right] + \frac{\eta C^2}{8} + \frac{\mathcal{K}(\rho,\pi_1)}{\eta T}\right\}.$$

Proof. At each step t, the loss suffered by the algorithm is

$$\frac{1}{m_t}\sum_{i=1}^{m_t}\ell\big(\hat{y}_{t,i}, y_{t,i}\big) = \frac{1}{m_t}\sum_{i=1}^{m_t}\ell\bigg(\int \hat{y}^g_{t,i}\pi_t(\mathrm{d}g), y_{t,i}\bigg)$$
$$\leq \frac{1}{m_t}\sum_{i=1}^{m_t}\int \ell\big(\hat{y}^g_{t,i}, y_{t,i}\big)\pi_t(\mathrm{d}g) = \int \hat{L}_t(g)\pi_t(\mathrm{d}g)$$

and we can just apply Theorem 3.1. □

In practice, for an infinite set $\mathcal{G}$ we are not able to run simultaneously the within-task algorithm for all $g \in \mathcal{G}$. So, we cannot compute the prediction (5.1) exactly. A possible strategy is to draw N elements of $\mathcal{G}$ i.i.d. from π_t, say $\hat{g}_t(1), \ldots, \hat{g}_t(N)$, and to replace (5.1) by

$$\hat{y}^{(N)}_{t,i} = \frac{1}{N}\sum_{j=1}^{N}\hat{y}^{\hat{g}_t(j)}_{t,i}.$$

An application of Hoeffding's inequality shows for any $\delta > 0$, with probability at least $1 - \delta$, the bound in Theorem 5.1 will still hold, up to an additional term $C\sqrt{\log(T/\delta)/2N}$.

Algorithm 5 Integrated EWA-LL

Data and Input same as in Algorithm 1.

Loop For $t = 1, \ldots, T$

i Run the within-task learning algorithm on $\mathcal{S}_t$ for each $g \in \mathcal{G}$ and return as predictions:

$$\hat{y}_{t,i} = \int \hat{y}^g_{t,i} \pi_t(\mathrm{d}g). \tag{5.1}$$

ii Update $\pi_{t+1}(\mathrm{d}g) := \frac{\exp(-\eta \hat{L}_t(g))\pi_t(\mathrm{d}g)}{\int \exp(-\eta \hat{L}_t(\gamma))\pi_t(\mathrm{d}\gamma)}$.

6 LEARNING-TO-LEARN

In this section, we show how our analysis of lifelong learning can be used to derive bounds for learning-to-learn. In this setting, the tasks and their datasets are generated by first sampling task distributions $P_1, \ldots, P_T$ i.i.d. from a"meta-distribution"Q, called *environment* by Baxter (2000), and then for each task t, a dataset $\mathcal{S}_t = ((x_{t,1}, y_{t,1}), \ldots, (x_{t,m}, y_{t,m}))$ is sampled i.i.d. from P_t. We stress that in this setting, the entire data $(x_{t,i}, y_{t,i})_{1 \le i \le m, 1 \le t \le T}$ is given all at once to the learner. Note that for simplicity, we assumed that all the sample sizes are the same.

We wish to design a strategy which, given a new task $P \sim Q$ and a new sample $(x_1, y_1), \ldots, (x_m, y_m)$ i.i.d. from P, computes a function $f : \mathcal{X} \to \mathcal{Y}$, that will predict y well when $(x, y) \sim P$. For this purpose we propose the following strategy:

1. Run EWA-LL on $(x_{t,i}, y_{t,i})_{1 \le i \le m, 1 \le t \le T}$. We obtain a sequence of representations $\hat{g}_1, \ldots, \hat{g}_T$,
2. Draw uniformly $\mathcal{T} \in \{1, \ldots, T\}$ and put $\hat{g} = \hat{g}_{\mathcal{T}}$,
3. Run the within task algorithm on the sample $(x_i, y_i)_{1 \le i \le m}$, obtaining a sequence $h^{\hat{g}}_1, \ldots, h^{\hat{g}}_m$ of functions,
4. Draw uniformly $\mathcal{I} \in \{1, \ldots, m\}$ and put $\hat{h} = h^{\hat{g}}_{\mathcal{I}}$.

Our next result establishes that this strategy leads indeed to safe predictions.

Theorem 6.1. *Let $\mathbb{E}$ be the expectation over all data pairs $(x_{t,i}, y_{t,i})_{1 \le i \le m} \sim P_t$, $(P_t)_{1 \le t \le T} \sim Q$, $(x_i, y_i)_{1 \le i \le m} \sim P$, $(x, y) \sim P$, $P \sim Q$ and also over the randomized decisions of the learner $(\hat{g}_t)_{1 \le t \le T}$, $\mathcal{T}$ and $\mathcal{I}$. Then*

$$\mathbb{E}[\ell(\hat{h} \circ \hat{g}(x), y)] \le \inf_{\rho} \left\{ \mathbb{E}_{g \sim \rho} \left[\mathbb{E}_{P \sim Q} \inf_{h \in \mathcal{H}} \mathbb{E}_{(x,y) \sim P} \left[\ell(h \circ g(x), y) \right] + \beta(g, m) \right] + \frac{\eta C^2}{8} + \frac{\mathcal{K}(\rho, \pi_1)}{\eta T} \right\}.$$

The proof is given in Appendix A. As in Theorem 3.1, the result is given in expectation with respect to the randomized decisions of the learner. Assuming that ℓ is convex with respect to its first argument, we can state a similar result for a non-random procedure, as was done in Section 5. Details are left to the reader.

Remark 6.1. In (Baxter, 2000; Maurer et al., 2013; Pentina and Lampert, 2014), the results on learning-to-learn are given with large probability with respect to $(x_{t,i}, y_{t,i})_{1 \le i \le m, 1 \le t \le T}$, rather than in expectation. Using the machinery in (Cesa-Bianchi and Lugosi, 2006, Lemma 4.1) we conjecture that it is possible to derive a bound in probability from Theorem 6.1.

7 CONCLUDING REMARKS

We presented a meta-algorithm for lifelong learning and derived a fully online analysis of its regret. An important advantage of this algorithm is that it inherits the good properties of any algorithm used to learn within tasks. Furthermore, using online-to-batch conversion techniques, we derived bounds for the related framework of learning-to-learn.

We discussed the implications of our general regret bounds for two applications: dictionary learning and finite set $\mathcal{G}$ of representations. Further applications of this algorithm which may be studied within our framework are deep neural networks and kernel learning. In the latter case, which has been addressed by Pentina and Ben-David (2015) in the learning-to-learn setting, $g : \mathcal{X} \to \mathcal{Z}$ is a feature map to a reproducing kernel Hilbert space $\mathcal{Z}$, and $h_t(g(x)) = \langle z^{(t)}, g(x) \rangle_{\mathcal{Z}}$. In the former case, $\mathcal{X} = \mathbb{R}^d$ and $g : \mathcal{X} \to \mathbb{R}^K$ is a multilayer network, that is a vector-valued function obtained by successive application of a linear transformation and a nonlinear activation function. The predictor $h : \mathbb{R}^K \to \mathbb{R}$ is typically a linear function. The vector-valued function $(h_1 \circ g, \ldots, h_T \circ g))$ models a multilayer network with shared hidden weights. This is discussed in (Maurer et al., 2016), again in the learning-to-learn setting.

Perhaps the most fundamental question is to extend our analysis to more computationally efficient algorithms such as approximations of EWA, like Algorithm 4, or fully gradient based algorithms.

Acknowledgements

We are grateful to Mark Herbster for useful comments. This work was supported by the research programme *New Challenges for New Data* from LCL and GENES, hosted by the *Fondation du Risque*, from Labex ECODEC (ANR - 11-LABEX-0047) and EPSRC grant EP/P009069/1.

References

Audibert, J.-Y. (2006). A randomized online learning algorithm for better variance control. In *Proc. 19th Annual Conference on Learning Theory*, pages 392–407. Springer.

Balcan, M.-F., Blum, A., and Vempala, S. (2015). Efficient representations for lifelong learning and autoencoding. In *Proc. 28th Conference on Learning Theory*, pages 191–210.

Baxter, J. (1997). A bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning*, 28(1):7–39.

Baxter, J. (2000). A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198.

Catoni, O. (2004). *Statistical learning theory and stochastic optimization*, volume 1851 of *Saint-Flour Summer School on Probability Theory 2001 (Jean Picard ed.), Lecture Notes in Mathematics*. Springer-Verlag, Berlin.

Cavallanti, G., Cesa-Bianchi, N., and Gentile, C. (2010). Linear algorithms for online multitask classification. *Journal of Machine Learning Research*, 1:2901–2934.

Cesa-Bianchi, N. and Lugosi, G. (2006). *Prediction, learning, and games*. Cambridge University Press.

Crammer, K. and Mansour, Y. (2012). Learning multiple tasks using shared hypotheses. In *Advances in Neural Information Processing Systems 25*, pages 1475–1483.

Gerchinovitz, S. (2011). *Prédiction de suites individuelles et cadre statistique classique: étude de quelques liens autour de la régression parcimonieuse et des techniques d'agrégation*. PhD thesis, Paris 11.

Gerchinovitz, S. (2013). Sparsity regret bounds for individual sequences in online linear regression. *Journal of Machine Learning Research*, 14(1):729–769.

Hazan, E., Agarwal, A., and Kale, S. (2007). Logarithmic regret algorithms for online convex optimization. *Machine Learning*, 69(2-3):169–192.

Herbster, M., Pasteris, S., and Pontil, M. (2016). Mistake bounds for binary matrix completion. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 3954–3962.

Maurer, A. (2005). Algorithmic stability and meta-learning. *Journal of Machine Learning Research*, 6:967–994.

Maurer, A., Pontil, M., and Romera-Paredes, B. (2013). Sparse coding for multitask and transfer learning. In *Proc. 30th International Conference on Machine Learning*, pages 343–351.

Maurer, A., Pontil, M., and Romera-Paredes, B. (2016). The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81):1–32.

McAllester, D. A. (1998). Some pac-bayesian theorems. In *Proc. 11th Annual Conference on Computational Learning Theory*, pages 230–234. ACM.

Pentina, A. and Ben-David, S. (2015). Multi-task and lifelong learning of kernels. In *Proc. 26th International Conference on Algorithmic Learning Theory*, pages 194–208.

Pentina, A. and Lampert, C. (2014). A pac-bayesian bound for lifelong learning. In *Proc. 31st International Conference on Machine Learning*, pages 991–999.

Pentina, A. and Urner, R. (2016). Lifelong learning with weighted majority votes. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R., editors, *Advances in Neural Information Processing Systems 29*, pages 3612–3620. Curran Associates, Inc.

Robert, C. and Casella, G. (2013). *Monte Carlo statistical methods*. Springer Science & Business Media.

Ruvolo, P. and Eaton, E. (2013). Ella: An efficient lifelong learning algorithm. In *Proc. 30th International Conference on Machine Learning*, pages 507–515.

Shalev-Shwartz, S. (2011). Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194.

Thrun, S. (1996). Is learning the n-th thing any easier than learning the first? In *Advances in neural information processing systems*, pages 640–646.

Thrun, S. and Pratt, L. (1998). *Learning to Learn*. Kluwer Academic Publishers.

Vapnik, V. (1998). *Statistical Learning Theory*. Wiley.