Generalize Across Tasks: Efficient Algorithms for Linear Representation Learning

Brian Bullins BBULLINS@CS.PRINCETON.EDU
Google AI Princeton
Princeton University

Elad Hazan EHAZAN@CS.PRINCETON.EDU
Google AI Princeton
Princeton University

Adam Kalai ADUM@MICROSOFT.COM
Microsoft Research

Roi Livni RLIVNI@TAUEX.TAU.AC.IL
Tel Aviv University

Editors: Aurélien Garivier and Satyen Kale

Abstract

We present provable algorithms for learning linear representations which are trained in a supervised fashion across a number of tasks. Furthermore, whereas previous methods in the context of multi-task learning only allow for generalization within tasks that have already been observed, our representations are both efficiently learnable and accompanied by *generalization guarantees to unseen tasks*. Our method relies on a certain convex relaxation of a non-convex problem, making it amenable to online learning procedures. We further ensure that a low-rank representation is maintained, and we allow for various trade-offs between sample complexity and per-iteration cost, depending on the choice of algorithm.

Keywords: Multi-task learning, representation learning, online learning, generalization bounds.

1. Introduction

There are very few examples of *provable* methods that efficiently learn a succinct representation of data that can generalize well to unseen tasks. Even formally stating the goal of such a representation is non-trivial. Despite the lack of theoretical guarantees, representation learning is immensely successful in practice: deep neural representations for vision (Krizhevsky et al., 2012) and word embeddings for natural language processing (Mikolov et al., 2013; Pennington et al., 2014) are two notable examples.

In this paper we ask: what kind of representation can be learned over training data, such that it will provably generalize well to future tasks? One option is to consider generative models that give rise to numerous successful examples such as topic models and deep generative networks. However, the optimization problems arising in these contexts are typically NP-hard.

We instead take a discriminative approach. Consider a distribution over *learning problems*, each with its own training and test set. In this setting, a good representation of the data is such that we can efficiently learn a linear classifier over this representation and attain small error. This gives rise to a natural loss function over the class of representations: namely, for each task we apply a predefined

learning algorithm over the new representation (with its own generalization guarantees), and the loss of the representation is measured in terms of the loss of the (sub)-learner.

We analyze our setting when the sub-learners are linear separators (learned by optimizing a regularized convex risk minimization problem), and we focus on learning *linear* representations, a.k.a. dimensionality reduction. Surprisingly, we show that this double optimization problem of learning a linear representation that can be optimized via linear separators can be formulated as a convex optimization problem. Thus, it can be learned efficiently. We further show how to efficiently learn a linear representation that *generalizes across tasks*, the first result of its kind in multi-task learning, to the best of our knowledge.

1.1. Related Work

Multi-task learning (Caruana, 1997) and the related approaches for learning to learn (Thrun and Pratt, 1998) have been empirically effective on numerous problems. For instance, in computer vision, the supervised learning of 1,000 classes of images (Krizhevsky et al., 2012) has led to improved underlying image representations useful across a number of tasks. In the context of word embeddings, due to their successful application across a number of fields, a common approach to representing sentences and paragraphs has been to use a weighted average of the word vectors, though more advanced techniques have been shown to improve performance (Le and Mikolov, 2014; Arora et al., 2017). It is also common to seed NLP algorithms with word embeddings and then optimize the embedding for the particular downstream task at hand (e.g. Peng and Dredze, 2015).

Several papers analyze multi-task learning in the context of Gaussian processes (Schwaighofer et al., 2004; Yu et al., 2005; Bonilla et al., 2007), as well as other generative assumptions (Hernández-Lobato et al., 2010; Guo et al., 2011). Issues such as sample complexity have also been investigated in several papers (Baxter et al., 2000; Ando and Zhang, 2005; Maurer and Pontil, 2013; Maurer et al., 2016). Baxter et al. (2000) and Maurer et al. (2016) in particular note their focus on the statistical complexity of the problem, rather than the algorithmic component.

Learning a linear representation has been studied by Balcan et al. (2015). Under isotropic log concave distributions, Balcan et al. (2015) learn a low-dimensional representation for linear threshold functions (half-spaces) (see also (Rish et al., 2008)). Additionally, Argyriou et al. (2008) study a related problem in finding a linear representation using multi-task learning, whereby they learn a rotation of the data to allow sparse classifiers. We do not restrict our attention to just rotations, and we allow any linear transformation that will improve classification.

Amit et al. (2007) and Harchaoui et al. (2012) suggest learning a representation coupled with classifiers using a trace norm constraint, then using factorization to output a representation. This differs from our approach in several aspects. First, this approach is not amenable to tasks being streamlined in an online manner. More importantly, our algorithm comes with an online-to-batch generalization guarantee with respect to *future tasks*. In contrast, the known bounds for trace norm constraints are restricted to *observed tasks* (see (Maurer and Pontil, 2013)). Furthermore, framing the problem in the view of online learning provides the option for optimization algorithms that induce a low-rank representation. The theoretical framework for defining generalization of representations is inspired by the non-generative framework of Hazan and Ma (2016).

The online Frank-Wolfe algorithm was first presented in the work by Hazan and Kale (2012), and improvements to the regret were shown to be possible by Garber and Hazan (2013) when optimizing over polyhedral decision sets. The works of Arora and Kale (2007) and Warmuth and Kuzmin (2012)

independently discovered the matrix multiplicative weights algorithm, the latter being based on the earlier work of Tsuda et al. (2005).

2. Problem Setup

We begin by formally defining the learning model of interest. An *empirical minimization task* in d-dimensions consists of a pair $(\mathcal{X}, \mathcal{Y})$ of a matrix $\mathcal{X} \in [-1,1]^{m \times d}$ and a label vector $\mathcal{Y} \in \{-1,1\}^m$ for some $m \in \mathbb{N}$. As different tasks can have different m, we define $|\mathcal{X}| := m$ to be the corresponding number of examples for a given task $(\mathcal{X}, \mathcal{Y})$. For normalization, we will assume that the rows of $\mathcal{X}$ (i.e., the samples within the task) are restricted to the unit ball. Our setting assumes that a learner is faced with different tasks, generated i.i.d. according to some distribution $\mathcal{D}$. Throughout the paper, unless otherwise noted, we let $\|\cdot\|$ refer to the ℓ_2-norm. In addition, we let $A \bullet B := \sum_{i,j} A_{ij} \cdot B_{ij}$ denote the matrix inner product, and we let $\|\cdot\|_F$ and $\|\cdot\|_{\mathrm{tr}}$ denote the Frobenius and trace norms of a matrix, respectively.

The aim of the learner is to choose a linear representation over the tasks from a pre-specified class of matrices. We will be concerned with two classes, namely the class of low-rank embeddings:

$$\mathcal{F}_r^{\mathrm{low}} := \left\{ M \in \mathbb{R}^{d \times r} : M \succeq 0,\ \|M\|_F = 1 \right\}$$

and the class of Frobenius norm constrained matrices:

$$\mathcal{F} := \left\{ M \in \mathbb{R}^{d \times d} : M \succeq 0,\ \|M\|_F = 1 \right\}.$$

The performance of the representation over a given task $(\mathcal{X}, \mathcal{Y})$ is measured in terms of the performance of the optimal linear classifier over the regularized empirical loss. Specifically, for a representation and a given task, the learner suffers a loss of

$$\mathcal{L}_\lambda(M; (\mathcal{X}, \mathcal{Y})) := \min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{|\mathcal{X}|}\sum_{i=1}^{|\mathcal{X}|} \ell(\mathbf{w}^\top M^\top \mathbf{x}_i, y_i), \tag{1}$$

where $\mathcal{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m]^\top$ and $\mathcal{Y} = [y_1, \ldots, y_m]^\top$.

Without loss of generality, we restrict our attention to Frobenius norm constrained matrices, as we can always learn the more complex class by scaling the λ regularizer which inhibits the norm of the classifiers to be learned.

For a fixed empirical task, the loss function $\mathcal{L}_\lambda$ is a proxy to the generalization performance of the linear classifier. For $C > 0$ and some fixed M, choosing $\lambda = O(C/\varepsilon)$ and assuming $|\mathcal{X}| = O(C/\varepsilon^2)$, it is known that solving Eq.1 over a set $\mathcal{X}$ generated i.i.d. will lead to a classifier that competes against the optimal C-bounded norm classifier in terms of generalization error (Sridharan et al., 2009). In turn, once we learn and fix a representation that leads to small expected empirical error, as long as the sample size of future tasks is large enough, we are guaranteed to learn future tasks.

The objective of the learner then is to choose a linear representation that minimizes, in expectation, the regularized empirical error objective. Namely, we wish to choose $M \in \mathcal{F}$ that would minimize

$$\mathcal{L}_\lambda(M; \mathcal{D}) := \mathop{\mathbb{E}}_{(\mathcal{X},\mathcal{Y})\sim\mathcal{D}} \left[\mathcal{L}_\lambda(M; \mathcal{X}, \mathcal{Y}) \right]. \tag{2}$$

2.1. Main Results

We next describe our main theoretical guarantees, namely two supervised algorithms for learning the optimal linear representation over empirical tasks. When using online Frank-Wolfe (OFW) as our black-box optimization procedure, we have the following generalization guarantee:

Theorem 1 *Assume for simplicity $\ell(\cdot)$ is a 1-Lipschitz convex function. Let $\mathcal{S}$ be a sample of empirical tasks drawn i.i.d. according to some distribution $\mathcal{D}$ over empirical tasks. Then, Algorithm 1 outputs $M_S \in \mathcal{F}^{\text{low}}_{|\mathcal{S}|}$ such that if $|\mathcal{S}| = O\left(\left(\frac{1}{\lambda\varepsilon}\right)^4 \log^2(\frac{1}{\delta})\right)$, then for any $\delta < \frac{1}{2}$, with probability at least $1-\delta$ over $\mathcal{S}$ we have that*

$$\mathcal{L}_\lambda(M_{\mathcal{S}}; \mathcal{D}) \leq \min_{M \in \mathcal{F}} \mathcal{L}_\lambda(M; \mathcal{D}) + \varepsilon. \tag{3}$$

We can obtain faster rates using a different algorithm, namely matrix multiplicative weights (MMW) (Tsuda et al., 2005; Arora et al., 2012), at a higher cost for each iteration:

Theorem 2 *Assume for simplicity $\ell(\cdot)$ is a 1-Lipschitz convex function. Let $\mathcal{S}$ be a sample of empirical tasks drawn i.i.d. according to some distribution $\mathcal{D}$ over empirical tasks. Then, Algorithm 2 outputs $M_S \in \mathcal{F}^{\text{low}}_{|\mathcal{S}|}$ such that if $|\mathcal{S}| = O\left(\left(\frac{1}{\lambda\varepsilon}\right)^2 \left(\sqrt{\log(d)} + \lambda\sqrt{\log(\frac{1}{\delta})}\right)^2\right)$ and $|\mathcal{S}| \geq \log(d)$, then for any $\delta < \frac{1}{2}$, with probability at least $1-\delta$ over $\mathcal{S}$ we have that*

$$\mathcal{L}_\lambda(M_{\mathcal{S}}; \mathcal{D}) \leq \min_{M \in \mathcal{F}} \mathcal{L}_\lambda(M; \mathcal{D}) + \varepsilon. \tag{4}$$

Note that whenever the problem dimension exceeds the necessary sample complexity, both Theorem 1 and Theorem 2 can provide a low-rank representation of the data due to the fact that their respective procedures inherently construct low-rank solutions.

3. Algorithm and Proofs

Our general strategy for learning the linear representation is to minimize the empirical objective using a sequential online algorithm. In the context of online learning, we may consider any repeated game between a decision maker and an adversary. The decision maker iteratively generates a hypothesis $\mathbf{x}_t$ from a convex set $\mathcal{K}$. An adversary then picks a convex loss function f_t, and the decision maker suffers a loss of $f_t(\mathbf{x}_t)$. We then define the *regret* of the online decision maker after T rounds to be

$$Regret_T := \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{x}).$$

In order to successfully apply such an online algorithm, we first need a convex formulation for our problem. Therefore, we begin with the observation that the loss over an empirical task $\mathcal{L}_\lambda(M; (\mathcal{X}, \mathcal{Y}))$ can be represented by a convex formulation using an appropriate change of variable. After having done so, we may present our online algorithm for minimizing the regret.

3.1. Convexification

Note that a priori, the loss objective (Eq.1) is the minimum of convex functions (i.e., for each fixed $\mathbf{w}$ in Eq.1 we obtain a convex problem). In general, the minimum over convex functions does not yield a convex function. However, by turning to the dual, one obtains a natural representation of the problem which *is* convex. Indeed, for a fixed empirical task $(\mathcal{X}, \mathcal{Y})$, considering the dual formulation of the problem over $\mathbf{w}$, we obtain the identity

$$\mathcal{L}_\lambda(M; (\mathcal{X}; \mathcal{Y})) = \mathcal{G}_\lambda(MM^\top; (\mathcal{X}, \mathcal{Y})),$$

where we have

$$\mathcal{G}_\lambda(MM^\top; (\mathcal{X}, \mathcal{Y})) := \max_{\boldsymbol{\alpha}} \phi_{\mathcal{X},\mathcal{Y}}(MM^\top, \boldsymbol{\alpha}),$$

and

$$\phi_{(\mathcal{X},\mathcal{Y})}(B, \boldsymbol{\alpha}) := \frac{1}{|\mathcal{X}|} \sum_{i=1}^{|\mathcal{X}|} -\ell^*(-\lambda|\mathcal{X}|\alpha_i, y_i) - \frac{\lambda}{2}\boldsymbol{\alpha}^\top \mathcal{X} B \mathcal{X}^\top \boldsymbol{\alpha}.$$

Here, $\ell^*(b, y_i)$ is the convex conjugate of $\ell(a, y_i)$, i.e., $\ell^*(b, y) := \max_a\{a \cdot b - \ell(a, y)\}$. We may note that, by definition, having $\|MM^\top\|_{\mathrm{tr}} = 1$ is equivalent to having $\|M\|_F = 1$.

The following lemma shows that by considering the dual formulation, we can obtain a convex formulation of our problem.

Lemma 3 *For every empirical task, the function $\mathcal{G}(B, (\mathcal{X}, \mathcal{Y}))$ is convex in B. Furthermore, the gradient $\nabla\mathcal{G}(B, (\mathcal{X}, \mathcal{Y}))$ is efficiently computable. Finally, if $B = MM^\top$ then*

$$\mathcal{G}_\lambda(B, (\mathcal{X}, \mathcal{Y})) = \mathcal{L}_\lambda(M, (\mathcal{X}, \mathcal{Y})).$$

Proof The identity follows from standard Fenchel duality, fixing M and considering the loss objective as a function of the variable $\mathbf{w}$. Next, note that $\phi_{\mathcal{X},\mathcal{Y}}(B, \boldsymbol{\alpha})$ is a linear function over B, hence $\mathcal{G}_\lambda(B, (\mathcal{X}, \mathcal{Y}))$ is a convex function in B (being the maximum over linear functions). Moreover, since $\mathcal{G}_\lambda(B, (\mathcal{X}, \mathcal{Y})) = \max_{\boldsymbol{\alpha}} \phi_{(\mathcal{X},\mathcal{Y})}(B, \boldsymbol{\alpha})$, we can also compute its gradient efficiently. Indeed, it is known that for a convex function that is the maximum over convex functions, its subgradient coincides with the gradient of the maximizer. In particular, we have that if $\boldsymbol{\alpha}^* = \mathrm{argmax}_{\boldsymbol{\alpha}} \phi_{(\mathcal{X},\mathcal{Y})}(B, \boldsymbol{\alpha})$, then

$$\nabla\mathcal{G}(B, (\mathcal{X}, \mathcal{Y})) = \nabla\phi_{(\mathcal{X},\mathcal{Y})}(B, \boldsymbol{\alpha}^*) = -\frac{\lambda}{2}\mathcal{X}^\top \boldsymbol{\alpha}^* \boldsymbol{\alpha}^{*\top} \mathcal{X}. \tag{5}$$

■

3.2. Algorithm

We next outline our algorithm for learning the representation, LRL (Linear Representation Learner), using both online Frank-Wolfe and matrix multiplicative weights. Algorithm 1 and Algorithm 2 perform regret minimization with respect to the class of PSD trace-norm bounded matrices B over the sequence of convex functions $\{\mathcal{G}_\lambda(B, (\mathcal{X}_t, \mathcal{Y}_t))\}_{t=1}^T$. Specifically, we apply both the online Frank-Wolfe (Hazan and Kale, 2012) and matrix multiplicative weights (Tsuda et al., 2005; Arora and Kale, 2007) algorithms as black-box procedures for an online convex optimization task.

Algorithm 1 LRL-OFW (Linear Representation Learner, via Online Frank-Wolfe)

Input: iterations T, $\varepsilon > 0$
Initialize $B_1 = e_1 e_1^\top$, $\eta = O(1/\sqrt{T})$
for $t = 1$ **to** T **do**
  Get example $(\mathcal{X}_t, \mathcal{Y}_t)$
  Let $f_t(B) := \mathcal{G}(B, (\mathcal{X}_t, \mathcal{Y}_t))$
  Predict B_t, suffer loss $f_t(B_t)$
  Compute $\boldsymbol{\alpha}_t = \operatorname{argmax}_{\boldsymbol{\alpha}} \phi_{(\mathcal{X}_t, \mathcal{Y}_t)}(B_t, \boldsymbol{\alpha})$
  Set $\nabla f_t(B) = -\frac{\lambda}{2} \mathcal{X}_t^\top \boldsymbol{\alpha}_t \boldsymbol{\alpha}_t^\top \mathcal{X}_t$
  $F_t(B) = \eta \sum_{\tau=1}^{t} \nabla f_\tau(B_\tau) \bullet B + \frac{2}{\lambda \sqrt[4]{t}} \|B - e_1 e_1^\top\|^2$
  $v_t \leftarrow$ Approx-EV $(-\nabla F_t(B_t), \varepsilon)$
  $B_{t+1} = (1 - t^{-1}) B_t + t^{-1} v_t v_t^\top$
end for
$\bar{B} = \frac{1}{T+1} \sum_{t=1}^{T+1} B_t$
return M s.t. $\bar{B} = MM^\top$

To present our algorithms, we include the notation Approx-EV(X, ε) to denote a procedure that computes (approximately, up to ε error) the largest singular value of the matrix X. The first method is presented in Algorithm 1.

One advantage to online Frank-Wolfe is that it removes the projection step and instead performs a linear optimization step, which is more efficient than the eigendecomposition needed for matrix multiplicative weights. We next state our main theorem concerning the regret achieved by Algorithm 1.

Algorithm 2 LRL-MMW (Linear Representation Learner, via Matrix Multiplicative Weights)

Input: iterations T, λ
Initialize $B_1 = \frac{1}{d} \mathbf{I}_d$, $\eta = O\left(\lambda \sqrt{\frac{\log(d)}{T}}\right)$
for $t = 1$ **to** T **do**
  Get example $(\mathcal{X}_t, \mathcal{Y}_t)$
  Let $f_t(B) := \mathcal{G}(B, (\mathcal{X}_t, \mathcal{Y}_t))$
  Predict B_t, suffer loss $f_t(B_t)$
  Compute $\boldsymbol{\alpha}_t = \operatorname{argmax}_{\boldsymbol{\alpha}} \phi_{(\mathcal{X}_t, \mathcal{Y}_t)}(B_t, \boldsymbol{\alpha})$
  Set $\nabla f_t(B) = -\frac{\lambda}{2} \mathcal{X}_t^\top \boldsymbol{\alpha}_t \boldsymbol{\alpha}_t^\top \mathcal{X}_t$
  Let $P_t := \nabla f_t(B_t)$
  $V_{t+1} = \exp\left(-\eta \sum_{\tau=1}^{t} P_\tau\right)$
  $B_{t+1} = \frac{V_{t+1}}{\|V_{t+1}\|_{\mathrm{tr}}}$
end for
$\bar{B} = \frac{1}{T+1} \sum_{t=1}^{T+1} B_t$
return M s.t. $\bar{B} = MM^\top$

Theorem 4 *Let $\mathcal{S} = \{(\mathcal{X}_t, \mathcal{Y}_t)\}_{t=1}^T$ be a stochastic sequence of empirical tasks, and let B_t be defined by the output of Algorithm 1 applied to $\mathcal{S}$. Then, for $\mathcal{K} = \{B : B \succeq 0, \|B\|_{\mathrm{tr}} = 1\}$, we have*

$$\sum_{t=1}^{T} \mathcal{G}_\lambda(B_t, (\mathcal{X}_t, \mathcal{Y}_t)) - \min_{B \in \mathcal{K}} \sum_{t=1}^{T} \mathcal{G}_\lambda(B, (\mathcal{X}_t, \mathcal{Y}_t)) \leq O\left(\frac{T^{3/4}}{\lambda}\right). \tag{6}$$

The following lemma will be useful for the proofs of both Theorem 4 and Theorem 8.

Lemma 5 *For all $t = 1, \ldots, T$, we have that*

$$\|\nabla \mathcal{G}_\lambda(B, (\mathcal{X}_t, \mathcal{Y}_t))\|_F \leq \frac{2}{\lambda}.$$

Proof As discussed, $\nabla \mathcal{G}_\lambda(B, (\mathcal{X}_t, \mathcal{Y}_t))$ is given by $-\frac{\lambda}{2}\mathcal{X}_t^\top \boldsymbol{\alpha}\boldsymbol{\alpha}^\top \mathcal{X}_t$. It is well known that we may obtain a solution to the maximizer, such that $\|\boldsymbol{\alpha}\|_1 \leq \frac{2}{\lambda}$ (Shalev-Shwartz et al., 2007). Thus, we obtain a bound on the Frobenius norm of $\nabla \mathcal{G}_\lambda(B, (\mathcal{X}_t, \mathcal{Y}_t))$:

$$\|\nabla \mathcal{G}_\lambda(B, (\mathcal{X}_t, \mathcal{Y}_t))\|_F = \frac{\lambda}{2}\|\sum_i \alpha_i \mathbf{x}_i\|^2 \leq \frac{\lambda}{2}\|\boldsymbol{\alpha}\|_1^2 \leq \frac{2}{\lambda}.$$

■

Proof [Proof of Theorem 4] We start by stating the regret bounds for online Frank-Wolfe below.

Theorem 6 *(Hazan and Kale (2012), Thm 4.4) Let $\{f_t\}_{t=1}^T$ be an arbitrary sequence of L-Lipschitz convex functions. Let $\mathcal{K}$ be a convex set with diameter at most D, and let $\{\mathbf{x}_t\}_{t=1}^T \in \mathcal{K}$ be the output of the online Frank-Wolfe algorithm applied to $\{\hat{f}_t\}_{t=1}^T$ where $\hat{f}_t = \nabla f_t(\mathbf{x}_t) + (L/D)t^{-1/4}$. Then we have that*

$$\sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x} \in \mathcal{K}} \sum_{t=1}^{T} f_t(\mathbf{x}) \leq O(LDT^{3/4}).$$

Algorithm 1 essentially performs online Frank-Wolfe as described in Theorem 6, with the sequence of functions $\{\mathcal{G}_\lambda(B, (\mathcal{X}_t, \mathcal{Y}_t))\}_{t=1}^T$ which are defined over the bounded domain $\mathcal{K} = \{B : B \succeq 0, \|B\|_{\mathrm{tr}} = 1\}$. To see how the regret is achieved, we may observe that the diameter D of $\mathcal{K}$ is indeed bounded by $O(1)$. Furthermore, by Lemma 5, we know that $\{\mathcal{G}_\lambda(B, (\mathcal{X}_t, \mathcal{Y}_t))\}_{t=1}^T$ are $O\left(\frac{1}{\lambda}\right)$-Lipschitz. Also note that for the class $\mathcal{K}$, the procedure Approx-EV calculates (approximately) the largest singular vector v of the matrix $-\nabla F_t(B)$, which in turn defines $X = vv^\top$ which is known to be the solution to $\min_{\|X\|_{\mathrm{tr}}=1} \nabla f_t(B) \bullet X$.

The rest of the proof follows from the observations made in Lemma 3 that the functions $f_t(B) = \mathcal{G}_\lambda(B, (\mathcal{X}_t, \mathcal{Y}_t))$ are indeed convex, and that the gradient of f_t is indeed $\nabla f_t(B)$, as we compute in Algorithm 1. ■

3.3. Proof of Theorem 1

Having derived the necessary bound on the regret for Algorithm 1, we may continue to prove the main theorem of the paper. The proof follows from an online-to-batch argument (Cesa-Bianchi et al., 2002; Cesa-Bianchi and Lugosi, 2006) applied on the regret bound obtained in Theorem 4. The following theorem, which is known as the online-to-batch conversion theorem, can be found in, e.g., Hazan et al. (2016).

Theorem 7 *(Hazan et al. (2016), Theorem 9.3) Let $\{\mathbf{x}_t\}_{t=1}^T$ be a sequence of decisions with regret bounded by $Regret_T$, such that $\bar{\mathbf{x}} = \frac{1}{T}\sum_t \mathbf{x}_t \in \mathcal{K}$, and let $f_t \sim \mathcal{D}$ be convex loss functions chosen i.i.d. from $\mathcal{D}$. Then, for $\delta < \frac{1}{2}$, w.p. at least $1-\delta$,*

$$\mathop{\mathbf{E}}_{f\sim D}(f(\bar{\mathbf{x}})) \leq \min_{\mathbf{x}\in\mathcal{K}} \mathop{\mathbf{E}}_{f\sim D}(f(\mathbf{x})) + \frac{Regret_T}{T} + \sqrt{\frac{8\log\frac{2}{\delta}}{T}}\,.$$

We can now apply this theorem with the decision set being all matrices B that are in $\mathcal{K} = \{B : B \succeq 0, \|B\|_{\text{tr}} = 1\}$, and convex loss functions being $\mathcal{G}_\lambda(B;(\mathcal{X},\mathcal{Y}))$, for $(\mathcal{X},\mathcal{Y}) \sim \mathcal{D}$ chosen at random from the underlying distribution $\mathcal{D}$ over tasks. We get that with probability at least $1-\delta$,

$$\mathcal{G}_\lambda(\bar{B},\mathcal{D}) \leq \min_{B\in\mathcal{K}} \mathcal{G}_\lambda(B,\mathcal{D}) + O\left(\frac{\sqrt{\log\frac{1}{\delta}}}{\lambda T^{1/4}}\right).$$

The only thing we have to take care of is the bound on the rank of the matrix M. However, note that $\bar{B} = M_{\mathcal{S}} M_{\mathcal{S}}^\top$ is in the convex hull of the $|\mathcal{S}|$ 1-rank matrices $v_t v_t^\top$. This implies that the rank of $\bar{B}$ is at most $|\mathcal{S}|$ which in turn implies the same on $M_{\mathcal{S}}$. Furthermore, this implies $\|B\|_{\text{tr}} = 1$, since the singular values of $M_{\mathcal{S}}$ are such that $\|M_{\mathcal{S}}\|_F^2 = \|B\|_{\text{tr}} = 1$. Therefore $M_{\mathcal{S}} \in \mathcal{F}_{|\mathcal{S}|}^{\text{low}}$. Let $B^* := \text{argmin}_{B\in\mathcal{K}} \mathcal{G}_\lambda(B,\mathcal{D})$, and consider M^* such that $B^* = M^* M^{*\top}$. Then, since by definition $\mathcal{G}_\lambda(\bar{B},\mathcal{D}) = \mathcal{L}_\lambda(M_{\mathcal{S}},\mathcal{D})$ and $\mathcal{G}_\lambda(B^*,\mathcal{D}) = \mathcal{L}_\lambda(M^*,\mathcal{D})$, we have

$$\mathcal{L}_\lambda(M_{\mathcal{S}},\mathcal{D}) \leq \min_{M\in\mathcal{F}} \mathcal{L}_\lambda(M,\mathcal{D}) + O\left(\frac{\sqrt{\log\frac{1}{\delta}}}{\lambda T^{1/4}}\right).$$

3.4. Proof of Theorem 2

Having dealt with the guarantees for the online Frank-Wolfe algorithm, we now move on to the setting of matrix multiplicative weights. We begin by showing the following regret bound for Algorithm 2.

Theorem 8 *Assume $T \geq \log(d)$. Let $\mathcal{S} = \{(\mathcal{X}_t,\mathcal{Y}_t)\}_{t=1}^T$ be a stochastic sequence of empirical tasks, and let B_t be defined by the output of Algorithm 2 applied to $\mathcal{S}$. Then, for $\mathcal{K} = \{B : B \succeq 0, \|B\|_{\text{tr}} = 1\}$, we have*

$$\sum_{t=1}^T \mathcal{G}_\lambda(B_t,(\mathcal{X}_t,\mathcal{Y}_t)) - \min_{B\in\mathcal{K}} \sum_{t=1}^T \mathcal{G}_\lambda(B,(\mathcal{X}_t,\mathcal{Y}_t)) \leq O\left(\frac{\sqrt{\log(d)T}}{\lambda}\right). \tag{7}$$

Proof As was the case in proving Theorem 1, we first need to establish the relevant regret bounds attained when using matrix multiplicative weights as our black-box algorithm.

Theorem 9 *(Tsuda et al. (2005); Arora et al. (2012), Theorem 5.1) Let $\{P_t\}_{t=1}^T$ be an arbitrary sequence of cost matrices, let $\mathcal{K} = \{B : B \succeq 0, \|B\|_{\text{tr}} = 1\}$, let $f_t(B) = B \bullet P_t$, and let $\{B_t\}_{t=1}^T$ be the matrices played by the matrix multiplicative weights algorithm. Further, suppose η is chosen such that $\eta\|P_t\|_{\text{sp}} \leq 1$ for all t, where $\|\cdot\|_{\text{sp}}$ is the spectral norm. Then, we have that*

$$\sum_{t=1}^T f_t(B_t) - \min_{B\in\mathcal{K}} \sum_{t=1}^T f_t(B) \leq \eta \sum_{t=1}^T B_t \bullet P_t^2 + \frac{\log(d)}{\eta} .$$

Note that we may apply the algorithm since we are optimizing over the convex set $\mathcal{K} = \{B : B \succeq 0, \|B\|_{\text{tr}} = 1\}$. The computation of the gradient is done in the exact same manner as before, and the gradients are treated as the cost matrices in the algorithm. Furthermore, by setting $\eta = \frac{\lambda}{2}\sqrt{\frac{\log(d)}{T}}$, we ensure that $\eta\|P_t\|_{\text{sp}} \leq 1$ since $\|P_t\|_{\text{sp}} \leq \|P_t\|_F \leq \frac{2}{\lambda}$ by Lemma 5, and $T \geq \log(d)$ by assumption. In addition, it follows from Lemma 5 and the fact that $B_t \in \mathcal{K}$ that

$$B_t \bullet P_t^2 \leq \|B_t\|_{\text{tr}} \cdot \|P_t^2\|_{\text{sp}} \leq \|B_t\|_{\text{tr}} \cdot \|P_t\|_F^2 \leq \frac{4}{\lambda^2} .$$

Taken together with our choice of η, we have that

$$\eta \sum_{t=1}^T B_t \bullet P_t^2 + \frac{\log(d)}{\eta} \leq \frac{4\eta T}{\lambda^2} + \frac{\log(d)}{\eta} = \frac{4\sqrt{\log(d)T}}{\lambda} ,$$

which completes the proof of Theorem 8. ■

We may observe that the regret bound achieved by the algorithm improves upon that of online Frank-Wolfe by a factor of $T^{1/4}$, thus leading to an improved sample complexity in the final theorem. Additionally, an important observation in our particular setting is that $P_t = \nabla f_t(B_t) = -\frac{\lambda}{2}\mathcal{X}_t^\top \boldsymbol{\alpha}_t \boldsymbol{\alpha}_t^\top \mathcal{X}_t$ is a rank-one matrix. Thus, Algorithm 2 can be seen to naturally enforce a low-rank structure of the output solution, as the rank of $\bar{B}$ is again at most $|\mathcal{S}|$, and so $M_{\mathcal{S}} \in \mathcal{F}^{\text{low}}_{|\mathcal{S}|}$. The sample complexity advantage comes at the cost of an eigendecomposition for each iteration of the algorithm, although this can still be done in polynomial time. A guarantee on the low-rank structure of the solution cannot hold for, e.g., projected online gradient descent, as any projection step could destroy the low-rank structure.

To complete the proof of Theorem 2, we can again apply Theorem 7, using the regret bound derived in Theorem 8. Consider the set of convex loss functions as $\mathcal{G}_\lambda(B; (\mathcal{X}, \mathcal{Y}))$, for $(\mathcal{X}, \mathcal{Y}) \sim \mathcal{D}$. Then, we have that with probability at least $1 - \delta$,

$$\mathcal{G}_\lambda(\bar{B}, \mathcal{D}) \leq \min_{B\in\mathcal{K}} \mathcal{G}_\lambda(B, \mathcal{D}) + O\left(\frac{\sqrt{\log(d)} + \lambda\sqrt{\log\left(\frac{1}{\delta}\right)}}{\lambda\sqrt{T}}\right) .$$

Following the same reasoning as before, where we let $M_{\mathcal{S}}$ be such that $\bar{B} = M_{\mathcal{S}} M_{\mathcal{S}}^\top$, we may conclude that

$$\mathcal{L}_\lambda(M_{\mathcal{S}}, \mathcal{D}) \leq \min_{M\in\mathcal{F}} \mathcal{L}_\lambda(M, \mathcal{D}) + O\left(\frac{\sqrt{\log(d)} + \lambda\sqrt{\log\left(\frac{1}{\delta}\right)}}{\lambda\sqrt{T}}\right) .$$

4. Discussion

We have presented efficient algorithms for linear representation learning that provably generalize, using a methodology that stems from online learning and convex relaxation. This allows us to simultaneously derive efficient algorithms and prove generalization error bounds that follow from the regret guarantees.

Our model assumes a learner who is presented with tasks arriving online and attempts to find a linear representation that would minimize the classification error on future tasks. An important aspect of our work is that we measure the loss of the representation in terms of the *empirical error* of the learner. This allows us to decouple the generalization error of the learner from the generalization error of the sub-learners. In other words, when we discuss the generalization abilities of our algorithm, we do not need to worry about the generalization of the sub-learners.

The generalization performance of the sub-learners is then susceptible to the classical analysis of learning algorithms, and would depend on the sample size of the problems we observe. To summarize, we guarantee that over the new embedded space, the empirical error of future sub-learners will be small. Their generalization error would then depend on the expected sample size and the complexity of the representation we have learned.

Because we could reduce our task to an online convex optimization task, our method outputs a representation that generalizes toward future tasks. An interesting direction would be to explore generalization by uniform convergence for linear representations. This seems elusive due to the inherent double-optimization, for both the representation learning and the classification over tasks.

In addition, dimensionality reduction, which can be achieved in the case of both algorithms, is especially useful when one is faced with a novel classification task with little data. Otherwise, data from other problems would not be necessary and one could perform representation learning together with classification on data from the problem at hand.

References

Yonatan Amit, Michael Fink, Nathan Srebro, and Shimon Ullman. Uncovering shared structures in multiclass classification. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, pages 17–24. ACM, 2007.

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

Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing (STOC)*, pages 227–236. ACM, 2007.

Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A simple but tough-to-beat baseline for sentence embedding. In *International Conference on Learning Representations (ICLR)*, 2017.

Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. Efficient representations for lifelong learning and autoencoding. In *Conference on Learning Theory (COLT)*, pages 191–210, 2015.

Jonathan Baxter et al. A model of inductive bias learning. *J. Artif. Intell. Res.(JAIR)*, 12(149-198):3, 2000.

Edwin V Bonilla, Kian Ming Adam Chai, and Christopher KI Williams. Multi-task gaussian process prediction. In *Advances in Neural Information Processing Systems (NIPS)*, volume 20, pages 153–160, 2007.

Rich Caruana. Multitask learning. *Mach. Learn.*, 28(1):41–75, July 1997. ISSN 0885-6125. doi: 10.1023/A:1007379606734. URL http://dx.doi.org/10.1023/A:1007379606734.

Nicolo Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

Nicoló Cesa-Bianchi, Alex Conconi, and Claudio Gentile. On the generalization ability of on-line learning algorithms. *Advances in Neural Information Processing Systems (NIPS)*, 1:359–366, 2002.

Dan Garber and Elad Hazan. Playing non-linear games with linear oracles. In *Proceedings of the 2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 420–428. IEEE Computer Society, 2013.

Shengbo Guo, Onno Zoeter, and Cédric Archambeau. Sparse bayesian multi-task learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1755–1763, 2011.

Zaid Harchaoui, Matthijs Douze, Mattis Paulin, Miroslav Dudik, and Jérôme Malick. Large-scale image classification with trace-norm regularization. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3386–3393. IEEE, 2012.

Elad Hazan and Satyen Kale. Projection-free online learning. *Proceedings of the 29th International Conference on Machine Learning (ICML)*, 2012.

Elad Hazan and Tengyu Ma. A non-generative framework and convex relaxations for unsupervised learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

Elad Hazan et al. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

Daniel Hernández-Lobato, José Miguel Hernández-Lobato, Thibault Helleputte, and Pierre Dupont. Expectation propagation for bayesian multi-task feature selection. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 522–537. Springer, 2010.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1097–1105, 2012.

Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, volume 14, pages 1188–1196, 2014.

Andreas Maurer and Massimiliano Pontil. Excess risk bounds for multitask learning with trace norm regularization. In *Conference on Learning Theory (COLT)*, volume 30, pages 55–76, 2013.

Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning. *The Journal of Machine Learning Research*, 17(1):2853–2884, 2016.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3111–3119, 2013.

Nanyun Peng and Mark Dredze. Named entity recognition for chinese social media with jointly trained embeddings. In *EMNLP*, pages 548–554, 2015.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

Irina Rish, Genady Grabarnik, Guillermo Cecchi, Francisco Pereira, and Geoffrey J. Gordon. Closed-form supervised dimensionality reduction with generalized linear models. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 832–839, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-205-4. doi: 10.1145/1390156.1390261. URL http://doi.acm.org/10.1145/1390156.1390261.

Anton Schwaighofer, Volker Tresp, and Kai Yu. Learning gaussian process kernels via hierarchical bayes. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1209–1216, 2004.

Shai Shalev-Shwartz, Yoram Singer, and Nathan Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, pages 807–814. ACM, 2007.

Karthik Sridharan, Shai Shalev-Shwartz, and Nathan Srebro. Fast rates for regularized objectives. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1545–1552, 2009.

Sebastian Thrun and Lorien Pratt, editors. *Learning to Learn*. Kluwer Academic Publishers, Norwell, MA, USA, 1998. ISBN 0-7923-8047-9.

Koji Tsuda, Gunnar Rätsch, and Manfred K Warmuth. Matrix exponentiated gradient updates for on-line learning and bregman projection. *Journal of Machine Learning Research*, 6(Jun): 995–1018, 2005.

Manfred K Warmuth and Dima Kuzmin. Online variance minimization. *Machine Learning*, 87(1): 1–32, 2012.

Kai Yu, Volker Tresp, and Anton Schwaighofer. Learning gaussian processes from multiple tasks. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, pages 1012–1019. ACM, 2005.