



AFRL-AFOSR-UK-TR-2020-0015

TTESLA: Towards Tera-Scale Nonparametric Learning

**Lorenzo Rosasco
FONDAZIONE ISTITUTO ITALIANO DI TECNOLOGIA
VIA MOREGO 30
GENOVA, 16163
IT**

**05/29/2020
Final Report**

DISTRIBUTION A: Distribution approved for public release.

Air Force Research Laboratory
Air Force Office of Scientific Research
European Office of Aerospace Research and Development
Unit 4515 Box 14, APO AE 09421

REPORT DOCUMENTATION PAGE					Form Approved OMB No. 0704-0188	
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Executive Services, Directorate (0704-0188). Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.						
1. REPORT DATE (DD-MM-YYYY) 29-05-2020		2. REPORT TYPE Final		3. DATES COVERED (From - To) 15 Feb 2017 to 14 Feb 2020		
4. TITLE AND SUBTITLE TTESLA: Towards Tera-Scale Nonparametric Learning				5a. CONTRACT NUMBER		
				5b. GRANT NUMBER FA9550-17-1-0390		
				5c. PROGRAM ELEMENT NUMBER 61102F		
6. AUTHOR(S) Lorenzo Rosasco				5d. PROJECT NUMBER		
				5e. TASK NUMBER		
				5f. WORK UNIT NUMBER		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) FONDAZIONE ISTITUTO ITALIANO DI TECNOLOGIA VIA MOREGO 30 GENOVA, 16163 IT				8. PERFORMING ORGANIZATION REPORT NUMBER		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) EOARD Unit 4515 APO AE 09421-4515				10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/AFOSR IOE		
				11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-AFOSR-UK-TR-2020-0015		
12. DISTRIBUTION/AVAILABILITY STATEMENT A DISTRIBUTION UNLIMITED: PB Public Release						
13. SUPPLEMENTARY NOTES						
14. ABSTRACT The project has contributed to the development of optimal and efficient algorithms for large scale machine learning and their applications, with results in three main directions: 1) dealing with data having non Euclidean structure, 2) developing efficient approaches based on random projections, 3) application to the development of efficient AI systems for humanoid robotics. The results of the project have led to new theoretical results, new software and new intelligent systems for robotics. They have been published and presented in the top venues in the field. The publications are attached to the final report.						
15. SUBJECT TERMS Big Data, Non-Parametric, Machine Learning						
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT SAR	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON FRIEND, MARK	
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified			19b. TELEPHONE NUMBER (Include area code) 011-44-1895-616292	

“TTESLA: Towards Tera-Scale Nonparametric Learning ”

May, 15, 2020

Name of Principal Investigators (PI and Co-PIs): Lorenzo Rosasco

- e-mail address : lorenzo.rosasco@iit.it
- Institution : FONDAZIONE ISTITUTO ITALIANO DI TECNOLOGIA
- Mailing Address : Via Morego 30, Genoa, 16163, Italy
- Phone : +39 010 28961
- Fax : +39 010 28961

Period of Performance: 02.15.2017 – 02.15.2020

Abstract: The project has contributed to the development of optimal and efficient algorithms for large scale machine learning and their applications, with results in three main directions: 1) dealing with data having non Euclidean structure, 2) developing efficient approaches based on random projections, 3) application to the development of efficient AI systems for humanoid robotics. The results of the project have led to new theoretical results, new software and new intelligent systems for robotics. They have been published and presented in the top venues in the field.

Introduction: The project was related to processing and extraction of information from massive data-sets from diverse sources. In this context machine learning approaches have been crucial and provide the building block of modern artificial intelligent systems. Despite the successes of these systems, good performances come at the expenses of enormous computational costs, which are often not desirable or even affordable. Further, it is often assumed that data come in vectorial form. This becomes an important limitation since in practice data have often more complex structures. In this project we have tackled both this issues merging modelling and computational ideas to derive new machine learning algorithms as we discuss next.

Results and Discussion: We discuss research along three main directions and report below the publications where these results were presented.

- 1) *Learning manifold valued functions.* Supervised learning deal with the problem of estimating input/output relationships. When dealing with vector data, there are plenty of different way of modelling and estimating such relationships. In practice however data come in different shapes and form. For example, in multi-category classification problems, categories are often connected and describe by node in a graph or a tree. In estimating the dynamics, which is classical problem in robotics, outputs are points on a curve. As it turns out both the above problems can be seen as special cases of estimating a function with values on a manifold. As described in the first two publications in the list, we have developed and proposed a novel approach to deal with general manifold valued data and in particular for hyperbolic manifolds where tree structures can be naturally embedded. Unlike most previously proposed approaches our algorithms has optimal statistical guarantees. Experimental results, detailed in the papers, show promising performances.
- 2) *Efficient learning with fast optimization and random projections.* Much of recent work in machine learning has been devoted to numerically efficient solutions. However most works focus on the implications that algorithmic developments have on training rather than on test performances. Here we have taken a different perspectives and consider the effect of random projections and acceleration. On the one hand we have considered random projections based on data sampling and their fast implementation. On the other hand we have considered the test performance of accelerated optimization and its combination with random projections. The proposed approach provably achieves optimal performance with only a fraction of the computational resources previously

required. Beyond supervised learning we have started considering the extension of these ideas studying clustering and bandit optimization, showing similar properties.

- 3) Finally we have considered the practical impact of the proposed solutions in a robotic setting. Here, we have incorporated the proposed methods in a object detection pipeline which is one of the main elements of the robot perception. Compared to other approaches we were able to solve the problem in real time, and the system was indeed included in the robotic platform. The excellent practical performances were confirmed through extensive testing.

List of Publications and Significant Collaborations that resulted from your AOARD supported project: (attached)

1. A Rudi, C Ciliberto, GM Marconi, L Rosasco Manifold structured prediction Advances in Neural Information Processing Systems, 5610-5621
2. GM Marconi, C Ciliberto, L Rosasco Hyperbolic Manifold Regression to appear in Proceedings of International Conference on Artificial Intelligence and Statistics Advances in Neural Information Processing Systems, 5610-5621
3. L Carratino, A Rudi, L Rosasco Learning with sgd and random features Advances in Neural Information Processing Systems, 10192-10203
4. A Rudi, D Calandriello, L Carratino, L Rosasco On fast leverage score sampling and optimal learning Advances in Neural Information Processing Systems, 5672-5682
5. D Calandriello, L Rosasco Statistical and computational trade-offs in kernel k-means Advances in Neural Information Processing Systems, 9357-9367
6. D Calandriello, L Carratino, A Lazaric, M Valko, L Rosasco Gaussian process optimization with adaptive sketching: Scalable and no regret 32nd Annual Conference on Learning Theory
7. N Pagliana, L Rosasco Implicit Regularization of Accelerated Methods in Hilbert Spaces Advances in Neural Information Processing Systems, 14454-14464
8. E Maiettini, G Pasquale, L Rosasco, L Natale [Interactive data collection for deep learning object detectors on humanoid robots](#) 2017 IEEE-RAS 17th International Conference on Humanoid Robotics
9. E Maiettini, G Pasquale, L Rosasco, L Natale [Speeding-up Object Detection Training for Robotics with FALKON](#) 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems

Manifold Structured Prediction

Alessandro Rudi ^{•,1} **Carlo Ciliberto** ^{•,*,2} **Gian Maria Marconi** ³ **Lorenzo Rosasco** ^{3,4}

alessandro.rudi@inria.fr c.ciliberto@imperial.ac.uk gian.maria.marconi@iit.it lrosasco@mit.edu

¹INRIA - Département d'informatique, École Normale Supérieure - PSL Research University, Paris, France.

²Department of Electrical and Electronic Engineering, Imperial College, London, UK.

³Università degli studi di Genova & Istituto Italiano di Tecnologia, Genova, Italy.

⁴Massachusetts Institute of Technology, Cambridge, USA.

• Equal Contribution

Abstract

Structured prediction provides a general framework to deal with supervised problems where the outputs have semantically rich structure. While classical approaches consider finite, albeit potentially huge, output spaces, in this paper we discuss how structured prediction can be extended to a continuous scenario. Specifically, we study a structured prediction approach to manifold valued regression. We characterize a class of problems for which the considered approach is statistically consistent and study how geometric optimization can be used to compute the corresponding estimator. Promising experimental results on both simulated and real data complete our study.

1 Introduction

Regression and classification are probably the most classical machine learning problems and correspond to estimating a function with scalar and binary values, respectively. In practice, it is often interesting to estimate functions with more structured outputs. When the output space can be assumed to be a vector space, many ideas from regression can be extended, think for example to multivariate [20] or functional regression [32]. However, a lack of a natural vector structure is a feature of many practically interesting problems, such as ranking [18], quantile estimation [26] or graph prediction [38]. In this latter case, the outputs are typically provided only with some distance or similarity function that can be used to design appropriate loss function. Knowledge of the loss is sufficient to analyze an abstract empirical risk minimization approach within the framework of statistical learning theory, but deriving approaches that are at the same time statistically sound and computationally feasible is a key challenge. While ad-hoc solutions are available for many specific problems [15, 37, 24, 7], structured prediction [5] provides a unifying framework where a variety of problems can be tackled as special cases.

Classically, structured prediction considers problems with finite, albeit potentially huge, output spaces. In this paper, we study how these ideas can be applied to non-discrete output spaces. In particular, we consider the case where the output space is a Riemannian manifold, that is the problem of manifold structured prediction (also called manifold valued regression [46]). While also in this case ad-hoc methods are available [47], in this paper we adopt and study a structured prediction approach starting from a framework proposed in [13]. Within this framework, it is possible to derive a statistically sound, and yet computationally feasible, structured prediction approach as long as the loss function satisfies suitable structural assumptions [14, 17, 29, 25, 12, 36]. Moreover we can guarantee that the computed prediction is always an element of the manifold.

Our main technical contribution is a characterization of loss functions for manifold structured prediction satisfying such a structural assumption. In particular, we consider the case where the

*Work performed while C.C. was at the University College London.

Riemannian metric is chosen as a loss function. As a byproduct of these results, we derive a manifold structured learning algorithm that is universally consistent and corresponding finite sample bounds. From a computational point of view, the proposed algorithm requires solving a linear system (at training time) and a minimization problem over the output manifold (at test time). To tackle this latter problem, we investigate the application of geometric optimization methods, and in particular Riemannian gradient descent [1]. We consider both numerical simulations and benchmark datasets reporting promising performances. The rest of the paper is organized as follows. In Section 2, we define the problem and explain the proposed algorithm. In Section 3 we state and prove the theoretical results of this work. In Section 4 we explain how to compute the proposed algorithm and we show the performance of our method on synthetic and real data.

2 Structured Prediction for Manifold Valued Regression

The goal of supervised learning is to find a functional relation between an input space $\mathcal{X}$ and an output space $\mathcal{Y}$ given a finite set of observations. Traditionally, the output space is either a linear space (e.g. $\mathcal{Y} = \mathbb{R}^M$) or a discrete set (e.g. $\mathcal{Y} = \{0, 1\}$ in binary classification). In this paper, we consider the problem of manifold structured prediction [47], in which output data lies on a manifold $\mathcal{M} \subset \mathbb{R}^d$. In this context, statistical learning corresponds to solving

$$\operatorname{argmin}_{f \in \mathcal{X} \rightarrow \mathcal{Y}} \mathcal{E}(f) \quad \text{with} \quad \mathcal{E}(f) = \int_{\mathcal{X} \times \mathcal{Y}} \Delta(f(x), y) \rho(x, y) \quad (1)$$

where $\mathcal{Y}$ is a subset of the manifold $\mathcal{M}$ and ρ is an unknown distribution on $\mathcal{X} \times \mathcal{Y}$. Here, $\Delta: \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ is a loss function that measures prediction errors for points estimated on the manifold. The minimization is meant over the set of all measurable functions from $\mathcal{X}$ to $\mathcal{Y}$. The distribution is fixed but unknown and a learning algorithm seeks an estimator $\hat{f}: \mathcal{X} \rightarrow \mathcal{Y}$ that approximately solves Eq. (1), given a set of training points $(x_i, y_i)_{i=1}^n$ sampled independently from ρ .

A concrete example of loss function that we will consider in this paper is $\Delta = d^2$ the squared geodesic distance $d: \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ [27]. The geodesic distance is the natural metric on a Riemannian manifold (it corresponds to the Euclidean distance when $\mathcal{M} = \mathbb{R}^d$) and is a natural loss function in the context of manifold regression [46, 47, 19, 23, 21].

2.1 Manifold Valued Regression via Structured Prediction

In this paper we consider a structured prediction approach to manifold valued regression following ideas in [13]. Given a training set $(x_i, y_i)_{i=1}^n$, an estimator for problem Eq. (1) is defined by

$$\hat{f}(x) = \operatorname{argmin}_{y \in \mathcal{Y}} \sum_{i=1}^n \alpha_i(x) \Delta(y, y_i) \quad (2)$$

for any $x \in \mathcal{X}$. The coefficients $\alpha(x) = (\alpha_1(x), \dots, \alpha_n(x))^T \in \mathbb{R}^n$ are obtained solving a linear system for a problem akin to kernel ridge regression (see Section 2.2): given a positive definite kernel $k: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ [4] over $\mathcal{X}$, we have

$$\alpha(x) = (\alpha_1(x), \dots, \alpha_n(x))^T = (K + n\lambda I)^{-1} K_x \quad (3)$$

where $K \in \mathbb{R}^{n \times n}$ is the empirical kernel matrix with $K_{i,j} = k(x_i, x_j)$, and $K_x \in \mathbb{R}^n$ the vector whose i -th entry corresponds to $(K_x)_i = k(x, x_i)$. Here, $\lambda \in \mathbb{R}_+$ is a regularization parameter and $I \in \mathbb{R}^{n \times n}$ denotes identity matrix.

Computing the estimator in Eq. (2) can be divided into two steps. During a *training step* the score function $\alpha: \mathcal{X} \rightarrow \mathbb{R}^n$ is learned, while during the *prediction step*, the output $\hat{f}(x) \in \mathcal{Y}$ is estimated on a new test point $x \in \mathcal{X}$. This last step requires minimizing the linear combination of distances $\Delta(y, y_i)$ between a candidate $y \in \mathcal{Y}$ and the training outputs $(y_i)_{i=1}^n$, weighted by the corresponding scores $\alpha_i(x)$. Next, we recall the derivation of the above estimator following [13].

2.2 Derivation of the Proposed Estimator

The derivation of $\hat{f}$ in Eq. (2) is based on the following key structural assumption on the loss.

Definition 1 (Structure Encoding Loss Function (SELF)). *Let $\mathcal{Y}$ be a compact set. A function $\Delta : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ is a Structure Encoding Loss Function if there exist a separable Hilbert space $\mathcal{H}$, a continuous feature map $\psi : \mathcal{Y} \rightarrow \mathcal{H}$ and a continuous linear operator $V : \mathcal{H} \rightarrow \mathcal{H}$ such that for all $y, y' \in \mathcal{Y}$*

$$\Delta(y, y') = \langle \psi(y), V\psi(y') \rangle_{\mathcal{H}}. \quad (4)$$

Intuitively, the SELF definition requires a loss function to be “bi-linearizable” over the space $\mathcal{H}$. This is similar, but more general, than requiring the loss to be a kernel since it allows also to consider distances (which are not positive definite) or even non-symmetric loss functions. As observed in [13], a wide range of loss functions often used in machine learning are SELF. In Section 3 we study how the above assumption applies to manifold structured loss functions, including the squared geodesic distance.

We first recall how the estimator Eq. (2) can be obtained assuming Δ to be SELF. We begin by rewriting the expected risk in Eq. (1) as

$$\mathcal{E}(f) = \int_{\mathcal{X}} \left\langle \psi(f(x)), V \int_{\mathcal{Y}} \psi(y) d\rho(y|x) \right\rangle_{\mathcal{H}} d\rho(x) \quad (5)$$

where we have conditioned $\rho(y, x) = \rho(y|x)\rho_{\mathcal{X}}(x)$ and used the linearity of the integral and the inner product. Therefore, any function $f^* : \mathcal{X} \rightarrow \mathcal{Y}$ minimizing the above functional must satisfy the following condition

$$f^*(x) = \operatorname{argmin}_{y \in \mathcal{Y}} \langle \psi(y), Vg^*(x) \rangle_{\mathcal{H}} \quad \text{where} \quad g^*(x) = \int_{\mathcal{Y}} \psi(y) d\rho(y|x) \quad (6)$$

where we have introduced the function $g^* : \mathcal{X} \rightarrow \mathcal{H}$ that maps each point $x \in \mathcal{X}$ to the conditional expectation of $\psi(y)$ given x . However we cannot compute explicitly g^* , but noting that it minimizes the expected least squares error

$$\int \|\psi(y) - g(x)\|_{\mathcal{H}}^2 d\rho(x, y) \quad (7)$$

suggests that a least squares estimator can be considered. We first illustrate this idea for $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{H} = \mathbb{R}^k$. In this case we can consider a ridge regression estimator

$$\hat{g}(x) = \widehat{W}^\top x \quad \text{with} \quad \widehat{W} = \operatorname{argmin}_{W \in \mathbb{R}^{d \times k}} \frac{1}{n} \|XW - \psi(Y)\|_F^2 + \lambda \|W\|_F^2 \quad (8)$$

where $X = (x_1, \dots, x_n)^\top \in \mathbb{R}^{n \times d}$ and $\psi(Y) = (\psi(y_1), \dots, \psi(y_n))^\top \in \mathbb{R}^{n \times k}$ are the matrices whose i -th row correspond respectively to the training sample $x_i \in \mathcal{X}$ and the (mapped) training output $\psi(y_i) \in \mathcal{H}$. We have denoted $\|\cdot\|_F^2$ the squared Frobenius norm of a matrix, namely the sum of all its squared entries. The ridge regression solution can be obtained in closed form as $\widehat{W} = (X^\top X + n\lambda I)^{-1} X^\top \psi(Y)$. For any $x \in \mathcal{X}$ we have

$$\hat{g}(x) = \psi(Y)^\top X (X^\top X + n\lambda I)^{-1} x = \psi(Y)^\top \alpha(x) = \sum_{i=1}^n \alpha_i(x) \psi(y_i) \quad (9)$$

where we have introduced the coefficients $\alpha(x) = X(X^\top X + n\lambda I)^{-1} x \in \mathbb{R}^n$. By substituting $\hat{g}$ to g^* in Eq. (6) we have

$$\hat{f}(x) = \operatorname{argmin}_{y \in \mathcal{M}} \left\langle \psi(y), V \left(\sum_{i=1}^n \alpha_i(x) \psi(y_i) \right) \right\rangle_{\mathcal{H}} = \operatorname{argmin}_{y \in \mathcal{M}} \sum_{i=1}^n \alpha_i(x) \Delta(y, y_i) \quad (10)$$

where we have used the linearity of the sum and the inner product to move the coefficients α_i outside of the inner product. Since the loss is SELF, we then obtain $\langle \psi(y), V\psi(y_i) \rangle_{\mathcal{H}} = \Delta(y, y_i)$ for any y_i in the training set. This recovers the estimator $\hat{f}$ introduced in Eq. (2), as desired.

We end noting how the above idea can be extended. First, we can consider $\mathcal{X}$ to be a set and $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ a positive definite kernel. Then $\hat{g}$ can be computed by kernel ridge regression (see e.g. [43]) to obtain the scores $\alpha(x) = (K + n\lambda I)^{-1} Kx$, see Eq. (3). Second, the above discussion applies if $\mathcal{H}$ is infinite dimensional. Indeed, thanks to the SELF assumption, $\hat{f}$ does not depend on explicit knowledge of the space $\mathcal{H}$ but only on the loss function.

We next discuss the main results of the paper, showing that a large class of loss functions for manifold structured prediction are SELF. This will allow us to prove consistency and learning rates for the manifold structured estimator considered in this work.

3 Characterization of SELF Function on Manifolds

In this section we provide sufficient conditions for a wide class of functions on manifolds to satisfy the definition of SELF. A key example will be the case of the squared geodesic distance. To this end we will make the following assumptions on the manifold $\mathcal{M}$ and the output space $\mathcal{Y} \subseteq \mathcal{M}$ where the learning problem takes place.

Assumption 1. $\mathcal{M}$ is a complete d -dimensional smooth connected Riemannian manifold, without boundary, with Ricci curvature bounded below and positive injectivity radius.

The assumption above imposes basic regularity conditions on the output manifold. In particular we require the manifold to be locally diffeomorphic to $\mathbb{R}^d$ and that the tangent space of $\mathcal{M}$ at any $p \in \mathcal{M}$ varies smoothly with respect to p . This assumption avoids pathological manifolds and is satisfied for instance by any smooth compact manifold (e.g. the sphere, torus, etc.) [27]. Other notable examples are the statistical manifold (without boundary) [3] any open bounded sub-manifold of the cone of positive definite matrices, which is often studied in geometric optimization settings [1]. This assumption will be instrumental to guarantee the existence of a space of functions $\mathcal{H}$ on $\mathcal{M}$ rich enough to contain the squared geodesic distance.

Assumption 2. $\mathcal{Y}$ is a compact geodesically convex subset of the manifold $\mathcal{M}$.

A subset $\mathcal{Y}$ of a manifold is geodesically convex if for any two points in $\mathcal{Y}$ there exists one and only one minimizing geodesic curve connecting them. The effect of Asm. 2 is twofold. On one hand it guarantees a generalized notion of convexity for the space $\mathcal{Y}$ on which we will solve the optimization problem in Eq. (2). On the other hand it avoids the geodesic distance to have singularities on $\mathcal{Y}$ (which is key to our main result below). For a detailed introduction to most definitions and results reviewed in this section we refer the interested reader to standard references for differential and Riemannian geometry (see e.g. [27]). We are ready to prove the main result of this work.

Theorem 1 (Smooth Functions are SELF). *Let $\mathcal{M}$ satisfy Asm. 1 and $\mathcal{Y} \subseteq \mathcal{M}$ satisfy Asm. 2. Then, any smooth function $h : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ is SELF on $\mathcal{Y}$.*

Sketch of the proof (Thm. 1). The complete proof of Thm. 1 is reported in ???. The proof hinges around the following key steps:

Step 1 If there exists an RKHS $\mathcal{H}$ on $\mathcal{M}$, then any $h \in \mathcal{H} \otimes \mathcal{H}$ is SELF. Let $\mathcal{H}$ be a reproducing kernel Hilbert space (RKHS) [4] of functions on $\mathcal{M}$ with associated bounded kernel $k : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}$. Let $\mathcal{H} \otimes \mathcal{H}$ denote the RKHS of functions $h : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}$ with associated kernel $\bar{k}$ such that $\bar{k}((y, z), (y', z')) = k(y, y')k(z, z')$ for any $y, y', z, z' \in \mathcal{M}$. Let, $h : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}$ be such that $h \in \mathcal{H} \otimes \mathcal{H}$. Recall that $\mathcal{H} \otimes \mathcal{H}$ is isometric to the space of Hilbert-Schmidt operators from $\mathcal{H}$ to itself. Let $V_h : \mathcal{H} \rightarrow \mathcal{H}$ be the operator corresponding to h via such isometry. We show that the SELF definition is satisfied with $V = V_h$ and $\psi(y) = k(y, \cdot) \in \mathcal{H}$ for any $y \in \mathcal{M}$. In particular, we have $\|V\| \leq \|V\|_{\text{HS}} = \|h\|_{\mathcal{H} \otimes \mathcal{H}}$, with $\|V\|_{\text{HS}}$ denoting the Hilbert-Schmidt norm of V .

Step 2: Under Asm. 2, $C_0^\infty(\mathcal{M}) \otimes C_0^\infty(\mathcal{M})$ “contains” $C^\infty(\mathcal{Y} \times \mathcal{Y})$. If $\mathcal{Y}$ is compact and geodesically convex, then it is diffeomorphic to a compact set of $\mathbb{R}^d$. By using this fact, we prove that any function in $C^\infty(\mathcal{Y} \times \mathcal{Y})$, the space of smooth functions on $\mathcal{Y} \times \mathcal{Y}$, admits an extension in $C_0^\infty(\mathcal{M} \times \mathcal{M})$ the space of smooth functions on $\mathcal{M} \times \mathcal{M}$ vanishing at infinity (this is well defined since $\mathcal{M}$ is diffeomorphic to $\mathbb{R}^d$, see ?? in the supplementary material), and that $C_0^\infty(\mathcal{M} \times \mathcal{M}) = C_0^\infty(\mathcal{M}) \otimes C_0^\infty(\mathcal{M})$, with $\otimes$ the canonical topological tensor product [50].

Step 3: Under Asm. 1, there exists an RKHS on $\mathcal{M}$ containing $C_0^\infty(\mathcal{M})$. Under Asm. 1, the Sobolev space $\mathcal{H} = H_s^2(\mathcal{M})$ of square integrable functions with smoothness s is an RKHS for any $s > d/2$ (see [22] for a definition of Sobolev spaces on Riemannian manifolds).

The proof proceeds as follows: from Step 1, we see that to guarantee h to be SELF it is sufficient to prove the existence of an RKHS $\mathcal{H}$ such that $h \in \mathcal{H} \otimes \mathcal{H}$. The rest of the proof is therefore devoted to showing that for smooth functions this is satisfied for $\mathcal{H} = H_s^2(\mathcal{M})$. Since h is smooth, by Step 2 we have that under Asm. 2, there exists a $\bar{h} \in C_0^\infty(\mathcal{M}) \otimes C_0^\infty(\mathcal{M})$ whose restriction $\bar{h}|_{\mathcal{Y} \times \mathcal{Y}}$ to $\mathcal{Y} \times \mathcal{Y}$ corresponds to h . Now, denote by $H_s^2(\mathcal{M})$ the Sobolev space of squared integrable functions on $\mathcal{M}$ with smoothness index $s > 0$. By construction, (see [22]) for any $s > 0$, we have $C_0^\infty(\mathcal{M})|_{\mathcal{Y}} \subseteq H_s^2(\mathcal{M})|_{\mathcal{Y}}$, namely for any function. In particular, $\bar{h} \in C_0^\infty(\mathcal{M}) \otimes C_0^\infty(\mathcal{M}) \subseteq H_s^2(\mathcal{M}) \otimes H_s^2(\mathcal{M})$. Finally, Step 3 guarantees that under Asm. 1, $\mathcal{H} = H_s^2(\mathcal{M})$ with $s > d/2$ is an RKHS, showing that $h \in \mathcal{H} \otimes \mathcal{H}$ as desired. $\square$

Interestingly, Thm. 1 shows that the SELF estimator proposed in Eq. (2) can tackle *any* manifold valued learning problem in the form of Eq. (1) with smooth loss function. In the following we study the specific case of the squared geodesic distance.

Theorem 2 (d^2 is SELF). *Let $\mathcal{M}$ satisfy Asm. 1 and $\mathcal{Y} \subseteq \mathcal{M}$ satisfy Asm. 2. Then, the squared geodesic distance $\Delta = d^2 : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}$ is smooth on $\mathcal{Y}$. Therefore Δ is SELF on $\mathcal{Y}$.*

The proof of the result above is reported in the supplementary material. The main technical aspect is to show that regularity provided by Asm. 2 guarantees the squared geodesic distance to be smooth. The fact that Δ is SELF is then an immediate corollary of Thm. 1.

3.1 Statistical Properties of Manifold Structured Prediction

In this section, we characterize the generalization properties of the manifold structured estimator Eq. (2) in light of Thm. 1 and Thm. 2.

Theorem 3 (Universal Consistency). *Let $\mathcal{M}$ satisfy Asm. 1 and $\mathcal{Y} \subseteq \mathcal{M}$ satisfy Asm. 2. Let $\mathcal{X}$ be a compact set and $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ be a bounded continuous universal kernel². For any $n \in \mathbb{N}$ and any distribution ρ on $\mathcal{X} \times \mathcal{Y}$ let $\hat{f}_n : \mathcal{X} \rightarrow \mathcal{Y}$ be the manifold structured estimator in Eq. (2) for a learning problem with smooth loss function Δ , with $(x_i, y_i)_{i=1}^n$ training points independently sampled from ρ and $\lambda_n = n^{-1/4}$. Then*

$$\lim_{n \rightarrow \infty} \mathcal{E}(\hat{f}_n) = \mathcal{E}(f^*) \quad \text{with probability 1.} \quad (11)$$

The result above follows from Thm. 4 in [13] combined with our result in Thm. 1. It guarantees that the algorithm considered in this work finds a consistent estimator for the manifold structured problem, when the loss function is smooth (thus also in the case of the squared geodesic distance). As it is standard in statistical learning theory, we can impose regularity conditions on the learning problem, in order to derive also generalization bounds for $\hat{f}$. In particular, if we denote by $\mathcal{F}$ the RKHS associated to the kernel k , we will require g^* to belong to the same space $\mathcal{H} \otimes \mathcal{F}$ where the estimator $\hat{g}$ introduced in Eq. (9) is learned. In the simplified case discussed in Section 2.2, with linear kernel on $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{H} = \mathbb{R}^k$ finite dimensional, we have $\mathcal{F} = \mathbb{R}^d$ and this assumption corresponds to require the existence of a matrix $W_*^\top \in \mathbb{R}^{k \times d} = \mathcal{H} \otimes \mathcal{F}$, such that $g^*(x) = W_*^\top x$ for any $x \in \mathcal{X}$. In the general case, the space $\mathcal{H} \otimes \mathcal{F}$ extends to the notion of *reproducing kernel Hilbert space for vector-valued functions* (see e.g. [30, 2]) but the same intuition applies [28, 10, 49].

Theorem 4 (Generalization Bounds). *Let $\mathcal{M}$ satisfy Asm. 1 and $\mathcal{Y} \subseteq \mathcal{M}$ satisfy Asm. 2. Let $\mathcal{H} = H_s^2(\mathcal{M})$ with $s > d/2$ and $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ be a bounded continuous reproducing kernel with associated RKHS $\mathcal{F}$. For any $n \in \mathbb{N}$, let $\hat{f}_n$ denote the manifold structured estimator in Eq. (2) for a learning problem with smooth loss $\Delta : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ and $\lambda_n = n^{-1/2}$. If the conditional mean g^* belongs to $\mathcal{H} \otimes \mathcal{F}$, then*

$$\mathcal{E}(\hat{f}_n) - \mathcal{E}(f^*) \leq c_\Delta q \tau^2 n^{-\frac{1}{4}} \quad (12)$$

holds with probability not less than $1 - 8e^{-\tau}$ for any $\tau > 0$, with $c_\Delta = \|\Delta\|_{\mathcal{H} \otimes \mathcal{H}}$ and q a constant not depending on n, τ or the loss Δ .

The generalization bound of Thm. 4 is obtained by adapting Thm. 5 of [13] to our results in Thm. 1 as detailed in the supplementary material. To our knowledge these are the first results characterizing in such generality the generalization properties of an estimator for manifold structured learning with generic smooth loss function. We conclude with a remark on a key quantity in the bound of Thm. 4.

Remark 1 (The constant c_Δ). *We comment on the role played in the learning rate by c_Δ , the norm of the loss function Δ seen as an element of the Hilbert space $\mathcal{H} \otimes \mathcal{H}$. Indeed, from the discussion of Thm. 1 we have seen that any smooth function on $\mathcal{Y}$ is SELF and belongs to the set $\mathcal{H} \otimes \mathcal{H}$ with $\mathcal{H} = H_s^2(\mathcal{M})$, the Sobolev space of squared integrable functions for $s > d/2$. Following this interpretation, we see that the bound in Thm. 4 can improve significantly (in terms of the constants) depending on the regularity of the loss function: smoother loss functions will result in “simpler” learning problems and vice-versa. In particular, when Δ corresponds to the squared geodesic distance, the more “regular” is the manifold $\mathcal{M}$, the learning problem will be. A refined quantitative characterization of c_Δ in terms of the Ricci curvature and the injective radius of the manifold is left to future work.*

²This is standard assumption for universal consistency (see [48]). An example of continuous universal kernel on $\mathcal{X} = \mathbb{R}^d$ is the Gaussian $k(x, x') = \exp(-\|x - x'\|^2/\sigma)$, for $\sigma > 0$.

Table 1: Structured loss, gradient of the structured loss and retraction for P_{++}^m and S_{d-1} . $Z_i \in P_{++}^m$ and $z_i \in S_{d-1}$ are the training set points. $I \in \mathbb{R}^{d \times d}$ is the identity matrix.

	Positive definite matrix manifold (P_{++}^m)	Sphere (S_{d-1})
$F(y)$	$\sum_{i=1}^n \alpha_i \ \log(Y^{-\frac{1}{2}} Z_i Y^{-\frac{1}{2}})\ _F^2$	$\sum_{i=1}^n \alpha_i \arccos(\langle z_i, y \rangle)^2$
$\nabla_{\mathcal{M}} F(y)$	$2 \sum_{i=1}^n \alpha_i Y^{\frac{1}{2}} \log(Y^{\frac{1}{2}} Z_i^{-1} Y^{\frac{1}{2}}) Y^{\frac{1}{2}}$	$4 \sum_{i=1}^n \alpha_i (yy^T - I) \frac{\arccos(\langle z_i, y \rangle)}{\sqrt{1 - \langle z_i, y \rangle^2}} z_i$
$R_y(v)$	$Y^{\frac{1}{2}} \exp(Y^{-\frac{1}{2}} v Y^{-\frac{1}{2}}) Y^{\frac{1}{2}}$	$\frac{v}{\ v\ }$

4 Manifold Structured Prediction Algorithm and Experiments

In this section we recall geometric optimization algorithms that can be adopted to perform the estimation of $\hat{f}$ on a novel test point x . We then evaluate the performance of the proposed method in practice, reporting numerical results on simulated and real data.

The algorithm presented in this paper consists in two steps. In the *training* phase, the matrix $C = (K + \lambda n I)^{-1}$ is computed, requiring a computational cost of essentially $O(n^3)$ in time and $O(n^2)$ in space (see Eq. (3)). In the *inference* phase, given a test point x , the coefficients in Eq. (3) are computed, $\alpha(x) := Cv(x)$, requiring essentially $O(n)$ in time, and then the optimization problem in Eq. (2) is solved (see next subsection for more details). Note that it is possible to reduce the computational complexity of the training and evaluation of the coefficients, while retaining the statistical guarantees of the proposed method. Indeed the computation of the coefficients consists essentially in solving a kernel ridge regression problem [10] as analyzed in [13], for which methods based on random projection, as *Nystrom* [44] or *random features* [39], reduce the complexity up to $O(n\sqrt{n})$ in time and $O(n)$ in space, while guaranteeing the same statistical properties [40, 42, 9, 11, 41].

4.1 Optimization on Manifolds

We begin discussing the computational aspects related to evaluating the manifold structured estimator. In particular, we discuss how to address the optimization problem in Eq. (2) in specific settings. Given a test point $x \in \mathcal{X}$, this process consists in solving a minimization over $\mathcal{Y}$, namely

$$\min_{y \in \mathcal{Y}} F(y) \quad (13)$$

where $F(y)$ corresponds to the linear combination of $\Delta(y, y_i)$ weighted by the scores $\alpha_i(x)$ computed according to Eq. (3). If $\mathcal{Y}$ is a linear manifold or a subset of $\mathcal{M} = \mathbb{R}^d$, this problem can be solved by means of gradient-based minimization algorithms, such as Gradient Descent (GD):

$$y_{t+1} = y_t - \eta_t \nabla F(y_t) \quad (14)$$

for a step size $\eta_t \in \mathbb{R}$. This algorithm can be extended to *Riemannian gradient descent* (RGD) [52] on manifolds, as

$$y_{t+1} = \text{Exp}_{y_t}(\eta_t \nabla_{\mathcal{M}} F(y_t)) \quad (15)$$

Where $\nabla_{\mathcal{M}} F$ is the gradient defined with respect to the Riemannian metric (see [1]) and $\text{Exp}_y : T_y \mathcal{M} \rightarrow \mathcal{M}$ denotes the exponential map on $y \in \mathcal{Y}$, mapping a vector from the tangent space $T_y \mathcal{M}$ to the associated point on the manifold according to the Riemannian metric [27]. For completeness, the algorithm is recalled in ?? . For this family of gradient-based algorithms it is possible to substitute the exponential map with a retraction $R_y : T_y \mathcal{M} \rightarrow \mathcal{M}$, which is a first order approximation to the exponential map. Retractions are often faster to compute and still offer convergence guarantees [1]. In the following experiments we will use both retractions and exponential maps. We mention that the step size η_t can be found with a line search over the validation set, for more see [1]. Note that also stochastic optimization algorithms have been generalized to the Riemannian setting such as R-SGD [8]. Interestingly, methods such as R-SGD can be advantageous in our setting. Indeed, solving the inference in Eq. (2) requires solving the minimization of a sum over n elements, it might be favorable from the computational perspective to adopt a stochastic method that at each iteration minimizes the functional with respect to a single (or a mini-batch) of randomly sampled elements of the entire sum.

Table 2: Simulation experiment: average squared loss (First two columns) and Δ_{PD} (Last two columns) error of the proposed structured prediction (SP) approach and the KRLS baseline on learning the inverse of a PD matrix for increasing matrix dimension.

Dim	Squared loss		Δ_{PD} loss	
	KRLS	SP	KRLS	SP
5	0.72 $\pm$ 0.08	0.89 $\pm$ 0.08	111 $\pm$ 64	0.94 $\pm$ 0.06
10	0.81 $\pm$ 0.03	0.92 $\pm$ 0.05	44 $\pm$ 8.3	1.24 $\pm$ 0.06
15	0.83 $\pm$ 0.03	0.91 $\pm$ 0.06	56 $\pm$ 10	1.25 $\pm$ 0.05
20	0.85 $\pm$ 0.02	0.91 $\pm$ 0.03	59 $\pm$ 12	1.33 $\pm$ 0.03
25	0.87 $\pm$ 0.01	0.91 $\pm$ 0.02	72 $\pm$ 9	1.44 $\pm$ 0.03
30	0.88 $\pm$ 0.01	0.91 $\pm$ 0.02	67 $\pm$ 7.2	1.55 $\pm$ 0.03

Table 1 reports gradients and retraction maps for the geodesic distance of two problems of interest considered in this work: positive definite manifold and the sphere. See Sections 4.2 and 4.3 for more details on the related manifolds.

We point out that using optimization algorithms that comply with the geometry of the manifold, such as RGD, guarantees that the computed value is an element of the manifold. This is in contrast with algorithms that compute a solution in a linear space that contains $\mathcal{M}$ and then need to project the computed solution onto $\mathcal{M}$. We next discuss empirical evaluations of the proposed manifold structured estimator on both synthetic and real datasets.

4.2 Synthetic Experiments: Learning Positive Definite Matrices

We consider the problem of learning a function $f : \mathbb{R}^d \rightarrow \mathcal{Y} = P_{++}^m$, where P_{++}^m denotes the *cone of positive definite (PD) $m \times m$ matrices*. Note that P_{++}^m is a manifold with squared geodesic distance Δ_{PD} between any two PD matrices $Z, Y \in P_{++}^m$ defined as

$$\Delta_{PD}(Z, Y) = \|\log(Y^{-\frac{1}{2}} Z Y^{-\frac{1}{2}})\|_F^2 \quad (16)$$

where, for any $M \in P_{++}^m$, we have that $M^{\frac{1}{2}}$ and $\log(M)$ correspond to the matrices with same eigenvectors of M but with respectively the square root and logarithm of the eigenvalues of M . In Table 1 we show the computation of the structured loss, the gradient of the structured loss and the exponential map of the PD matrix manifold. We refer the reader to [31, 6] for a more detailed introduction on the manifold of positive definite matrices.

For the experiments reported in the following we compared the performance of the manifold structured estimator minimizing the loss Δ_{PD} and a Kernel Regularized Least Squares classifier (KRLS) baseline (see Appendix ??), both trained using the Gaussian kernel $k(x, x') = \exp(-\|x - x'\|^2 / 2\sigma^2)$. The matrices predicted by the KRLS estimator are projected on the PD manifold by setting to a small positive constant ($1e - 12$) the negative eigenvalues. For the manifold structured estimator, the optimization problem at Eq. (2) was performed with the Riemannian Gradient Descent (RGD) algorithm [1]. We refer to [52] regarding the implementation of the RGD in the case of the geodesic distance on the PD cone.

Learning the Inverse of a Positive Definite Matrix. We consider the problem of learning the function $f : P_{++}^m \rightarrow P_{++}^m$ such that $f(X) = X^{-1}$ for any $X \in P_{++}^m$. Input matrices are generated as $X_i = U \Sigma U^\top \in P_{++}^m$ with U a random orthonormal matrix sampled from the Haar distribution [16] and $\Sigma \in P_{++}^m$ a diagonal matrix with entries randomly sampled from the uniform distribution on $[0, 10]$. We generated datasets of increasing dimension m from 5 to 50, each with 1000 points for training, 100 for validation and 100 for testing. The kernel bandwidth σ was chosen and the regularization parameter λ were selected by cross-validation respectively in the ranges 0.1 to 1000 and 10^{-6} to 1 (logarithmically spaced).

Table 2 reports the performance of the manifold structured estimator (SP) and the KRLS baseline with respect to both the Δ_{PD} loss and the least squares loss (normalized with respect to the number of dimensions). Note that the KRLS estimator target is to minimize the least squares (Frobenius) loss and is not designed to capture the geometry of the PD cone. We notice that the proposed approach significantly outperforms the KRLS baseline with respect to the Δ_{PD} loss. This is expected: Δ_{PD}

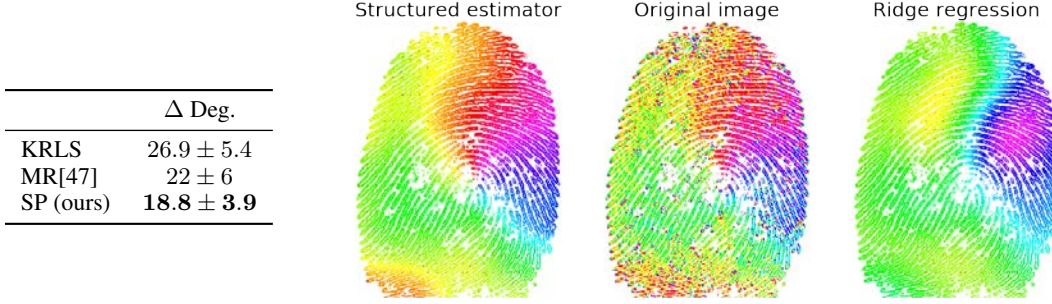


Figure 1: (Left) Fingerprints reconstruction: Average absolute error (in degrees) for the manifold structured estimator (SP), the manifold regression (MR) approach in [47] and the KRLS baseline. (Right) Fingerprint reconstruction of a single image where the structured predictor achieves 15.7 of average error while KRLS 25.3.

penalizes especially matrices with very different eigenvalues and our method cannot predict matrices with non-positive eigenvalues, as opposed to KRLS which computes a linear solution in $\mathbb{R}^{d^2}$ and then projects it onto the manifold. However the two methods perform comparably with respect to the squared loss. This is consistent with the fact that our estimator is aware of the natural structure of the output space and uses it profitably during learning.

4.3 Fingerprint Reconstruction

We consider the fingerprint reconstruction application in [47] in the context of manifold regression. Given a partial image of a fingerprint, the goal is to reconstruct the contour lines in output. Each fingerprint image is interpreted as a separate structured prediction problem where training input points correspond to the 2D position $x \in \mathbb{R}^2$ of valid contour lines and the output is the local orientation of the contour line, interpreted as a point on the circumference $\mathcal{S}_1$. The space $\mathcal{S}_1$ is a manifold with squared geodesic distance $\Delta_{\mathcal{S}_1}$ between two points $z, y \in \mathcal{S}_1$ corresponding to

$$\Delta_{\mathcal{S}_1}(z, y) = \arccos(\langle z, y \rangle)^2 \quad (17)$$

where $\arccos$ is the inverse cosine function. In Table 1 we show the computation of the structured loss, the gradient of the structured loss and the chosen retraction for the sphere manifold. We compared the performance of the manifold structured estimator proposed in this paper with the manifold regression approach in [47] on the FVC fingerprint verification challenge dataset³. The dataset consists of 48 fingerprint pictures, each with ~ 1400 points for training, ~ 1000 points for validation and the rest (~ 25000) for test.

Fig. 1 reports the average absolute error (in degrees) between the true contour orientation and the one estimated by our structured prediction approach (SP), the manifold regression (MR) in [47] and the KRLS baseline. Our method outperforms the MR competitor by a significant margin. As expected, the KRLS baseline is not able to capture the geometry of the output space and has a significantly larger error of the two other approaches. This is also observed on the qualitative plot in Fig. 1 (Left) where the predictions of our SP approach and the KRLS baseline are compared with the ground truth on a single fingerprint. Output orientations are reported for each pixel with a color depending on their orientation (from 0 to π). While the KRLS predictions are quite inconsistent, it can be noticed that our estimator is very accurate and even “smoother” than the ground truth.

4.4 Multilabel Classification on the Statistical Manifold

We evaluated our algorithm on multilabel prediction problems. In this context the output is an m -dimensional histogram, i.e. a discrete probability distribution over m points. We consider as manifold the space of probability distributions over m points, that is the m -dimensional simplex Δ^m endowed with the *Fisher information metric* [3]. We will consider $\mathcal{Y} = \Delta_\epsilon^m$ where we require $y_1, \dots, y_m \geq \epsilon$, for $\epsilon > 0$. In the experiment we considered $\epsilon = 1e-5$. The geodesic induced by the

³<http://bias.csr.unibo.it/fvc2004>

Table 3: Area under the curve (AUC) on multilabel benchmark datasets [51] for KRLS and SP.

	KRLS	SP (Ours)
Emotions	0.63	0.73
CAL500	0.92	0.92
Scene	0.62	0.73

Fisher metric is, $d(y, y') = \arccos(\sum_{i=1}^m \sqrt{y_i y'_i})$ [35]. This geodesic comes from applying the map $\pi: \Delta^m \rightarrow \mathcal{S}_{m-1}$, $\pi(y) = (\sqrt{y_1}, \dots, \sqrt{y_{m+1}})$ to the points $\{y_i\}_{i=1}^n \in \Delta^m$. This results in points that belong to the intersection of the positive quadrant $\mathbb{R}_{++}^m$ and the sphere $\mathcal{S}_{m-1}$. We can therefore use the geodesic distance on the Sphere and gradient and retraction map described in Table 1. We test our approach on some of the benchmark multilabel datasets described in [51] and we compare the results with the KRLS baseline. We cross-validate λ and σ taking values, respectively, from the intervals $[10^{-6}, 10^{-1}]$ and $[0.1, 10]$. We compute the area under curve (AUC) [45] metric to evaluate the quality of the predictions, results are shown in Table 3.

4.5 Additional Example

We conclude this section with a further relevant example of our proposed approach to the setting of applications to relational knowledge, which we plan to investigate in future work. In particular, we consider settings where the output space $\mathcal{M}$ corresponds to the Hyperbolic space (or Poincaré disk), which has recently been adopted by the knowledge representation community to learn embeddings of relational data to encode discrete semantic/hierarchical information [34, 33]. The embedding is such that symbolic objects (e.g. words, entities, concepts) with high semantic or functional similarity are mapped into points with small hyperbolic geodesic distance. Typically, learning the embedding is a time consuming process that requires training from scratch on the whole dataset whenever a new example is provided. To address this issue, with our manifold regression approach we could learn $f: X \rightarrow \mathcal{M}$ with $x \in X$ the observed entity and $f(x)$ its predicted embedding. This would allow to transfer the embedding learned using techniques such as those in [34] to new points, without retraining the entire system from scratch. Interestingly, our theory is applicable to this setting since $\mathcal{M}$ satisfies Asm. 1.

5 Conclusions

In this paper we studied a structured prediction approach for manifold valued learning problems. In particular we characterized a wide class of loss functions (including the geodesic distance) for which we proved the considered algorithm to be statistically consistent, additionally providing finite sample bounds under standard regularity assumptions. Our experiments show promising results on synthetic and real data using two common manifolds: the positive definite matrices cone and the sphere. With the latter we considered applications on fingerprint reconstruction and multi-labeling. The proposed method leads to some open questions. From a statistical point of view it is of interest how invariants of the manifold explicitly affect the learning rates, see Remark 1. From a more computational perspective, even if experimentally our algorithm achieves good results we did not investigate convergence guarantees in terms of optimization.

Acknowledgments.

A. R. acknowledges the support of the European Research Council (grant SEQUOIA 724063). L. R. acknowledges the support of the AFOSR projects FA9550-17-1-0390 and BAA-AFRL-AFOSR-2016-0007 (European Office of Aerospace Research and Development), and the EU H2020-MSCA-RISE project NoMADS - DLV-777826. The work of L. R. and G. M. M. is supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216, and the Italian Institute of Technology. We gratefully acknowledge the support of NVIDIA Corporation for the donation of the Titan Xp GPUs and the Tesla k40 GPU used for this research. This work was supported in part by EPSRC grant EP/P009069/1.

References

- [1] P-A Absil, Robert Mahony, and Rodolphe Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.
- [2] Mauricio A Alvarez, Lorenzo Rosasco, Neil D Lawrence, et al. Kernels for vector-valued functions: A review. *Foundations and Trends® in Machine Learning*, 4(3):195–266, 2012.
- [3] Shun-ichi Amari and Hiroshi Nagaoka. *Methods of information geometry*, volume 191. American Mathematical Soc., 2007.
- [4] Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404, 1950.
- [5] GH Bakir, T Hofmann, B Schölkopf, AJ Smola, B Taskar, and SVN Vishwanathan. Predicting structured data. *neural information processing*, 2007.
- [6] Rajendra Bhatia. *Positive definite matrices*. Princeton university press, 2009.
- [7] Veli Bicer, Thanh Tran, and Anna Gossen. Relational kernel machines for learning from graph-structured rdf data. In *Extended Semantic Web Conference*, pages 47–62. Springer, 2011.
- [8] Silvere Bonnabel. Stochastic gradient descent on riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, 2013.
- [9] Raffaello Camoriano, Tomás Angles, Alessandro Rudi, and Lorenzo Rosasco. Nytro: When subsampling meets early stopping. In *Artificial Intelligence and Statistics*, pages 1403–1411, 2016.
- [10] Andrea Caponnetto and Ernesto De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007.
- [11] Luigi Carratino, Alessandro Rudi, and Lorenzo Rosasco. Learning with sgd and random features. *Advances in Neural Information Processing Systems*, 2018.
- [12] Carlo Ciliberto, Francis Bach, and Alessandro Rudi. Localized structured prediction. *arXiv preprint arXiv:1806.02402*, 2018.
- [13] Carlo Ciliberto, Lorenzo Rosasco, and Alessandro Rudi. A consistent regularization approach for structured prediction. *Advances in Neural Information Processing Systems 29 (NIPS)*, pages 4412–4420, 2016.
- [14] Carlo Ciliberto, Alessandro Rudi, Lorenzo Rosasco, and Massimiliano Pontil. Consistent multitask learning with nonlinear output relations. In *Advances in Neural Information Processing Systems*, pages 1986–1996, 2017.
- [15] Harold Charles Daume and Daniel Marcu. *Practical structured learning techniques for natural language processing*. Citeseer, 2006.
- [16] Joe Diestel and Angela Spalsbury. *The joys of Haar measure*. American Mathematical Soc., 2014.
- [17] Moussab Djerrab, Alexandre Garcia, Maxime Sangnier, and Florence d’Alché Buc. Output fisher embedding regression. *Machine Learning*, pages 1–28, 2018.
- [18] John C Duchi, Lester W Mackey, and Michael I Jordan. On the consistency of ranking algorithms. In *ICML*, pages 327–334, 2010.
- [19] P Thomas Fletcher. Geodesic regression and the theory of least squares on riemannian manifolds. *International journal of computer vision*, 105(2):171–185, 2013.
- [20] Wolfgang Härdle and Léopold Simar. *Applied multivariate statistical analysis*, volume 22007. Springer, 2007.
- [21] Søren Hauberg, Oren Freifeld, and Michael J Black. A geometric take on metric learning. In *Advances in Neural Information Processing Systems*, pages 2024–2032, 2012.
- [22] Emmanuel Hebey. *Nonlinear analysis on manifolds: Sobolev spaces and inequalities*, volume 5. American Mathematical Soc., 2000.
- [23] Jacob Hinkle, Prasanna Muralidharan, P Thomas Fletcher, and Sarang Joshi. Polynomial regression on riemannian manifolds. In *European Conference on Computer Vision*, pages 1–14. Springer, 2012.

- [24] Mohammed Waleed Kadous and Claude Sammut. Classification of multivariate time series and structured data using constructive induction. *Machine learning*, 58(2):179–216, 2005.
- [25] Anna Korba, Alexandre Garcia, and Florence d’Alché Buc. A structured prediction approach for label ranking. *Advances in Neural Information Processing Systems*, 2018.
- [26] Quoc V Le, Tim Sears, and Alexander J Smola. Nonparametric quantile estimation. 2005.
- [27] John M Lee. Smooth manifolds. In *Introduction to Smooth Manifolds*, pages 1–29. Springer, 2003.
- [28] Junhong Lin, Alessandro Rudi, Lorenzo Rosasco, and Volkan Cevher. Optimal rates for spectral algorithms with least-squares regression over hilbert spaces. *Applied and Computational Harmonic Analysis*, 2018.
- [29] Giulia Luise, Alessandro Rudi, Massimiliano Pontil, and Carlo Ciliberto. Differential properties of sinkhorn approximation for learning with wasserstein distance. *Advances in Neural Information Processing Systems*, 2018.
- [30] Charles A Micchelli and Massimiliano Pontil. On learning vector-valued functions. *Neural computation*, 17(1):177–204, 2005.
- [31] Maher Moakher and Philipp G Batchelor. Symmetric positive-definite matrices: From geometry to applications and visualization. In *Visualization and Processing of Tensor Fields*, pages 285–298. Springer, 2006.
- [32] Jeffrey S Morris. Functional regression. *Annual Review of Statistics and Its Application*, 2:321–359, 2015.
- [33] Maximilian Nickel and Douwe Kiela. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. *International Conference on Machine Learning*, 2018.
- [34] Maximilian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in neural information processing systems*, pages 6338–6347, 2017.
- [35] Frank Nielsen and Ke Sun. Clustering in hilbert simplex geometry. *arXiv preprint arXiv:1704.00454*, 2017.
- [36] Alex Nowak-Vila, Francis Bach, and Alessandro Rudi. Sharp analysis of learning with discrete losses. *arXiv preprint arXiv:1810.06839*, 2018.
- [37] Sebastian Nowozin, Christoph H Lampert, et al. Structured learning and prediction in computer vision. *Foundations and Trends® in Computer Graphics and Vision*, 6(3–4):185–365, 2011.
- [38] Benjamin Paaßen, Christina Göpfert, and Barbara Hammer. Time series prediction for graphs in kernel and dissimilarity spaces. *Neural Processing Letters*, pages 1–21, 2017.
- [39] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.
- [40] Alessandro Rudi, Raffaello Camoriano, and Lorenzo Rosasco. Less is more: Nyström computational regularization. In *Advances in Neural Information Processing Systems*, pages 1657–1665, 2015.
- [41] Alessandro Rudi, Luigi Carratino, and Lorenzo Rosasco. Falkon: An optimal large scale kernel method. In *Advances in Neural Information Processing Systems*, pages 3888–3898, 2017.
- [42] Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. In *Advances in Neural Information Processing Systems*, pages 3215–3225, 2017.
- [43] John Shawe-Taylor and Nello Cristianini. *Kernel methods for pattern analysis*. Cambridge university press, 2004.
- [44] Alex J Smola and Bernhard Schölkopf. Sparse greedy matrix approximation for machine learning. 2000.
- [45] Ashwin Srinivasan. Note on the location of optimal classifiers in n-dimensional roc space. 1999.
- [46] Florian Steinke and Matthias Hein. Non-parametric regression between manifolds. In *Advances in Neural Information Processing Systems*, pages 1561–1568, 2009.
- [47] Florian Steinke, Matthias Hein, and Bernhard Schölkopf. Nonparametric regression between general riemannian manifolds. *SIAM Journal on Imaging Sciences*, 3(3):527–563, 2010.
- [48] Ingo Steinwart and Andreas Christmann. *Support vector machines*. Springer Science & Business Media, 2008.

- [49] Ingo Steinwart, Don R Hush, Clint Scovel, et al. Optimal rates for regularized least squares regression. In *COLT*, 2009.
- [50] François Trèves. *Topological Vector Spaces, Distributions and Kernels: Pure and Applied Mathematics*, volume 25. Elsevier, 2016.
- [51] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Mining multi-label data. In *Data mining and knowledge discovery handbook*, pages 667–685. Springer, 2009.
- [52] Hongyi Zhang and Suvrit Sra. First-order methods for geodesically convex optimization. In *Conference on Learning Theory*, pages 1617–1638, 2016.

Hyperbolic Manifold Regression

Gian Maria Marconi¹

Lorenzo Rosasco^{1,3,4}

Carlo Ciliberto²

¹Istituto Italiano di Tecnologia, Genoa, Italy ²Imperial College of London, London, UK

³Massachusetts Institute of Technology, MA, USA ⁴University of Genoa, Genoa, Italy

Abstract

Geometric representation learning has recently shown great promise in several machine learning settings, ranging from relational learning to language processing and generative models. In this work, we consider the problem of performing manifold-valued regression onto an hyperbolic space as an intermediate component for a number of relevant machine learning applications. In particular, by formulating the problem of predicting nodes of a tree as a manifold regression task in the hyperbolic space, we propose a novel perspective on two challenging tasks: 1) hierarchical classification via label embeddings and 2) taxonomy extension of hyperbolic representations. To address the regression problem we consider previous methods as well as proposing two novel approaches that are computationally more advantageous: a parametric deep learning model that is informed by the geodesics of the target space and a non-parametric kernel-method for which we also prove excess risk bounds. Our experiments show that the strategy of leveraging the hyperbolic geometry is promising. In particular, in the taxonomy expansion setting, we find that the hyperbolic-based estimators significantly outperform methods performing regression in the ambient Euclidean space.

1 Introduction

Representation learning is a key paradigm in machine learning and artificial intelligence. It has enabled im-

portant breakthroughs in computer vision (Krizhevsky et al., 2012; He et al., 2016) natural language processing (Mikolov et al., 2013; Bojanowski et al., 2016; Joulin et al., 2016), relational learning (Nickel et al., 2011; Perozzi et al., 2014), generative modeling (Kingma and Welling, 2013; Radford et al., 2015), and many other areas (Bengio et al., 2013; LeCun et al., 2015). Its objective is typically to infer latent feature representations of objects (e.g., images, words, entities, concepts) such that their similarity or distance in the representation space captures their *semantic* similarity. For this purpose, the geometry of the representation space has recently received increased attention (Wilson et al., 2014; Falorsi et al., 2018; Davidson et al., 2018; Xu and Durrett, 2018). Here, we focus on Riemannian representation spaces and in particular on hyperbolic geometry. Nickel and Kiela (2017) introduced Poincaré embeddings to infer hierarchical representations of symbolic data, which led to substantial gains in representational efficiency and generalization performance. Hyperbolic representations have since been extended to other manifolds (Nickel and Kiela, 2018; De Sa et al., 2018), word embeddings (Tifrea et al., 2018; Le et al., 2019), recommender systems (Chamberlain et al., 2019), and image embeddings (Khrulkov et al., 2019).

However, it is yet an open problem how to efficiently integrate hyperbolic representations with standard machine learning methods which often make a Euclidean or vector space assumption. The work of Ganea et al. (2018) establishes some fundamental steps in this direction by proposing a generalization of fully connected neural network layers from Euclidean space to hyperbolic space. However most of the experiments shown were from hyperbolic to Euclidean space using recurrent models. In this paper we focus on the task of learning manifold-valued functions from Euclidean on to hyperbolic space that allows us to leverage its hierarchical structure for supervised learning. For this purpose, we propose two novel approaches: a deep learning model trained with a geodesic-based loss to learn hyperbolic-valued functions and a non-parametric

Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108. Copyright 2020 by the author(s).

kernel-based model for which we provide a theoretical analysis.

We illustrate the effectiveness of this strategy on two challenging tasks, i.e., hierarchical classification via label embeddings and taxonomy expansion by predicting concept embeddings from text. For standard classification tasks, label embeddings have shown great promise as they allow to scale supervised learning methods to datasets with massive label spaces (Chollet, 2016; Veit et al., 2018). By embedding labels in hyperbolic space according to their natural hierarchical structure (e.g, the underlying WordNet taxonomy of ImageNet labels) we are then able to combine the benefits of hierarchical classification with the scalability of label embeddings. Moreover, the continuous nature of hyperbolic space allows the model to *invent* new concepts by predicting their placement in a pre-embedded base taxonomy. We exploit this property for a novel task which we refer to as *taxonomy expansion*: Given an embedded taxonomy $\mathcal{T}$, we infer the placement of unknown novel concepts by predicting their features onto the embedding. In contrast to hierarchical classification, the predicted embeddings are here full members of the taxonomy, i.e., they can themselves act as parents of other points. For both tasks, we show empirically that the proposed strategy can often lead to more effective estimators than its Euclidean counterpart. These findings support the thesis of this work that leveraging the hyperbolic geometry can be advantageous for several machine learning settings. Additionally, we observe that the hyperbolic-based estimators introduced in this work achieve comparable performance to the previously proposed hyperbolic neural networks (Ganea et al., 2018). This suggests that, in practice, it is not necessary to work with hyperbolic layers as long as the training procedure exploits the geodesic as an error measure. This is advantageous from the computational perspective, since we found our proposed approaches to be generally significantly easier to train in practice.

The remainder of this paper is organized as follows: In Section 2 we briefly review hyperbolic embeddings and related concepts such as Riemannian optimization. In Section 3, we introduce our proposed methods and prove excess risk bounds for the kernel-based method. In Section 4 we evaluate our methods on the tasks of hierarchical classification and taxonomy expansion.

2 Hyperbolic Representations

Hyperbolic space is the unique, complete, simply connected Riemannian manifold with constant negative sectional curvature. There exist multiple equivalent models for hyperbolic space. To estimate the embeddings using stochastic optimization we will employ the

Lorentz model due to its numerical advantages. For analysis, we will map embeddings into the Poincaré disk which provides an intuitive visualization of hyperbolic embeddings. This can be easily done because the two models are isometric Nickel and Kiela (2018). We review both manifolds in the following.

Lorentz Model. Let $\mathbf{u}, \mathbf{v} \in \mathbb{R}^{n+1}$ and let $\langle \mathbf{u}, \mathbf{v} \rangle_{\mathcal{L}} = -u_0 v_0 + \sum_{i=1}^n u_i v_i$ denote the *Lorentzian scalar product*. The Lorentz model of n -dimensional hyperbolic space is then defined as the Riemannian manifold $\mathcal{L}^n = (\mathcal{H}^n, g_{\mathcal{L}})$, where

$$\mathcal{H}^n = \{\mathbf{u} \in \mathbb{R}^{n+1} : \langle \mathbf{u}, \mathbf{u} \rangle_{\mathcal{L}} = -1, x_0 > 0\}, \quad (1)$$

denotes the upper sheet of a two-sheeted n -dimensional hyperboloid and where $g_{\mathcal{L}}(\mathbf{u}) = \text{diag}([-1, 1, \dots, 1])$ is the associated metric tensor. Furthermore, the distance on $\mathcal{L}$ is defined as

$$d_{\mathcal{L}}(\mathbf{u}, \mathbf{v}) = \text{acosh}(-\langle \mathbf{u}, \mathbf{v} \rangle_{\mathcal{L}}). \quad (2)$$

An advantage of the Lorentz model is that its exponential map has as simple, closed-form expression. As showed by Nickel and Kiela (2018), this allows us to perform Riemannian optimization efficiently and with increased numerical stability. In particular, let $\mathbf{u} \in \mathcal{L}^n$ and let $\mathbf{z} \in \mathcal{T}_{\mathbf{u}}\mathcal{L}^n$ denote a point in the associated tangent space. The exponential map $\exp_{\mathbf{u}} : \mathcal{T}_{\mathbf{u}}\mathcal{L}^n \rightarrow \mathcal{L}^n$ is then defined as

$$\exp_{\mathbf{u}}(\mathbf{z}) = \cosh(\|\mathbf{z}\|_{\mathcal{L}})\mathbf{u} + \sinh(\|\mathbf{z}\|_{\mathcal{L}})\frac{\mathbf{z}}{\|\mathbf{z}\|_{\mathcal{L}}}. \quad (3)$$

Poincaré ball. The Poincaré ball model is the Riemannian manifold $\mathcal{P}^n = (\mathcal{B}^n, g_p)$, where $\mathcal{B}^n = \{\mathbf{u} \in \mathbb{R}^n : \|\mathbf{u}\| < 1\}$ is the *open* n -dimensional unit ball and where $g_p(\mathbf{u}) = 4/(1 - \|\mathbf{u}\|^2)^2$ is the associated metric tensor. The distance function on $\mathcal{P}$ is defined as

$$d_p(\mathbf{u}, \mathbf{v}) = \text{acosh}\left(1 + 2\frac{\|\mathbf{u} - \mathbf{v}\|^2}{(1 - \|\mathbf{u}\|^2)(1 - \|\mathbf{v}\|^2)}\right). \quad (4)$$

An advantage of the Poincaré ball is that it provides an intuitive model of hyperbolic space which is well suited for analysis and visualization of the embeddings. It can be seen from Eq. (4), that the distance within the Poincaré ball changes smoothly with respect to the norm of $\mathbf{u}$ and $\mathbf{v}$. This locality property of the distance is key for representing hierarchies efficiently (Hamann, 2018). For instance, by placing the root node of a tree at the origin of $\mathcal{B}^n$, it would have relatively small distance to all other nodes, as its norm is zero. On the other hand, leaf nodes can be placed close to the boundary of the ball, as the distance between points grows quickly with a norm close to one.

Hyperbolic embeddings. We consider supervised datasets $\{x_i, c_i\}_{i=1}^m \in \mathcal{X} \times \mathcal{C}$ where class labels c_i can be organized according to a taxonomy or class hierarchy $\mathcal{T} = (\mathcal{C}, \mathcal{E})$. Edges $(i, j) \in \mathcal{E}$ indicate that c_i is-a c_j . To compute hyperbolic embeddings of all c_i that capture these hierarchical relationships of $\mathcal{T}$, we follow the works of Nickel and Kiela (2017, 2018) and infer the embedding from pairwise similarities. In particular, let $\gamma : \mathcal{C} \times \mathcal{C} \rightarrow \mathbb{R}_+$ be the similarity function such that

$$\gamma(c_i, c_j) = \begin{cases} 1, & \text{if } c_i, c_j \text{ are adjacent in } \text{clos}(\mathcal{T}) \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

where $\text{clos}(\mathcal{T})$ is the transitive closure of $\mathcal{T}$. Furthermore, let $\mathcal{N}(i, j) = \{\ell : \gamma(i, \ell) < \gamma(i, j)\} \cup \{j\}$ denote the set of concepts that are *less* similar to c_i than c_j (including c_j) and let $\phi(i, j) = \arg \min_{k \in \mathcal{N}(i, j)} d(\mathbf{u}_i, \mathbf{u}_k)$ denote the nearest neighbor of c_i in the set $\mathcal{N}(i, j)$. We then learn embeddings $\Theta = \{\mathbf{u}\}_{i=1}^m$ by optimizing

$$\min_{\Theta} - \sum_{i, j} \log \Pr(\phi(i, j) = j \mid \Theta) \quad (6)$$

with

$$\Pr(\phi(i, j) = j \mid \Theta) = \frac{e^{d(\mathbf{u}_i, \mathbf{u}_j)}}{\sum_{k \in \mathcal{N}(i, j)} e^{d(\mathbf{u}_i, \mathbf{u}_k)}}. \quad (7)$$

Eq. (7) can be interpreted as a ranking loss that aims to extract latent hierarchical structures from $\mathcal{C}$. For computational efficiency, we follow Jean et al. (2014) and randomly subsample $\mathcal{N}(i, j)$ on large datasets. To infer the embeddings θ we then minimize Eq. (7) using Riemannian SGD (Bonnabel, 2013). In RSGD, updates to the parameters θ are computed via

$$\theta_{t+1} = \exp_{\theta_t} \left(-\eta \sum_{j \in B} \text{grad}_{\mathcal{L}} f_j(\theta_t) \right), \quad (8)$$

where $\text{grad}_{\mathcal{L}} f(\theta_t) \in \mathcal{T}_{\theta} \mathcal{L}$ denotes the *Riemannian gradient*, η denotes the learning rate, and $B = [j_1, \dots, j_B]$ is a set of random uniformly sampled indexes.

By computing hyperbolic embeddings of $\mathcal{T}$, we have then recast the learning problem from a discrete tree $\mathcal{D} = \{x_i, c_i\}_{i=1}^m, c_i \in \mathcal{C}$ to its embedding in a continuous manifold $\mathcal{D}^e = \{x_i, y_i\}_{i=1}^m$ with $y_i \in \mathcal{L}$. This allows us to apply manifold regression techniques as discussed in the following.

3 Manifold Valued Prediction in Hyperbolic Space

We study the problem of learning $f : \mathcal{X} \rightarrow \mathcal{Y} \subset \mathcal{L}^n$ a map taking values in the hyperbolic space, often referred to as *manifold regression* (Steinke and Hein,

2009; Steinke et al., 2010). We assume for simplicity that $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} \subset \mathcal{L}^n$ are compact subsets. In particular, we assume a training dataset $\{x_i, y_i\}_{i=1}^m$ of points independently sampled from a joint distribution ρ on $\mathcal{X} \times \mathcal{Y}$ and aim to find an estimator for the minimizer of the expected risk

$$\min_{f: \mathcal{X} \rightarrow \mathcal{Y}} \mathcal{E}(f) \quad \mathcal{E}(f) = \int d_{\mathcal{L}}(f(x), y)^2 d\rho(x, y). \quad (9)$$

Here we consider $\mathcal{L}^n$ as target space and $d_{\mathcal{L}}$ as loss function, but all results extend to $\mathcal{P}$. Eq. (9) is the natural generalization of standard vector-valued ridge regression (indeed the geodesic of $\mathcal{Y} = \mathbb{R}^n$ is the Euclidean distance $d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|$). We tackle this problem proposing two novel approaches: one leveraging recent results on structured prediction and one using geodesic neural networks.

Structured Prediction. Rudi et al. (2018) proposed a new approach to address manifold regression problems. The authors adopted a perspective based on structured prediction and interpreted the target manifold $\mathcal{Y}$ as a “structured” output. While standard structured prediction studies settings where $\mathcal{Y}$ is a discrete (often finite) space (Bakir et al., 2007), this extension allowed the authors to design a kernel-based approach for structured prediction for which they provided a theoretical analysis under suitable assumptions on the output space. We formulate the corresponding Hyperbolic Structured Prediction (HSP) estimator when applying this strategy to our problem (namely $\mathcal{Y} \subset \mathcal{L}^n$). In particular, we have $f_{hsp} : \mathcal{X} \rightarrow \mathcal{L}^n$ the function such that for any test point $\mathbf{x} \in \mathcal{X}$

$$f_{hsp}(\mathbf{x}) = \arg \min_{\mathbf{y} \in \mathcal{L}^n} \sum_{i=1}^m \alpha_i(\mathbf{x}) d_{\mathcal{L}}(\mathbf{y}, \mathbf{y}_i)^2, \quad (10)$$

where the weights $\alpha(\mathbf{x}) = (\alpha_1(\mathbf{x}), \dots, \alpha_n(\mathbf{x}))^\top \in \mathbb{R}^m$ are learned by solving a variant of kernel ridge regression: given $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ a reproducing kernel on the input space, we obtain

$$\alpha(\mathbf{x}) = (\mathbf{K} + \lambda \mathbf{I})^{-1} \mathbf{v}(\mathbf{x}), \quad (11)$$

where $\mathbf{K} \in \mathbb{R}^{m \times m}$ is the empirical kernel matrix and $\mathbf{v} \in \mathbb{R}^m$ is the evaluation vector with entries with entries respectively $\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ and $\mathbf{v}(\mathbf{x})_i = k(\mathbf{x}, \mathbf{x}_i)$ for $i, j \in \{1, \dots, m\}$.

In line with most literature on structured prediction, the estimator in (10) requires solving an optimization problem at every test point. Hence, while this approach offers a significant advantage at training time (when learning the weights α), it can lead to a more expensive operation at test time. To solve this problem in practice we resort to RSGD as defined in (8).

Rudi et al. (2018), studied the generalization properties of estimators of the form of (10). The authors proved that under suitable assumptions on the regularity of the output manifold, it was possible to give bounds on the excess risk in terms of the number of training examples available. The following theorem specializes this result to the case of f_{hsp} . A key role will be played by the $(s, 2)$ -Sobolev space $W^{s,2}$ of functions from $\mathcal{L}^n$ to $\mathbb{R}$, which generalizes the standard notion on Euclidean domains (see Hebey, 2000).

Theorem 1. *Let $(\mathbf{x}_i, \mathbf{y}_i)_{i=1}^m$ be sampled independently according to ρ on $\mathcal{X} \times \mathcal{Y}$ with $\mathcal{Y} \subset \mathcal{L}^m$ compact sets. Let f_{hsp} defined as in (10) with weights (11) learned with reproducing kernel $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ with reproducing kernel Hilbert space (RKHS) $\mathcal{F}$. If the map $x \mapsto \int d_{\mathcal{L}}(\cdot, \mathbf{y})^2 d\rho(\mathbf{y}|x)$ belongs to $W^{s,2}(\mathcal{Y}) \otimes \mathcal{F}$ with $s > n/2$, then for any $\tau \in (0, 1]$*

$$\mathcal{E}(f_{hsp}) - \inf_f \mathcal{E}(f) \leq \|d_{\mathcal{L}}^2\|_{s,2} \mathbf{q} \tau^2 \frac{1}{n^{1/4}}, \quad (12)$$

holds with probability at least $1 - 8e^{-\tau}$, where $\mathbf{q}$ is a constant not depending on n, τ or $\|d_{\mathcal{L}}\|_{s,2}$.

The result guarantees a learning rate of order $O(n^{-1/4})$. We comment on the assumptions and constants appearing in Thm. 1. First, we point out that, albeit the requirement $\int d_{\mathcal{L}}(\cdot, \mathbf{y})^2 d\rho(\mathbf{y}|x) \in \mathcal{F} \otimes W^{s,2}(\mathcal{Y})$ can seem overly abstract, it reduces to a standard assumption in statistical learning theory. Informally, it corresponds to a regularity assumption on the conditional mean embedding of the distribution $\rho(\cdot|x)$ (see the work of Song et al. (2013) for more details), and can be interpreted as requiring the solution of (9) to belong to the hypotheses space associated to the kernel k . Second, we comment on the constant in (12) that depending on the geodesic distance. In particular, we note that by Thm. 2 of Rudi et al. (2018) the squared geodesic on any compact subset of $\mathcal{L}^n$ belongs to $W^{s,2}(\mathcal{Y})$ for any $s \geq 0$. Hence $\|d_{\mathcal{L}}^2\|_{s,2} < +\infty$ also for any $s > n/2$, as required by Thm. 1.

Proof. The proof of Thm. 1 is a specialization of Thm. 2 and 4 by Rudi et al. (2018). We recall a key assumption that is required to apply such results.

Assumption 1. *$\mathcal{M}$ is a complete n -dimensional smooth connected Riemannian manifold, without boundary, with Ricci curvature bounded below and positive injectivity radius.*

The assumption above imposes basic regularity conditions on the output manifold. A first implication is indeed that.

Proposition 2 (Thm. 2 in Rudi et al. (2018)). *Let $\mathcal{M}$ satisfy assumption 1 and let $\mathcal{Y} \subset \mathcal{M}$ is a compact*

geodesically convex subset of $\mathcal{M}$. Then, the squared geodesic distance $d : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}$ is smooth on $\mathcal{Y}$. Moreover, by the proof of Thm.1 in the appendix of Manifold Structured Prediction (Rudi et al., 2018), we have $d^2 \in W^{s,2}(\mathcal{Y})$ for any $s > n/2$.

Leveraging standard results from Riemannian geometry, we can guarantee that the manifolds considered in this paper satisfy the above requirements. For simplicity, we restrict on $\mathcal{M}$ corresponding to an open bounded ball in either $\mathcal{P}^n$ or $\mathcal{L}^n$. In particular,

- $\mathcal{M}$ has sectional curvature constantly equal to -1 . Hence the Ricci curvature is bounded from below since we are in a bounded ball in either $\mathcal{P}^n$ or $\mathcal{L}^n$.
- The injectivity radius is positive (actually lower bounded by $1/(2 \cdot 9^{2+[n/2]})$ with $[n/2]$ the integer parts of $n/2$), see Main Theorem by Martin (1989).

We see that we are in the hypotheses of Prop. 2, from which we conclude the following.

Corollary 3. *For any $s \geq 0$, the geodesic distance $d_{\mathcal{L}}$ (respectively $d_{\mathcal{P}}$) belongs to $W^{s,2}(\mathcal{Y})$ for any compact subspace of $\mathcal{L}^n$ (respectively $\mathcal{P}^n$).*

This guarantees us that we are in the hypotheses of (Rudi et al., 2018, Thm. 4), from which Thm. 1 follows. We note in particular that $d_{\mathcal{L}}^2$ takes the role of the loss function Δ in the original theorem. Which needs to be a so-called ‘‘Structure Encoding Loss Function’’. The latter is guaranteed by Cor. 3 above. $\square$

Neural Network with Geodesic loss (NN-G). As an alternative to the non-parametric model f_{hsp} , we consider also a parametric method based on deep neural networks. An important challenge when dealing with manifold regression is how to design a suitable model for the estimator. While neural networks of the form $g_{\theta} : \mathbb{R}^d \rightarrow \mathbb{R}^k$ (parametrized by some weights θ) have proven to be powerful models for regression and feature representation (LeCun et al., 2015; Bengio et al., 2013; Xiao et al., 2016; Ngiam et al., 2011), it is unclear how to enforce the constraint for a candidate function to take values on the manifold since their canonical forms are designed to act between linear spaces. To address this limitation, we consider in the following the Poincaré ball model and develop a neural architecture mapping the Euclidean space into the open unit ball. In particular, let the element-wise hyperbolic tangent be defined as

$$h : \mathbb{R}^k \rightarrow \{\mathbf{x} \in \mathbb{R}^k : \|\mathbf{x}\|_{\infty} < 1\} \quad (13)$$

$$(x_1, \dots, x_k) \mapsto (\tanh x_1, \dots, \tanh x_k), \quad (14)$$

which maps a linear space onto the open ℓ_∞ ball. Moreover, we define a “squashing” function

$$s: \{\mathbf{x} \in \mathbb{R}^k : \|\mathbf{x}\|_\infty < 1\} \rightarrow \{\mathbf{x} \in \mathbb{R}^k : \|\mathbf{x}\|_2 < 1\} \quad (15)$$

$$s(\mathbf{x}) = \begin{cases} \mathbf{x} \mapsto \mathbf{x} \frac{\|\mathbf{x}\|_\infty}{\|\mathbf{x}\|_2}, & \text{if } \mathbf{x} \neq \mathbf{0} \\ \mathbf{0}, & \text{if } \mathbf{x} = \mathbf{0} \end{cases} \quad (16)$$

where $\mathbf{0}$ is the vector of all zeros. Since $\|\mathbf{x}\|_\infty < \|\mathbf{x}\|_2$, this function is continuous and maps the open ℓ_∞ ball into the open ℓ_2 ball. And because both s and h are bijective continuous function with continuous inverse, the composition $s \circ h: \mathbb{R}^k \rightarrow \{\mathbf{x} \in \mathbb{R}^k : \|\mathbf{x}\|_2 < 1\}$ is also a homeomorphism from $\mathbb{R}^k$ into the open ball ℓ_2 and therefore also on the Poincaré model manifold. By composing $s \circ h$ with the neural network feature extractor g_θ we obtain a deep model that jointly learns features into a linear space and maps them to the hyperbolic manifold:

$$f_{nn} = s \circ h \circ g_\theta: \mathbb{R}^d \rightarrow \mathcal{P}^k. \quad (17)$$

Note that the homeomorphism $s \circ h$ is sub-differentiable. Therefore learning the parameters θ of this model is akin to training a classical deep learning architecture with activation functions at the output layer corresponding to $s \circ h$. The key difference here lies in the loss used for training. In this setting, analogously to the task addressed by HSP, we replaced of the standard mean-squared error (Euclidean) loss with the squared geodesic distance between predictions and true labels.

Hyperbolic embeddings and manifold regression. In this work we propose to leverage the hyperbolic geometry to address machine learning tasks where hierarchical structures play a central role. In particular, we combine label embeddings approaches with hyperbolic regression to perform hierarchical classification. We do this by following a two step procedure: assuming a hierarchy $\mathcal{T}$, we consider an augmented $\mathcal{T}_\mathcal{X}$ where each example x_i corresponds to a child to its associated class c_i from the original $\mathcal{T}$. Then, we embed $\mathcal{T}_\mathcal{X}$ into the hyperbolic space using the procedure reviewed in Section 2. We compute similarity scores $\gamma(\cdot, \cdot)$ in the transitive closure of $\mathcal{T}_\mathcal{X}$, using either a Gaussian kernel on the features – when both nodes have a corresponding representation available – or otherwise employing the original γ . This allows us to incorporate information about feature similarities within the label embedding.

4 Experiments

We evaluate our proposed methods for hyperbolic manifold regression on the following experiments:

Hierarchical Classification via Label Embeddings. For this task, the goal is to classify examples with a single label from a class hierarchy with tree structure. We begin by computing label embeddings of the class hierarchy via hyperbolic representations. We then learn to regress examples onto label embeddings and classify them using the nearest label in the target space, i.e., by denoting $\mathbf{y}_c \in \mathcal{L}^n$ the embedding of class c and taking $f: \mathbb{R}^d \rightarrow \mathcal{L}^n$.

$$\hat{c} = \arg \min_{c \in \mathcal{C}} d(f(\mathbf{x}), \mathbf{y}_c) \quad (18)$$

Taxonomy expansion. For this task, the goal is to expand an existing taxonomy based on feature information about new concepts. As for hierarchical classification, we first embed the existing taxonomy in hyperbolic space and then learn to regress onto the label embeddings. However, a key difference is that a new example c can themselves act as the parent of another class c' .

Models and training details. For hierarchical classification, we compare to standard baselines such as top-down classification with logistic regression (TD-LR) and hierarchical SVM (HSVM). Furthermore, since both tasks can be regarded as regression problems onto the Poincaré ball (which has a canonical embedding in $\mathbb{R}^k$) we also compare to kernel regularized least squares regression (KRLS) and a neural network with squared Euclidean loss (NN-E). In both cases, we constrain predictions to remain within the Poincaré ball via the projection

$$\text{proj}(\mathbf{y}) = \begin{cases} \mathbf{y}/\|\mathbf{y}\| - \varepsilon & \text{if } \|\mathbf{y}\| \geq 1 \\ \mathbf{y} & \text{otherwise} \end{cases},$$

where ε is a small constant to ensure numerical stability, equal to $\varepsilon = 10^{-6}$. These regression baselines allows us to evaluate the advantages of training manifold-valued models with squared geodesic loss compared to standard methods that are agnostic of the underlying geodesics.

For kernel-based methods, we employ a Gaussian kernel selecting the bandwidth $\sigma \in [10^{-1}, 10^2]$ and regularization parameter $\lambda \in [10^{-6}, 10^{-2}]$ via cross-validation. Both parameter ranges are logarithmically spaced. For HSP inference we use RSGD with batch size equal to 50 and a maximum of 40000 iterations. We stop the minimization if the the gradient Euclidean norm is smaller than 10^{-5} (In most cases the inference stops before the 10000 iteration). The learning rate for RSGD is chosen via cross-validation on the interval $[10^{-5}, 10^{-1}]$. For the neural network models (NN-G, NN-E) we use the same architecture for g_θ : each layer is a fully connected network

$$z^\ell = \psi(W_\ell z^{\ell-1} + b_\ell)$$

		Model - Performance (Relative Rank)									
		TD-LR		HSVM		NN-E		NN-G		HSP	
News-20	μ F1	77.07	(3)	80.79	(1)	63.91	(5)	72.67	(4)	80.28	(2)
	MF1	77.94	(3)	80.04	(1)	64.21	(5)	72.70	(4)	79.56	(2)
Imclef07a	μ F1	73.86	(3)	74.98	(2)	65.49	(5)	67.49	(4)	75.95	(1)
	MF1	36.03	(3)	50.44	(1)	26.76	(5)	31.20	(4)	46.41	(2)
Wipo	μ F1	36.85	(2)	38.48	(1)	16.87	(5)	16.69	(6)	31.94	(3)
	MF1	52.18	(3)	52.21	(2)	42.77	(5)	42.86	(4)	52.41	(1)
Diatoms	μ F1	54.01	(1)	48.97	(3)	9.25	(5)	11.31	(4)	53.20	(2)
	MF1	55.53	(2)	44.61	(3)	14.90	(4)	14.61	(5)	62.10	(1)
Avg. Rank		(2.5)		(1.75)		(4.88)		(4.38)		(1.75)	

Table 1: Hierarchical classification on benchmark datasets. We report micro-F1 (μ F1), macro-F1 (MF1), as well as the rank relative to all other models on a dataset, e.g., (1) for the the best performing model.

where $\psi(x) = \max(0, x)$ is a ReLU non-linearity and $\theta = \{W \in \mathbb{R}^{s/2 \times s}, b \in \mathbb{R}^{s/2}\}$, with s the dimension of the previous layer (with the exception of the first and last layer which must fit input and output dimensions). We use a depth of 5 layers with intermediate dimensionalities $s \in (1024, 1024, 512, 256, 128)$ for taxonomy expansion and $s \in (2048, 2048, 1024, 512, 256)$ for hierarchical classification. We did not find significant improvements with deeper architectures in performance. We train the deep models using mini-batch stochastic gradient descent, with a scheduler until the model reaches convergence on the training loss. For taxonomy expansion we also compare our algorithms with a hyperbolic neural networks (HNN) as introduced by Ganea et al. (2018). This architecture is trained with Riemannian Stochastic Gradient Descent until convergence and has the same structure and the same number of parameters of NN-G and NN-E. Because NN-G uses fully connected layers until the homeomorphic transformation, it can be trained with traditional optimizers such as stochastic gradient descent or Adam (Kingma and Welling, 2013). In our experiments, we observe that this can be an important advantage as these models require typically one third of the training time compared to HNNs.

4.1 Hierarchical classification

For hierarchical classification, we are given a supervised training set $\mathcal{D} = \{x_i, c_i\}_{i=1}^m$ where the class labels c_i are organized in a tree $\mathcal{T}$. We first embed the augmented hierarchy $\mathcal{T}_x$ as discussed in Section 3 and learn a regression function $\hat{f}: \mathbb{R}^d \rightarrow \mathcal{L}^n$ using $\mathcal{D}^e = \{x_i, y_i\}_{i=1}^m$. For a test point $x' \in \mathbb{R}^d$, we first map it onto the target manifold $\hat{y} = \hat{f}(x')$ and then classify $\hat{y}$ according to Eq. (18). For evaluation, we use various benchmark datasets for hierarchical classification¹, and Newsgroups-20² for which we manually

extract TF-IDF features $x_i \in \mathbb{R}^{10000}$ from the original documents. We compute an embedding for the augmented hierarchies of each dataset. To make sure to obtain a good embedding, we perform parameter-tuning in order to attain mAP of at least 0.99. We then train HSP, NN-G and NN-E as described above and measure classification performance in terms of μ F1 and macroF1 scores. As a baseline we also train Hierarchical SVM (HSVM) (Vateekul et al., 2012) and Top-Down Logistic Regression (TD-LR) (Naik and Rangwala, 2018).

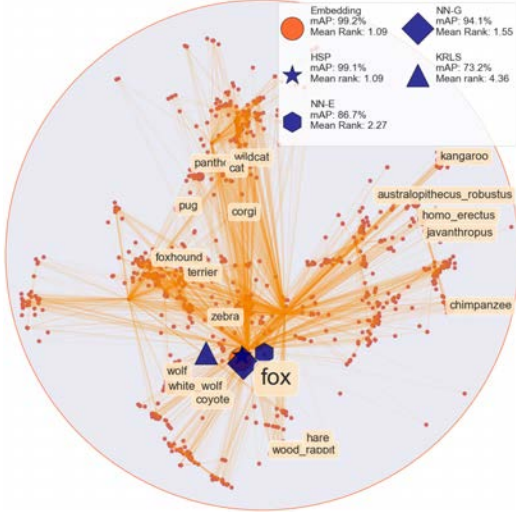
Table 1 shows the results of our experiments. It can be seen that the hyperbolic structured predictor achieves results comparable to state-of-the-art on this task although we did not explicitly optimize the embedding or training loss for hierarchical classification. We also observe that while NN-G outperforms NN-E, both algorithms perform significantly worse on Wipo and Diatoms datasets. Interestingly, these two datasets are significantly smaller compared to Newsgroup-20 and Imclef07a in terms of number of training points ($\sim 1K$ Vs $\sim 10K$ training samples). This seems to suggest that NN-G and NN-E models have a higher sample complexity.

4.2 Taxonomy expansion

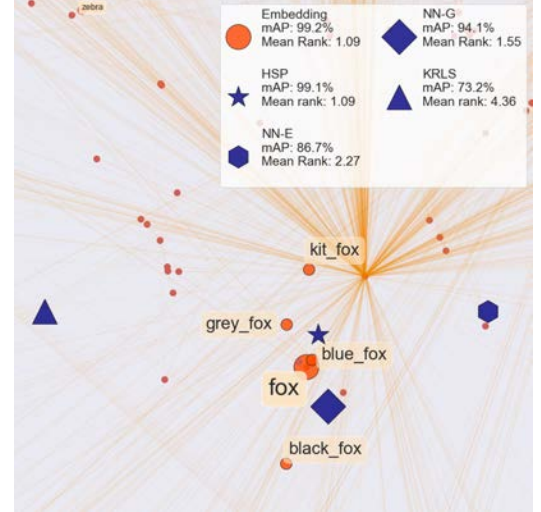
For taxonomy expansion, we assume a similar setting as for hierarchical classification. We are given a dataset $\mathcal{D} = \{x_i, c_i\}_{i=1}^m$ where concepts c_i are organized in a taxonomy $\mathcal{T}$ and for each concept we have an additional feature representation x_i . Again, we first embed the augmented hierarchy $\mathcal{T}_x$ as discussed in Section 3 and split it in train $\mathcal{D}_{\text{train}}^e$ and test set $\mathcal{D}_{\text{test}}^e$. We vary

¹<https://sites.google.com/site/hrsvmproject/>

²<http://qwone.com/~jason/20NewsGroups/>



(a) WordNet mammals embedding



(b) Close-up of predicted embedding for 'Fox'

Figure 1: Overview and close-up of predicted positions for entity 'Fox'. Models that do not use the geometry of the hyperbolic manifold fail at positioning the entity, while the geodesic neural network and the hyperbolic structured predictor position the entity accordingly to its real neighbours.

the size of the test set, i.e., the number of unknown concepts in $\mathcal{T}$ such that $|\mathcal{D}_{\text{test}}| \in \{5, 10, 20, 30, 50\}$. Whenever necessary, we also create a validation set from $\mathcal{D}_{\text{train}}$ for model selection with a 80 : 20 ratio for model selection. We then train all regression functions $\hat{f}: \mathbb{R}^d \rightarrow \mathcal{L}^n$ using $\mathcal{D}_{\text{train}}^e$ and predict embeddings for $\mathcal{D}_{\text{test}}$. In contrast to hierarchical classification, the predicted points $\hat{\mathbf{y}} = f(\mathbf{x})$ can themselves act as parents of other points, i.e., they are full members of the taxonomy $\mathcal{T}$. To assess the quality of the predictions we use mean average prediction (mAP) as proposed by Nickel and Kiela (2017). We report mAP for the predicted points as well as for the points originally embedded by the Lorentz embedding (Orig). This experiment is repeated 20 times for a given size of the test set, each time selecting a new training-test split. In our experiments, we consider the following datasets:

WordNet Mammals. For WordNet Mammals, the goal is to expand an existing taxonomy by predicting concept embeddings from text. For this purpose, we take the mammals hierarchy of WordNet and retrieve for each node its corresponding Wikipedia page. If a page is missing, we remove the corresponding node and if a page has multiple candidates we disambiguate manually. The transitive closure of $\mathcal{T}$ has 1036 nodes and 11222 edges. Next, we pre-process the retrieved Wikipedia descriptions by removing all non alphabetical characters, tokenizing words and removing stopwords using NLTK (Loper and Bird, 2002). Finally, we associate to each concept $c_i \in \mathcal{T}$ the TF-IDF vector of its Wikipedia description as feature representation $x_i \in \mathbb{R}^{10000}$ computed using Scikit-learn (Pedregosa et al., 2011). We

then embed $\mathcal{T}$ following Section 2 and obtain an embedding with mAP 0.86 and mean rank 4.74. This dataset is particularly difficult given the way features were collected: Wikipedia pages have a high variance in quality and amount of content, while some pages are detailed and rich in information other barely contain a full sentence.

Synthetic datasets. To better control for noise in the feature representations, we also generate datasets based on synthetic random trees, i.e., a smaller tree with 226 nodes and 1228 edges and a larger tree with 2455 nodes and 30829 edges after transitive closure. For each node we take as feature vector the corresponding row of the adjacency matrix of the transitive closure of the tree. We project these rows on the first d principal components of the adjacency matrix, where $d = 50$ for the small tree and $d = 500$ for the big tree. We then embed the nodes of the graph in $\mathcal{L}^5$ using both the tree structure and similarity scores computed using the vector features. The similarity is computed by a Gaussian kernel with σ equal to the average tenth nearest neighbour of the dataset.

Results We provide the results of our evaluation for different sizes on $\mathcal{D}_{\text{test}}^e$ in Table 2. It can be seen that all hyperbolic-based methods can successfully predict the embeddings of unknown concepts when the test set is small. The performance degrades as the size of the test set increases, since it becomes harder to leverage the original structure of the graph. While all methods are affected by this trend, we note that algorithms using the geodesic loss tend to perform better than those working in the linear space. This suggest that taking

		Number of new concepts				
		5	10	20	30	50
Wordnet Mammals	Orig	0.86 ± 0.06	0.88 ± 0.06	0.87 ± 0.03	0.87 ± 0.03	0.88 ± 0.02
	KRLS	0.54 ± 0.14	0.37 ± 0.07	0.26 ± 0.04	0.22 ± 0.03	0.15 ± 0.02
	NN-E	0.61 ± 0.12	0.47 ± 0.08	0.38 ± 0.04	0.31 ± 0.03	0.20 ± 0.03
	NN-G	0.79 ± 0.08	0.74 ± 0.06	0.63 ± 0.06	0.61 ± 0.06	0.50 ± 0.04
	HNN	0.82 ± 0.05	0.73 ± 0.05	0.63 ± 0.04	0.57 ± 0.05	0.46 ± 0.04
	HSP	0.72 ± 0.10	0.69 ± 0.07	0.69 ± 0.07	0.58 ± 0.09	0.50 ± 0.06
Synthetic Small	Orig	0.94 ± 0.03	0.93 ± 0.03	0.94 ± 0.02	0.94 ± 0.02	0.94 ± 0.01
	KRLS	0.63 ± 0.16	0.51 ± 0.12	0.36 ± 0.06	0.27 ± 0.03	0.21 ± 0.02
	NN-E	0.76 ± 0.07	0.72 ± 0.09	0.63 ± 0.09	0.56 ± 0.09	0.45 ± 0.08
	NN-G	0.80 ± 0.07	0.73 ± 0.06	0.61 ± 0.06	0.55 ± 0.05	0.45 ± 0.04
	HNN	0.82 ± 0.01	0.71 ± 0.07	0.60 ± 0.05	0.51 ± 0.04	0.41 ± 0.04
	HSP	0.82 ± 0.08	0.76 ± 0.07	0.66 ± 0.05	0.60 ± 0.04	0.50 ± 0.03
Synthetic Large	Orig	0.81 ± 0.06	0.79 ± 0.05	0.80 ± 0.03	0.80 ± 0.02	0.80 ± 0.01
	KRLS	0.30 ± 0.05	0.20 ± 0.02	0.13 ± 0.01	0.09 ± 0.01	0.07 ± 0.00
	NN-E	0.69 ± 0.09	0.68 ± 0.09	0.64 ± 0.05	0.61 ± 0.05	0.59 ± 0.05
	NN-G	0.77 ± 0.07	0.72 ± 0.07	0.71 ± 0.04	0.69 ± 0.04	0.65 ± 0.03
	HNN	0.83 ± 0.7	0.79 ± 0.4	0.72 ± 0.06	0.64 ± 0.06	0.63 ± 0.02
	HSP	0.76 ± 0.09	0.70 ± 0.07	0.69 ± 0.04	0.67 ± 0.05	0.63 ± 0.03

Table 2: Mean average precision for taxonomy expansion on WordNet mammals and synthetic data

into account the local geometry of the embedding is indeed beneficial in estimating the relative position of novel points in the space.

We conclude by noting that all hyperbolic-based methods have comparable performance across the three settings. However, we point out that HSP and NN-G offer significant practical advantages over HNN: in all our experiments they were faster to train and in general more amenable to model design. In particular, since HSP is based on a kernel method, it has relatively fewer hyperparameters and requires only solving a linear system at training time. NN-G consists of a standard neural architecture with the homeomorphism activation function introduced in Section 3 and trained with the geodesic loss. This allows one to leverage all current packages available to train neural networks, significantly reducing both modeling and training times.

5 Conclusion

In this paper, we showed how to recast supervised problems with hierarchical structure as manifold-valued regressions in the hyperbolic manifold. We then proposed two algorithms for learning manifold-valued functions mapping from Euclidean to hyperbolic space: a non-parametric kernel-based method for which we also proved generalization bounds and a parametric deep-learning model that is informed by the geodesics of the

output space. The latter makes possible to leverage traditional neural network layers for regression on hyperbolic space without resorting to hyperbolic layers, thus requiring a smaller training time. We evaluated both methods empirically on the task of hierarchical classification and showed that hyperbolic structured prediction shows strong generalization performance. We also showed that hyperbolic manifold regression enables new applications in supervised learning. By exploiting the continuous representation of hierarchies in hyperbolic space we were able to place unknown concepts in the embedding of a taxonomy using manifold regression. Moreover, by comparing to hyperbolic neural networks we showed that for this application, the key step is leveraging the geodesic of the manifold. In this work, we have aimed at developing a foundation for regressing onto hyperbolic representations. In future work, we plan to exploit this framework in dedicated methods for hierarchical machine learning and extending the applications to manifold product spaces.

Acknowledgments

We thank Maximilian Nickel for his invaluable support and feedback throughout this project. Without him, this paper would have not been possible.

This material is based upon work supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216, and the Italian Institute of Tech-

nology. We gratefully acknowledge the support of NVIDIA Corporation for the donation of the Titan Xp GPUs and the Tesla k40 GPU used for this research. This work has been carried out at the Machine Learning Genoa (MaLGa) center, Università di Genova (IT). L. R. acknowledges the financial support of the European Research Council (grant SLING 819789), the AFOSR projects FA9550-17-1-0390 and BAA-AFRL-AFOSR-2016-0007 (European Office of Aerospace Research and Development), and the EU H2020-MSCA-RISE project NoMADS - DLV-777826.

References

- G. Bakir, T. Hofmann, B. Schölkopf, A. Smola, B. Taskar, and S. Vishwanathan. Predicting structured data. *neural information processing*, 2007.
- Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.
- P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.
- S. Bonnabel. Stochastic gradient descent on Riemannian manifolds. *IEEE Trans. Automat. Contr.*, 58(9):2217–2229, 2013. doi: 10.1109/TAC.2013.2254619. URL <http://dx.doi.org/10.1109/TAC.2013.2254619>.
- B. P. Chamberlain, S. R. Hardwick, D. R. Wardrope, F. Dzo-gang, F. Daolio, and S. Vargas. Scalable hyperbolic recommender systems. *arXiv preprint arXiv:1902.08648*, 2019.
- F. Chollet. Information-theoretical label embeddings for large-scale image classification. *arXiv preprint arXiv:1607.05691*, 2016.
- T. R. Davidson, L. Falorsi, N. De Cao, T. Kipf, and J. M. Tomczak. Hyperspherical variational auto-encoders. *arXiv preprint arXiv:1804.00891*, 2018.
- C. De Sa, A. Gu, C. Ré, and F. Sala. Representation Tradeoffs for Hyperbolic Embeddings. In *International Conference on Machine Learning*, pages 4460–4469, jul 2018. URL <http://proceedings.mlr.press/v80/sala18a.html><http://arxiv.org/abs/1804.03329>.
- L. Falorsi, P. de Haan, T. R. Davidson, N. De Cao, M. Weiler, P. Forré, and T. S. Cohen. Explorations in homeomorphic variational auto-encoding. *arXiv preprint arXiv:1807.04689*, 2018.
- O.-E. Ganea, G. Bécigneul, and T. Hofmann. Hyperbolic Neural Networks. In *Advances in neural information processing systems*, pages 5345–5355, 2018. URL <https://papers.nips.cc/paper/7780-hyperbolic-neural-networks>.
- M. Hamann. On the tree-likeness of hyperbolic spaces. *Mathematical Proceedings of the Cambridge Philosophical Society*, 164(2):345–361, 2018. doi: 10.1017/S0305004117000238.
- K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
- E. Hebey. *Nonlinear analysis on manifolds: Sobolev spaces and inequalities*, volume 5. American Mathematical Soc., 2000.
- S. Jean, K. Cho, R. Memisevic, and Y. Bengio. On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007*, 2014.
- A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.
- V. Khrulkov, L. Mirvakhabova, E. Ustinova, I. Oseledets, and V. Lempitsky. Hyperbolic image embeddings. *arXiv preprint arXiv:1904.02239*, 2019.
- D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- M. Le, S. Roller, L. Papaxanthos, D. Kiela, and M. Nickel. Inferring concept hierarchies from text corpora via hyperbolic embeddings. *arXiv preprint arXiv:1902.00913*, 2019.
- Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436, 2015.
- E. Loper and S. Bird. NLTK. In *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics -*, volume 1, pages 63–70, Morristown, NJ, USA, 2002. Association for Computational Linguistics. doi: 10.3115/1118108.1118117. URL <http://portal.acm.org/citation.cfm?doid=1118108.1118117>.
- G. J. Martin. Balls in hyperbolic manifolds. *Journal of the London mathematical society*, 2(2):257–264, 1989.
- T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *CoRR*, abs/1310.4546, 2013. URL <http://arxiv.org/abs/1310.4546>.
- A. Naik and H. Rangwala. *Large Scale Hierarchical Classification : State of the Art*. 2018. ISBN 9783030016197.
- J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng. Multimodal deep learning. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 689–696, 2011.
- M. Nickel and D. Kiela. Poincaré Embeddings for Learning Hierarchical Representations. In *Neural Information Processing Systems Proceedings*, pages 6338–6347, 2017.
- M. Nickel and D. Kiela. Learning Continuous Hierarchies in the Lorentz Model of Hyperbolic Geometry. In *International Conference on Machine Learning*, jul 2018. URL <http://proceedings.mlr.press/v80/nickel18a.html><http://arxiv.org/abs/1806.03417>.
- M. Nickel, V. Tresp, and H. Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on Machine Learning*, pages 809–816, 2011.
- F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011. ISSN 1533-7928. URL <http://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html>.

- B. Perozzi, R. Al-Rfou, and S. Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.
- A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.
- A. Rudi, C. Ciliberto, G. M. Marconi, and L. Rosasco. Manifold Structured Prediction. (Nips):1–18, 2018. URL <http://arxiv.org/abs/1806.09908>.
- L. Song, K. Fukumizu, and A. Gretton. Kernel embeddings of conditional distributions: A unified kernel framework for nonparametric inference in graphical models. *IEEE Signal Processing Magazine*, 30(4):98–111, 2013.
- F. Steinke and M. Hein. Non-parametric regression between manifolds. In *Advances in Neural Information Processing Systems*, pages 1561–1568, 2009.
- F. Steinke, M. Hein, and B. Schölkopf. Nonparametric regression between general riemannian manifolds. *SIAM Journal on Imaging Sciences*, 3(3):527–563, 2010.
- A. Tifrea, G. Bécigneul, and O.-E. Ganea. Poincaré glove: Hyperbolic word embeddings. *arXiv preprint arXiv:1810.06546*, 2018.
- P. Vateekul, M. Kubat, and K. Sarinnapakorn. Top-down optimized svms for hierarchical multi-label classification: A case study in gene function prediction. *Intelligent Data Analysis*, 2012.
- A. Veit, M. Nickel, S. Belongie, and L. van der Maaten. Separating self-expression and visual content in hashtag supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5919–5927, 2018.
- R. C. Wilson, E. R. Hancock, E. Pekalska, and R. P. Duin. Spherical and hyperbolic embeddings of data. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2255–2269, 2014.
- T. Xiao, H. Li, W. Ouyang, and X. Wang. Learning deep feature representations with domain guided dropout for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1249–1258, 2016.
- J. Xu and G. Durrett. Spherical latent spaces for stable variational autoencoders. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4503–4513, 2018.

Learning with SGD and Random Features

Luigi Carratino*
University of Genoa,
Genoa, Italy

Alessandro Rudi
INRIA – Sierra Project-team,
École Normale Supérieure, Paris

Lorenzo Rosasco
University of Genoa,
LCSL – IIT & MIT

Abstract

Sketching and stochastic gradient methods are arguably the most common techniques to derive efficient large scale learning algorithms. In this paper, we investigate their application in the context of nonparametric statistical learning. More precisely, we study the estimator defined by stochastic gradient with mini batches and random features. The latter can be seen as form of nonlinear sketching and used to define approximate kernel methods. The considered estimator is not explicitly penalized/constrained and regularization is implicit. Indeed, our study highlights how different parameters, such as number of features, iterations, step-size and mini-batch size control the learning properties of the solutions. We do this by deriving optimal finite sample bounds, under standard assumptions. The obtained results are corroborated and illustrated by numerical experiments.

1 Introduction

The interplay between statistical and computational performances is key for modern machine learning algorithms [1]. On the one hand, the ultimate goal is to achieve the best possible prediction error. On the other hand, budgeted computational resources need be factored in, while designing algorithms. Indeed, time and especially memory requirements are unavoidable constraints, especially in large-scale problems.

In this view, stochastic gradient methods [2] and sketching techniques [3] have emerged as fundamental algorithmic tools. Stochastic gradient methods allow to process data points individually, or in small batches, keeping good convergence rates, while reducing computational complexity [4]. Sketching techniques allow to reduce data-dimensionality, hence memory requirements, by random projections [3]. Combining the benefits of both methods is tempting and indeed it has attracted much attention, see [5] and references therein.

In this paper, we investigate these ideas for nonparametric learning. Within a least squares framework, we consider an estimator defined by mini-batched stochastic gradients and random features [6]. The latter are typically defined by nonlinear sketching: random projections followed by a component-wise nonlinearity [3]. They can be seen as shallow networks with random weights [7], but also as approximate kernel methods [8]. Indeed, random features provide a standard approach to overcome the memory bottleneck that prevents large-scale applications of kernel methods. The theory of reproducing kernel Hilbert spaces [9] provides a rigorous mathematical framework to study the properties of stochastic gradient method with random features. The approach we consider is not based on penalizations or explicit constraints; regularization is implicit and controlled by different parameters. In particular, our analysis shows how the number of random features, iterations, step-size and mini-batch size control the stability and learning properties of the solution. By deriving finite sample bounds, we investigate how optimal learning rates can be achieved with different parameter choices. In particular, we show that similarly to ridge regression [10], a number of random features proportional to the square root of the number of samples suffices for $O(1/\sqrt{n})$ error bounds.

*Email: luigi.carratino@dibris.unige.it

The rest of the paper is organized as follows. We introduce problem, background and the proposed algorithm in section 2. We present our main results in section 3 and illustrate numerical experiments in section 4.

Notation: For any $T \in \mathbb{N}_+$ we denote by $[T]$ the set $\{1, \dots, T\}$, for any $a, b \in \mathbb{R}$ we denote by $a \vee b$ the maximum between a and b and with $\wedge$ the minimum. For any linear operator A and $\lambda \in \mathbb{R}$ we denote by A_λ the operator $(A + \lambda I)$ if not explicitly defined differently. When A is a bounded self-adjoint linear operator on a Hilbert space, we denote by $\lambda_{\max}(A)$ the biggest eigenvalue of A .

2 Learning with Stochastic Gradients and Random Features

In this section, we present the setting and discuss the learning algorithm we consider.

The problem we study is supervised statistical learning with squared loss [11]. Given a probability space $X \times \mathbb{R}$ with distribution ρ the problem is to solve

$$\min_f \mathcal{E}(f), \quad \mathcal{E}(f) = \int (f(x) - y)^2 d\rho(x, y), \quad (1)$$

given only a training set of pairs $(x_i, y_i)_i^n \in (X \times \mathbb{R})^n$, $n \in \mathbb{N}$, sampled independently according to ρ . Here the minimum is intended over all functions for which the above integral is well defined and ρ is assumed fixed but known only through the samples.

In practice, the search for a solution needs to be restricted to a suitable space of hypothesis to allow efficient computations and reliable estimation [12]. In this paper, we consider functions of the form

$$f(x) = \langle w, \phi_M(x) \rangle, \quad \forall x \in X, \quad (2)$$

where $w \in \mathbb{R}^M$ and $\phi_M : X \rightarrow \mathbb{R}^M$, $M \in \mathbb{N}$, denotes a family of finite dimensional feature maps, see below. Further, we consider a mini-batch stochastic gradient method to estimate the coefficients from data,

$$\hat{w}_1 = 0; \quad \hat{w}_{t+1} = \hat{w}_t - \gamma_t \frac{1}{b} \sum_{i=b(t-1)+1}^{bt} (\langle \hat{w}_t, \phi_M(x_{j_i}) \rangle - y_{j_i}) \phi_M(x_{j_i}), \quad t = 1, \dots, T. \quad (3)$$

Here $T \in \mathbb{N}$ is the number of iterations and $J = \{j_1, \dots, j_{bT}\}$ denotes the strategy to select training set points. In particular, in this work we assume the points to be drawn uniformly at random with replacement. Note that given this sampling strategy, one *pass* over the data is reached on average after $\lceil \frac{n}{b} \rceil$ iterations. Our analysis allows to consider multiple as well as single passes. For $b = 1$ the above algorithm reduces to a simple stochastic gradient iteration. For $b > 1$ it is a mini-batch version, where b points are used in each iteration to compute a gradient estimate. The parameter γ_t is the step-size.

The algorithm requires specifying different parameters. In the following, we study how their choice is related and can be performed to achieve optimal learning bounds. Before doing this, we further discuss the class of feature maps we consider.

2.1 From Sketching to Random Features, from Shallow Nets to Kernels

In this paper, we are interested in a particular class of feature maps, namely random features [6]. A simple example is obtained by sketching the input data. Assume $X \subseteq \mathbb{R}^D$ and

$$\phi_M(x) = (\langle x, s_1 \rangle, \dots, \langle x, s_M \rangle),$$

where $s_1, \dots, s_M \in \mathbb{R}^D$ is a set of identical and independent random vectors [13]. More generally, we can consider features obtained by nonlinear sketching

$$\phi_M(x) = (\sigma(\langle x, s_1 \rangle), \dots, \sigma(\langle x, s_M \rangle)), \quad (4)$$

where $\sigma : \mathbb{R} \rightarrow \mathbb{R}$ is a nonlinear function, for example $\sigma(a) = \cos(a)$ [6], $\sigma(a) = |a|_+ = \max(a, 0)$, $a \in \mathbb{R}$ [7]. If we write the corresponding function [2] explicitly, we get

$$f(x) = \sum_{j=1}^M w^j \sigma(\langle s_j, x \rangle), \quad \forall x \in X. \quad (5)$$

that is as shallow neural nets with random weights [7] (offsets can be added easily). For many examples of random features the inner product,

$$\langle \phi_M(x), \phi_M(x') \rangle = \sum_{j=1}^M \sigma(\langle x, s_j \rangle) \sigma(\langle x', s_j \rangle), \quad (6)$$

can be shown to converge to a corresponding positive definite kernel k as M tends to infinity [6, 14]. We now show some examples of kernels determined by specific choices of random features.

Example 1 (Random features and kernel). Let $\sigma(a) = \cos(a)$ and consider $(\langle x, s \rangle + b)$ in place of the inner product $\langle x, s \rangle$, with s drawn from a standard Gaussian distribution with variance σ^2 , and b uniformly from $[0, 2\pi]$. These are the so called Fourier random features and recover the Gaussian kernel $k(x, x') = e^{-\|x-x'\|^2/2\sigma^2}$ [6] as M increases. If instead $\sigma(a) = a$, and the s is sampled according to a standard Gaussian the linear kernel $k(x, x') = \sigma^2 \langle x, x' \rangle$ is recovered in the limit. [15].

These last observations allow to establish a connection with kernel methods [10] and the theory of reproducing kernel Hilbert spaces [9]. Recall that a reproducing kernel Hilbert space $\mathcal{H}$ is a Hilbert space of functions for which there is a symmetric positive definite function $k : X \times X \rightarrow \mathbb{R}$ called reproducing kernel, such that $k(x, \cdot) \in \mathcal{H}$ and $\langle f, k(x, \cdot) \rangle = f(x)$ for all $f \in \mathcal{H}, x \in X$. It is also useful to recall that k is a reproducing kernel if and only if there exists a Hilbert (feature) space $\mathcal{F}$ and a (feature) map $\phi : X \rightarrow \mathcal{F}$ such that

$$k(x, x') = \langle \phi(x), \phi(x') \rangle, \quad \forall x, x' \in X, \quad (7)$$

where $\mathcal{F}$ can be infinite dimensional.

The connection to RKHS is interesting in at least two ways. First, it allows to use results and techniques from the theory of RKHS to analyze random features. Second, it shows that random features can be seen as an approach to derive scalable kernel methods [10]. Indeed, kernel methods have complexity at least quadratic in the number of points, while random features have complexity which is typically linear in the number of points. From this point of view, the intuition behind random features is to relax [7] considering

$$k(x, x') \approx \langle \phi_M(x), \phi_M(x') \rangle, \quad \forall x, x' \in X. \quad (8)$$

where ϕ_M is finite dimensional.

2.2 Computational complexity

If we assume the computation of the feature map $\phi_M(x)$ to have a constant cost, the iteration (3) requires $O(M)$ operations per iteration for $b = 1$, that is $O(Mn)$ for one pass $T = n$. Note that for $b > 1$ each iteration cost $O(Mb)$ but one pass corresponds to $\lceil \frac{n}{b} \rceil$ iterations so that the cost for one pass is again $O(Mn)$. A main advantage of mini-batching is that gradient computations can be easily parallelized. In the multiple pass case, the time complexity after T iterations is $O(MbT)$.

Computing the feature map $\phi_M(x)$ requires to compute M random features. The computation of one random feature does not depend on n , but only on the input space X . If for example we assume $X \subseteq \mathbb{R}^D$ and consider random features defined as in the previous section, computing $\phi_M(x)$ requires M random projections of D dimensional vectors [6], for a total time complexity of $O(MD)$ for evaluating the feature map at one point. For different input spaces and different types of random features computational cost may differ, see for example Orthogonal Random Features [16] or Fastfood [17] where the cost is reduced from $O(MD)$ to $O(M \log D)$. Note that the analysis presented in his paper holds for random features which are independent, while Orthogonal and Fastfood random features are dependent. Although it should be possible to extend our analysis for Orthogonal and Fastfood random features, further work is needed. To simplify the discussion, in the following we treat the complexity of $\phi_M(x)$ to be $O(M)$.

One of the advantages of random features is that each $\phi_M(x)$ can be computed online at each iteration, preserving $O(MbT)$ as the time complexity of the algorithm (3). Computing $\phi_M(x)$ online also reduces memory requirements. Indeed the space complexity of the algorithm (3) is $O(Mb)$ if the mini-batches are computed in parallel, or $O(M)$ if computed sequentially.

²For all $x_1, \dots, x_n$ the matrix with entries $k(x_i, x_j)$, $i, j = 1, \dots, n$ is positive semi-definite.

2.3 Related approaches

We comment on the connection to related algorithms. Random features are typically used within an empirical risk minimization framework [18]. Results considering convex Lipschitz loss functions and ℓ_∞ constraints are given in [19], while [20] considers ℓ_2 constraints. A ridge regression framework is considered in [8], where it is shown that it is possible to achieve optimal statistical guarantees with a number of random features in the order of $\sqrt{n}$. The combination of random features and gradient methods is less explored. A stochastic coordinate descent approach is considered in [21], see also [22, 23]. A related approach is based on subsampling and is often called Nyström method [24, 25]. Here a shallow network is defined considering a nonlinearity which is a positive definite kernel, and weights chosen as a subset of training set points. This idea can be used within a penalized empirical risk minimization framework [26, 27, 28] but also considering gradient [29, 30] and stochastic gradient [31] techniques. An empirical comparison between Nyström method, random features and full kernel method is given in [23], where the empirical risk minimization problem is solved by block coordinate descent. Note that numerous works have combined stochastic gradient and kernel methods with no random projections approximation [32, 33, 34, 35, 36, 5]. The above list of references is only partial and focusing on papers providing theoretical analysis. In the following, after stating our main results we provide a further quantitative comparison with related results.

3 Main Results

In this section, we first discuss our main results under basic assumptions and then more refined results under further conditions.

3.1 Worst case results

Our results apply to a general class of random features described by the following assumption.

Assumption 1. Let (Ω, π) be a probability space, $\psi : X \times \Omega \rightarrow \mathbb{R}$ and for all $x \in X$,

$$\phi_M(x) = \frac{1}{\sqrt{M}} (\psi(x, \omega_1), \dots, \psi(x, \omega_M)), \quad (9)$$

where $\omega_1, \dots, \omega_M \in \Omega$ are sampled independently according to π .

The above class of random features cover all the examples described in section 2.1 as well as many others, see [8, 20] and references therein. Next we introduce the positive definite kernel defined by the above random features. Let $k : X \times X \rightarrow \mathbb{R}$ be defined by

$$k(x, x') = \int \psi(x, \omega) \psi(x', \omega) d\pi(\omega), \quad \forall x, x' \in X.$$

It is easy to check that k is a symmetric and positive definite kernel. To control basic properties of the induced kernel (continuity, boundedness), we require the following assumption, which is again satisfied by the examples described in section 2.1 (see also [8, 20] and references therein).

Assumption 2. The function ψ is continuous and there exists $\kappa \geq 1$ such that $|\psi(x, \omega)| \leq \kappa$ for any $x \in X, \omega \in \Omega$.

The kernel introduced above allows to compare random feature maps of different size and to express the regularity of the largest function class they induce. In particular, we require a standard assumption in the context of non-parametric regression (see [11]), which consists in assuming a minimum for the expected risk, over the space of functions induced by the kernel.

Assumption 3. If $\mathcal{H}$ is the RKHS with kernel k , there exists $f_{\mathcal{H}} \in \mathcal{H}$ such that

$$\mathcal{E}(f_{\mathcal{H}}) = \inf_{f \in \mathcal{H}} \mathcal{E}(f).$$

To conclude, we need some basic assumption on the data distribution. For all $x \in X$, we denote by $\rho(y|x)$ the conditional probability of ρ and by ρ_X the corresponding marginal probability on X . We need a standard moment assumption to derive probabilistic results.

Assumption 4. For any $x \in X$

$$\int_{\mathcal{Y}} y^{2l} d\rho(y|x) \leq l! B^l p, \quad \forall l \in \mathbb{N} \quad (10)$$

for constants $B \in (0, \infty)$ and $p \in (1, \infty)$, ρ_X -almost surely.

The above assumption holds when y is bounded, sub-gaussian or sub-exponential.

The next theorem corresponds to our first main result. Recall that, the *excess risk* for a given estimator $\hat{f}$ is defined as

$$\mathcal{E}(\hat{f}) - \mathcal{E}(f_{\mathcal{H}}),$$

and is a standard error measure in statistical machine learning [11, 18]. In the following theorem, we control the *excess risk* of the estimator with respect to the number of points, the number of RF, the step size, the mini-batch size and the number of iterations. We let $\hat{f}_{t+1} = \langle \hat{w}_{t+1}, \phi_M(\cdot) \rangle$, with $\hat{w}_{t+1}$ as in [3].

Theorem 1. Let $n, M \in \mathbb{N}_+$, $\delta \in (0, 1)$ and $t \in [T]$. Under Assumptions [1] to [4] for $b \in [n]$, $\gamma_t = \gamma$ s.t. $\gamma \leq \frac{n}{9T \log \frac{2}{\delta}} \wedge \frac{1}{8(1+\log T)}$, $n \geq 32 \log^2 \frac{2}{\delta}$ and $M \gtrsim \gamma T$ the following holds with probability at least $1 - \delta$:

$$\mathbb{E}_J[\mathcal{E}(\hat{f}_{t+1})] - \mathcal{E}(f_{\mathcal{H}}) \lesssim \frac{\gamma}{b} + \left(\frac{\gamma t}{M} + 1 \right) \frac{\gamma t \log \frac{1}{\delta}}{n} + \frac{\log \frac{1}{\delta}}{M} + \frac{1}{\gamma t}. \quad (11)$$

The above theorem bounds the excess risk with a sum of terms controlled by the different parameters. The following corollary shows how these parameters can be chosen to derive finite sample bounds.

Corollary 1. Under the same assumptions of Theorem [1] for one of the following conditions

(c_{1.1}). $b = 1$, $\gamma \simeq \frac{1}{n}$, and $T = n\sqrt{n}$ iterations ($\sqrt{n}$ passes over the data);

(c_{1.2}). $b = 1$, $\gamma \simeq \frac{1}{\sqrt{n}}$, and $T = n$ iterations (1 pass over the data);

(c_{1.3}). $b = \sqrt{n}$, $\gamma \simeq 1$, and $T = \sqrt{n}$ iterations (1 pass over the data);

(c_{1.4}). $b = n$, $\gamma \simeq 1$, and $T = \sqrt{n}$ iterations ($\sqrt{n}$ passes over the data);

a number

$$M = \tilde{O}(\sqrt{n}) \quad (12)$$

of random features is sufficient to guarantee with high probability that

$$\mathbb{E}_J[\mathcal{E}(\hat{f}_T)] - \mathcal{E}(f_{\mathcal{H}}) \lesssim \frac{1}{\sqrt{n}}. \quad (13)$$

The above learning rate is the same achieved by an exact kernel ridge regression (KRR) estimator [11, 37, 38], which has been proved to be optimal in a minimax sense [11] under the same assumptions. Further, the number of random features required to achieve this bound is the same as the kernel ridge regression estimator with random features [8]. Notice that, for the limit case where the number of random features grows to infinity for Corollary [1] under conditions (c_{1.2}) and (c_{1.3}) we recover the same results for one pass SGD of [39], [40]. In this limit, our results are also related to those in [41]. Here, however, averaging of the iterates is used to achieve larger step-sizes.

Note that conditions (c_{1.1}) and (c_{1.2}) in the corollary above show that, when no mini-batches are used ($b = 1$) and $\frac{1}{n} \leq \gamma \leq \frac{1}{\sqrt{n}}$, then the step-size γ determines the number of passes over the data required for optimal generalization. In particular the number of passes varies from constant, when $\gamma = \frac{1}{\sqrt{n}}$, to $\sqrt{n}$, when $\gamma = \frac{1}{n}$. In order to increase the step-size over $\frac{1}{\sqrt{n}}$ the algorithm needs to be run with mini-batches. The step-size can then be increased up to a constant if b is chosen equal to $\sqrt{n}$ (condition (c_{1.3})), requiring the same number of passes over the data of the setting (c_{1.2}). Interestingly condition (c_{1.4}) shows that increasing the mini-batch size over $\sqrt{n}$ does not allow to take larger step-sizes, while it seems to increase the number of passes over the data required to reach optimality.

We now compare the time complexity of algorithm [3] with some closely related methods which

achieve the same optimal rate of $\frac{1}{\sqrt{n}}$. Computing the classical KRR estimator [11] has a complexity of roughly $O(n^3)$ in time and $O(n^2)$ in memory. Lowering this computational cost is possible with random projection techniques. Both random features and Nyström method on KRR [8, 26] lower the time complexity to $O(n^2)$ and the memory complexity to $O(n\sqrt{n})$ preserving the statistical accuracy. The same time complexity is achieved by stochastic gradient method solving the full kernel method [33, 36], but with the higher space complexity of $O(n^2)$. The combination of the stochastic gradient iteration, random features and mini-batches allows our algorithm to achieve a complexity of $O(n\sqrt{n})$ in time and $O(n)$ in space for certain choices of the free parameters (like $(c_{1,2})$ and $(c_{1,3})$). Note that these time and memory complexity are lower with respect to those of stochastic gradient with mini-batches and Nyström approximation which are $O(n^2)$ and $O(n)$ respectively [31]. A method with similar complexity to SGD with RF is FALKON [30]. This method has indeed a time complexity of $O(n\sqrt{n} \log(n))$ and $O(n)$ space complexity. This method blends together Nyström approximation, a sketched preconditioner and conjugate gradient.

3.2 Refined analysis and fast rates

We next discuss how the above results can be refined under an additional regularity assumption. We need some preliminary definitions. Let $\mathcal{H}$ be the RKHS defined by k , and $L : L^2(X, \rho_X) \rightarrow L^2(X, \rho_X)$ the integral operator

$$Lf(x) = \int k(x, x')f(x')d\rho_X(x'), \quad \forall f \in L^2(X, \rho_X), x \in X,$$

where $L^2(X, \rho_X) = \{f : X \rightarrow \mathbb{R} : \|f\|_\rho^2 = \int |f|^2 d\rho_X < \infty\}$. The above operator is symmetric and positive definite. Moreover, Assumption 1 ensures that the kernel is bounded, which in turn ensures L is trace class, hence compact [18].

Assumption 5. For any $\lambda > 0$, define the effective dimension as $\mathcal{N}(\lambda) = \text{Tr}((L + \lambda I)^{-1}L)$, and assume there exist $Q > 0$ and $\alpha \in [0, 1]$ such that

$$\mathcal{N}(\lambda) \leq Q^2 \lambda^{-\alpha}. \quad (14)$$

Moreover, assume there exists $r \geq 1/2$ and $g \in L^2(X, \rho_X)$ such that

$$f_{\mathcal{H}}(x) = (L^r g)(x). \quad (15)$$

Condition (14) describes the *capacity/complexity* of the RKHS $\mathcal{H}$ and the measure ρ . It is equivalent to classic entropy/covering number conditions, see e.g. [18]. The case $\alpha = 1$ corresponds to making no assumptions on the kernel, and reduces to the worst case analysis in the previous section. The smaller is α the more stringent is the capacity condition. A classic example is considering $X = \mathbb{R}^D$ with $d\rho_X(x) = p(x)dx$, where p is a probability density, strictly positive and bounded away from zero, and $\mathcal{H}$ to be a Sobolev space with smoothness $s > D/2$. Indeed, in this case $\alpha = D/2s$ and classical nonparametric statistics assumptions are recovered as a special case. Note that in particular the worst case is $s = D/2$. Condition (15) is a regularity condition commonly used in approximation theory to control the bias of the estimator [42].

The following theorem is a refined version of Theorem 1 where we also consider the above *capacity* condition (Assumption 5).

Theorem 2. Let $n, M \in \mathbb{N}_+$, $\delta \in (0, 1)$ and $t \in [T]$, under Assumptions 1 to 4, for $b \in [n]$, $\gamma_t = \gamma$ s.t. $\gamma \leq \frac{n}{9T \log \frac{n}{\delta}} \wedge \frac{1}{8(1+\log T)}$, $n \geq 32 \log^2 \frac{2}{\delta}$ and $M \gtrsim \gamma T$ the following holds with high probability:

$$\mathbb{E}_J[\mathcal{E}(\hat{f}_{t+1})] - \mathcal{E}(f_{\mathcal{H}}) \lesssim \frac{\gamma}{b} + \left(\frac{\gamma t}{M} + 1\right) \frac{\mathcal{N}\left(\frac{1}{\gamma t}\right) \log \frac{1}{\delta}}{n} + \frac{\mathcal{N}\left(\frac{1}{\gamma t}\right)^{2r-1} \log \frac{1}{\delta}}{M(\gamma t)^{2r-1}} + \left(\frac{1}{\gamma t}\right)^{2r}. \quad (16)$$

The main difference is the presence of the effective dimension providing a sharper control of the stability of the considered estimator. As before, explicit learning bounds can be derived considering different parameter settings.

Corollary 2. Under the same assumptions of Theorem 2 for one of the following conditions

(c_{2,1}). $b = 1$, $\gamma \simeq n^{-1}$, and $T = n^{\frac{2r+\alpha+1}{2r+\alpha}}$ iterations ($n^{\frac{1}{2r+\alpha}}$ passes over the data);

(c_{2.2}). $b = 1$, $\gamma \simeq n^{-\frac{2r}{2r+\alpha}}$, and $T = n^{\frac{2r+1}{2r+\alpha}}$ iterations ($n^{\frac{1-\alpha}{2r+\alpha}}$ passes over the data);

(c_{2.3}). $b = n^{\frac{2r}{2r+\alpha}}$, $\gamma \simeq 1$, and $T = n^{\frac{1}{2r+\alpha}}$ iterations ($n^{\frac{1-\alpha}{2r+\alpha}}$ passes over the data);

(c_{2.4}). $b = n$, $\gamma \simeq 1$, and $T = n^{\frac{1}{2r+\alpha}}$ iterations ($n^{\frac{1}{2r+\alpha}}$ passes over the data);

a number

$$M = \tilde{O}(n^{\frac{1+\alpha(2r-1)}{2r+\alpha}}) \quad (17)$$

of random features suffices to guarantee with high probability that

$$\mathbb{E}_J[\mathcal{E}(\hat{w}_T)] - \mathcal{E}(f_{\mathcal{H}}) \lesssim n^{-\frac{2r}{2r+\alpha}}. \quad (18)$$

The corollary above shows that multi-pass SGD achieves a learning rate that is the same as kernel ridge regression under the regularity assumption [5] and is again minimax optimal (see [11]). Moreover, we obtain the minimax optimal rate with the same number of random features required for ridge regression with random features [8] under the same assumptions. Finally, when the number of random features goes to infinity we also recover the results for the infinite dimensional case of the single-pass and multiple pass stochastic gradient method [33].

It is worth noting that, under the additional regularity assumption [5], the number of both random features and passes over the data sufficient for optimal learning rates increase with respect to the one required in the worst case (see Corollary [1]). The same effect occurs in the context of ridge regression with random features as noted in [8]. In this latter paper, it is observed that this issue tackled can be using more refined, possibly more costly, sampling schemes [20].

Finally, we present a general result from which all our previous results follow as special cases. We consider a more general setting where we allow decreasing step-sizes.

Theorem 3. Let $n, M, T \in \mathbb{N}$, $b \in [n]$ and $\gamma > 0$. Let $\delta \in (0, 1)$ and $\hat{w}_{t+1}$ be the estimator in Eq. (3) with $\gamma_t = \gamma \kappa^{-2} t^{-\theta}$ and $\theta \in [0, 1[$. Under Assumptions [1] to [4] when $n \geq 32 \log^2 \frac{2}{\delta}$ and

$$\gamma \leq \frac{n}{9T^{1-\theta} \log \frac{n}{\delta}} \wedge \begin{cases} \frac{\theta \wedge (1-\theta)}{7} & \theta \in]0, 1[\\ \frac{1}{8(1+\log T)} & \text{otherwise,} \end{cases} \quad (19)$$

moreover

$$M \geq (4 + 18\gamma T^{1-\theta}) \log \frac{12\gamma T^{1-\theta}}{\delta}, \quad (20)$$

then, for any $t \in [T]$ the following holds with probability at least $1 - 9\delta$

$$\mathbb{E}_J[\mathcal{E}(\hat{w}_{t+1})] - \inf_{w \in \mathcal{F}} \mathcal{E}(w) \leq c_1 \frac{\gamma}{bt^{\min(\theta, 1-\theta)}} (\log t \vee 1) \quad (21)$$

$$+ \left(c_2 + c_3 \frac{1}{M} \log \frac{M}{\delta} (\gamma t^{1-\theta} \vee 1) \right) \frac{\mathcal{N}\left(\frac{\kappa^2}{\gamma t^{1-\theta}}\right)}{n} (\log^2(t) \vee 1) \log^2 \frac{4}{\delta} \quad (22)$$

$$+ c_4 \left(\frac{\mathcal{N}\left(\frac{\kappa^2}{\gamma t^{1-\theta}}\right)^{2r-1} \log \frac{2}{\delta}}{M(\gamma t^{1-\theta} \kappa^{-2})^{2r-1}} \log^{2-2r} (11\gamma t^{1-\theta}) + \left(\frac{1}{\gamma t^{1-\theta}} \right)^{2r} \right), \quad (23)$$

with c_1, c_2, c_3, c_4 constants which do not depend on $b, \gamma, n, t, M, \delta$.

We note that as the number of random features M goes to infinity, we recover the same bound of [33] for decreasing step-sizes. Moreover, the above theorem shows that there is no apparent gain in using a decreasing stepsize (i.e. $\theta > 0$) with respect to the regimes identified in Corollaries [1] and [2].

3.3 Sketch of the Proof

In this section, we sketch the main ideas in the proof. We relate $\hat{f}_t$ and $f_{\mathcal{H}}$ introducing several intermediate functions. In particular, the following iterations are useful,

$$\hat{v}_1 = 0; \quad \hat{v}_{t+1} = \hat{v}_t - \gamma_t \sum_{i=1}^n (\langle \hat{v}_t, \phi_M(x_i) \rangle - y_i) \phi_M(x_i), \quad \forall t \in [T]. \quad (24)$$

$$\tilde{v}_1 = 0; \quad \tilde{v}_{t+1} = \tilde{v}_t - \gamma_t \int_X (\langle \tilde{v}_t, \phi_M(x) \rangle - y) \phi_M(x) d\rho(x, y), \quad \forall t \in [T]. \quad (25)$$

$$v_1 = 0; \quad v_{t+1} = v_t - \gamma_t \int_X (\langle v_t, \phi_M(x) \rangle - f_{\mathcal{H}}(x)) \phi_M(x) d\rho_X(x), \quad \forall t \in [T]. \quad (26)$$

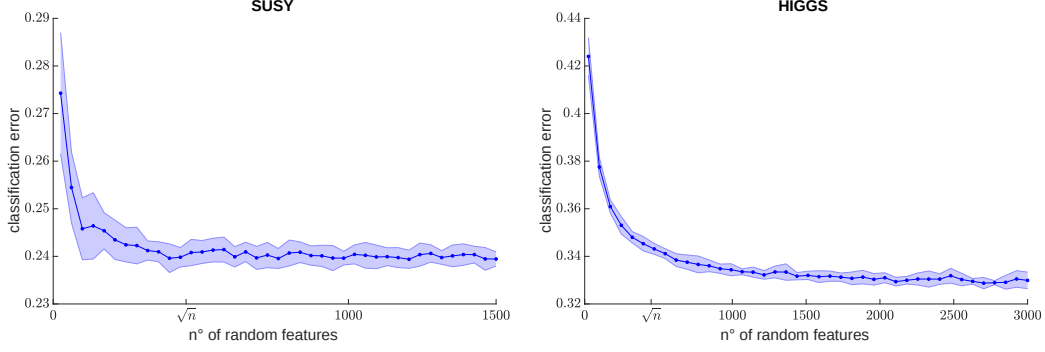


Figure 1: Classification error of SUSY (left) and HIGGS (right) datasets as the n^o of random features varies

Further, we let

$$\tilde{u}_\lambda = \operatorname{argmin}_{u \in \mathbb{R}^M} \int_X (\langle u, \phi_M(x) \rangle - f_{\mathcal{H}}(x))^2 d\rho_X(x) + \lambda \|u\|^2, \quad \lambda > 0, \quad (27)$$

$$u_\lambda = \operatorname{argmin}_{u \in \mathcal{F}} \int_X (\langle u, \phi(x) \rangle - y)^2 d\rho(x, y) + \lambda \|u\|^2, \quad \lambda > 0, \quad (28)$$

where $(\mathcal{F}, \phi)$ are feature space and feature map associated to the kernel k . The first three vectors are defined by the random features and can be seen as an empirical and population batch gradient descent iterations. The last two vectors can be seen as a population version of ridge regression defined by the random features and the feature map ϕ , respectively.

Since the above objects (24), (25), (26), (27), (28) belong to different spaces, instead of comparing them directly we compare the functions in $L^2(X, \rho_X)$ associated to them, letting

$$\hat{g}_t = \langle \hat{v}_t, \phi_M(\cdot) \rangle, \quad \tilde{g}_t = \langle \tilde{v}_t, \phi_M(\cdot) \rangle, \quad g_t = \langle v_t, \phi_M(\cdot) \rangle, \quad \tilde{g}_\lambda = \langle \tilde{u}_\lambda, \phi_M(\cdot) \rangle, \quad g_\lambda = \langle u_\lambda, \phi(\cdot) \rangle.$$

Since it is well known [11] that

$$\mathcal{E}(f) - \mathcal{E}(f_{\mathcal{H}}) = \|f - f_{\mathcal{H}}\|_{\rho}^2,$$

we than can consider the following decomposition

$$\hat{f}_t - f_{\mathcal{H}} = \hat{f}_t - \hat{g}_t \quad (29)$$

$$+ \hat{g}_t - \tilde{g}_t \quad (30)$$

$$+ \tilde{g}_t - g_t \quad (31)$$

$$+ g_t - \tilde{g}_\lambda \quad (32)$$

$$+ \tilde{g}_\lambda - g_\lambda \quad (33)$$

$$+ g_\lambda - f_{\mathcal{H}}. \quad (34)$$

The first two terms control how SGD deviates from the batch gradient descent and the effect of noise and sampling. They are studied in Lemma 1, 2, 3, 4, 5, 6 in the Appendix, borrowing and adapting ideas from [33, 36, 8]. The following terms account for the approximation properties of random features and the bias of the algorithm. Here the basic idea and novel result is the study of how the population gradient decent and ridge regression are related (32) (Lemma 9 in the Appendix). Then, results from the the analysis of ridge regression with random features are used [8].

4 Experiments

We study the behavior of the SGD with RF algorithm on subsets of $n = 2 \times 10^5$ points of the SUSY [3] and HIGGS [4] datasets [43]. The measures we show in the following experiments are an average over 10 repetitions of the algorithm. Further, we consider random Fourier features that

<https://archive.ics.uci.edu/ml/datasets/SUSY>

<https://archive.ics.uci.edu/ml/datasets/HIGGS>

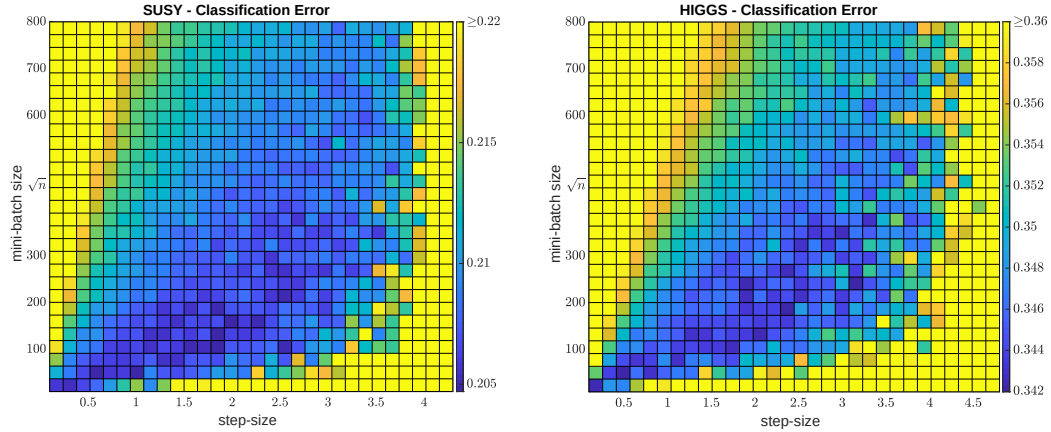


Figure 2: Classification error of SUSY (left) and HIGGS (right) datasets as step-size and mini-batch size vary

are known to approximate translation invariant kernels [6]. We use random features of the form $\psi(x, \omega) = \cos(w^T x + q)$, with $\omega := (w, q)$, w sampled according to the normal distribution and q sampled uniformly at random between 0 and 2π . Note that the random features defined this way satisfy Assumption 2.

Our theoretical analysis suggests that only a number of RF of the order of $\sqrt{n}$ suffices to gain optimal learning properties. Hence we study how the number of RF affect the accuracy of the algorithm on test sets of 10^5 points. In Figure 3.3 we show the classification error after 5 passes over the data of SGD with RF as the number of RF increases, with a fixed batch size of $\sqrt{n}$ and a step-size of 1. We can observe that over a certain threshold of the order of $\sqrt{n}$, increasing the number of RF does not improve the accuracy, confirming what our theoretical results suggest.

Further, theory suggests that the step-size can be increased as the mini-batch size increases to reach an optimal accuracy, and that after a mini-batch size of the order of $\sqrt{n}$ more than 1 pass over the data is required to reach the same accuracy. We show in Figure 2 the classification error of SGD with RF after 1 pass over the data, with a fixed number of random features $\sqrt{n}$, as mini-batch size and step-size vary, on test sets of 10^5 points. As suggested by theory, to reach the lowest error as the mini-batch size grows the step-size needs to grow as well. Further for mini-batch sizes bigger than $\sqrt{n}$ the lowest error can not be reached in only 1 pass even if increasing the step-size.

5 Conclusions

In this paper we investigate the combination of sketching and stochastic techniques in the context of non-parametric regression. In particular we studied the statistical and computational properties of the estimator defined by stochastic gradient descent with multiple passes, mini-batches and random features. We proved that the estimator achieves optimal statistical properties with a number of random features in the order of $\sqrt{n}$ (with n the number of examples). Moreover we analyzed possible trade-offs between the number of passes, the step and the dimension of the mini-batches showing that there exist different configurations which achieve the same optimal statistical guarantees, with different computational impacts.

Our work can be extended in several ways: First, (a) we can study the effect of combining random features with accelerated/averaged stochastic techniques as [32]. Second, (b) we can extend our analysis to consider more refined assumptions, generalizing [35] to SGD with random features. Additionally, (c) we can study the statistical properties of the considered estimator in the context of classification with the goal of showing fast decay of the classification error, as in [34]. Moreover, (d) we can apply the proposed method in the more general context of least squares frameworks for multitask learning [44, 45] or structured prediction [46, 47, 48], with the goal of obtaining faster algorithms, while retaining strong statistical guarantees. Finally, (e) to integrate our analysis with more refined methods to select the random features analogously to [49, 50] in the context of column sampling.

Acknowledgments.

This material is based upon work supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216, and the Italian Institute of Technology. We gratefully acknowledge the support of NVIDIA Corporation for the donation of the Titan Xp GPUs and the Tesla k40 GPU used for this research. L. R. acknowledges the support of the AFOSR projects FA9550-17-1-0390 and BAA-AFRL-AFOSR-2016-0007 (European Office of Aerospace Research and Development), and the EU H2020-MSCA-RISE project NoMADS - DLV-777826. A. R. acknowledges the support of the European Research Council (grant SEQUOIA 724063).

References

- [1] Alekh Agarwal, Sahand Negahban, and Martin J Wainwright. Stochastic optimization and sparse statistical recovery: Optimal algorithms for high dimensions. In *Advances in Neural Information Processing Systems*, pages 1538–1546, 2012.
- [2] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.
- [3] Haim Avron, Vikas Sindhwani, and David Woodruff. Sketching structured matrices for faster nonlinear regression. In *Advances in neural information processing systems*, pages 2994–3002, 2013.
- [4] Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Advances in neural information processing systems*, pages 161–168, 2008.
- [5] Francesco Orabona. Simultaneous model selection and optimization through parameter-free stochastic learning. In *Advances in Neural Information Processing Systems*, pages 1116–1124, 2014.
- [6] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.
- [7] Youngmin Cho and Lawrence K Saul. Kernel methods for deep learning. In *Advances in neural information processing systems*, pages 342–350, 2009.
- [8] Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. In *Advances in Neural Information Processing Systems 30*, pages 3215–3225, 2017.
- [9] Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404, 1950.
- [10] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. 2002.
- [11] Andrea Caponnetto and Ernesto De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007.
- [12] Luc Devroye, László Györfi, and Gábor Lugosi. *A probabilistic theory of pattern recognition*, volume 31. Springer Science & Business Media, 2013.
- [13] David P Woodruff et al. Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science*, 10(1–2):1–157, 2014.
- [14] Bharath Sriperumbudur and Zoltán Szabó. Optimal rates for random fourier features. In *Advances in Neural Information Processing Systems*, pages 1144–1152, 2015.
- [15] Raffay Hamid, Ying Xiao, Alex Gittens, and Dennis DeCoste. Compact random feature maps. In *International Conference on Machine Learning*, pages 19–27, 2014.
- [16] X Yu Felix, Ananda Theertha Suresh, Krzysztof M Choromanski, Daniel N Holtmann-Rice, and Sanjiv Kumar. Orthogonal random features. In *Advances in Neural Information Processing Systems*, pages 1975–1983, 2016.
- [17] Quoc Le, Tamás Sarlós, and Alex Smola. Fastfood-approximating kernel expansions in loglinear time. In *Proceedings of the international conference on machine learning*, volume 85, 2013.

- [18] Ingo Steinwart and Andreas Christmann. *Support vector machines*. Springer Science & Business Media, 2008.
- [19] Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in neural information processing systems*, pages 1313–1320, 2009.
- [20] Francis Bach. On the equivalence between kernel quadrature rules and random feature expansions. *Journal of Machine Learning Research*, 18(21):1–38, 2017.
- [21] Bo Dai, Bo Xie, Niao He, Yingyu Liang, Anant Raj, Maria-Florina F Balcan, and Le Song. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems*, pages 3041–3049, 2014.
- [22] Junhong Lin and Lorenzo Rosasco. Generalization properties of doubly online learning algorithms. *arXiv preprint arXiv:1707.00577*, 2017.
- [23] Stephen Tu, Rebecca Roelofs, Shivaram Venkataraman, and Benjamin Recht. Large scale kernel learning using block coordinate descent. *arXiv preprint arXiv:1602.05310*, 2016.
- [24] Alex J Smola and Bernhard Schölkopf. Sparse greedy matrix approximation for machine learning. 2000.
- [25] Christopher KI Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In *Advances in neural information processing systems*, pages 682–688, 2001.
- [26] Alessandro Rudi, Raffaello Camoriano, and Lorenzo Rosasco. Less is more: Nyström computational regularization. In *Advances in Neural Information Processing Systems*, pages 1657–1665, 2015.
- [27] Yun Yang, Mert Pilanci, and Martin J Wainwright. Randomized sketches for kernels: Fast and optimal non-parametric regression. *arXiv preprint arXiv:1501.06195*, 2015.
- [28] Ahmed Alaoui and Michael W Mahoney. Fast randomized kernel ridge regression with statistical guarantees. In *Advances in Neural Information Processing Systems*, pages 775–783, 2015.
- [29] Raffaello Camoriano, Tomás Angles, Alessandro Rudi, and Lorenzo Rosasco. Nytro: When subsampling meets early stopping. In *Artificial Intelligence and Statistics*, pages 1403–1411, 2016.
- [30] Alessandro Rudi, Luigi Carratino, and Lorenzo Rosasco. FALKON: An optimal large scale kernel method. In *Advances in Neural Information Processing Systems*, pages 3891–3901, 2017.
- [31] Junhong Lin and Lorenzo Rosasco. Optimal rates for learning with nyström stochastic gradient methods. *arXiv preprint arXiv:1710.07797*, 2017.
- [32] Aymeric Dieuleveut, Nicolas Flammarion, and Francis Bach. Harder, better, faster, stronger convergence rates for least-squares regression. *The Journal of Machine Learning Research*, 18(1):3520–3570, 2017.
- [33] Junhong Lin and Lorenzo Rosasco. Optimal rates for multi-pass stochastic gradient methods. *Journal of Machine Learning Research*, 18(97):1–47, 2017.
- [34] Loucas Pillaud-Vivien, Alessandro Rudi, and Francis Bach. Exponential convergence of testing error for stochastic gradient methods. In *Proceedings of the 31st Conference On Learning Theory*, volume 75, pages 250–296, 2018.
- [35] Loucas Pillaud-Vivien, Alessandro Rudi, and Francis Bach. Statistical optimality of stochastic gradient descent on hard learning problems through multiple passes. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 8125–8135. Curran Associates, Inc., 2018.
- [36] Lorenzo Rosasco and Silvia Villa. Learning with incremental iterative regularization. In *Advances in Neural Information Processing Systems*, pages 1630–1638, 2015.

- [37] Ingo Steinwart, Don R Hush, Clint Scovel, et al. Optimal rates for regularized least squares regression. In *COLT*, 2009.
- [38] Junhong Lin, Alessandro Rudi, Lorenzo Rosasco, and Volkan Cevher. Optimal rates for spectral algorithms with least-squares regression over hilbert spaces. *Applied and Computational Harmonic Analysis*, 2018.
- [39] Ohad Shamir and Tong Zhang. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *International Conference on Machine Learning*, pages 71–79, 2013.
- [40] Ofer Dekel, Ran Gilad-Bachrach, Ohad Shamir, and Lin Xiao. Optimal distributed online prediction using mini-batches. *Journal of Machine Learning Research*, 13(Jan):165–202, 2012.
- [41] Aymeric Dieuleveut, Francis Bach, et al. Nonparametric stochastic approximation with large step-sizes. *The Annals of Statistics*, 44(4):1363–1399, 2016.
- [42] Steve Smale and Ding-Xuan Zhou. Estimating the approximation error in learning theory. *Analysis and Applications*, 1(01):17–41, 2003.
- [43] Pierre Baldi, Peter Sadowski, and Daniel Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5:4308, 2014.
- [44] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.
- [45] Carlo Ciliberto, Alessandro Rudi, Lorenzo Rosasco, and Massimiliano Pontil. Consistent multi-task learning with nonlinear output relations. In *Advances in Neural Information Processing Systems*, pages 1986–1996, 2017.
- [46] Carlo Ciliberto, Lorenzo Rosasco, and Alessandro Rudi. A consistent regularization approach for structured prediction. *Advances in Neural Information Processing Systems 29 (NIPS)*, pages 4412–4420, 2016.
- [47] Anton Osokin, Francis Bach, and Simon Lacoste-Julien. On structured prediction theory with calibrated convex surrogate losses. In *Advances in Neural Information Processing Systems*, pages 302–313, 2017.
- [48] Carlo Ciliberto, Francis Bach, and Alessandro Rudi. Localized structured prediction. *arXiv preprint arXiv:1806.02402*, 2018.
- [49] Petros Drineas, Malik Magdon-Ismael, Michael W Mahoney, and David P Woodruff. Fast approximation of matrix coherence and statistical leverage. *Journal of Machine Learning Research*, 13(Dec):3475–3506, 2012.
- [50] Alessandro Rudi, Daniele Calandriello, Luigi Carratino, and Lorenzo Rosasco. On fast leverage score sampling and optimal learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 5677–5687. Curran Associates, Inc., 2018.
- [51] Felipe Cucker and Steve Smale. On the mathematical foundations of learning. *Bulletin of the American mathematical society*, 39(1):1–49, 2002.
- [52] Ernesto De Vito, Lorenzo Rosasco, Andrea Caponnetto, Umberto De Giovannini, and Francesca Odone. Learning from examples as an inverse problem. *Journal of Machine Learning Research*, 6(May):883–904, 2005.
- [53] Alessandro Rudi, Guillermo D Canas, and Lorenzo Rosasco. On the sample complexity of subspace learning. In *Advances in Neural Information Processing Systems*, pages 2067–2075, 2013.

On Fast Leverage Score Sampling and Optimal Learning

Alessandro Rudi*
INRIA – Sierra team,
ENS, Paris

Daniele Calandriello*
LCSL – IIT & MIT,
Genoa, Italy

Luigi Carratino
University of Genoa,
Genoa, Italy

Lorenzo Rosasco
University of Genoa,
LCSL – IIT & MIT

Abstract

Leverage score sampling provides an appealing way to perform approximate computations for large matrices. Indeed, it allows to derive faithful approximations with a complexity adapted to the problem at hand. Yet, performing leverage scores sampling is a challenge in its own right requiring further approximations. In this paper, we study the problem of leverage score sampling for positive definite matrices defined by a kernel. Our contribution is twofold. First we provide a novel algorithm for leverage score sampling and second, we exploit the proposed method in statistical learning by deriving a novel solver for kernel ridge regression. Our main technical contribution is showing that the proposed algorithms are currently the most efficient and accurate for these problems.

1 Introduction

A variety of machine learning problems require manipulating and performing computations with large matrices that often do not fit memory. In practice, randomized techniques are often employed to reduce the computational burden. Examples include stochastic approximations [1], columns/rows subsampling and more general sketching techniques [2, 3]. One of the simplest approach is uniform column sampling [4, 5], that is replacing the original matrix with a subset of columns chosen uniformly at random. This approach is fast to compute, but the number of columns needed for a prescribed approximation accuracy does not take advantage of the possible low rank structure of the matrix at hand. As discussed in [6], leverage score sampling provides a way to tackle this shortcoming. Here columns are sampled proportionally to suitable weights, called leverage scores (LS) [7, 6]. With this sampling strategy, the number of columns needed for a prescribed accuracy is governed by the so called *effective dimension* which is a natural extension of the notion of rank. Despite these nice properties, performing leverage score sampling provides a challenge in its own right, since it has complexity in the same order of an eigendecomposition of the original matrix. Indeed, much effort has been recently devoted to derive fast and provably accurate algorithms for approximate leverage score sampling [2, 8, 6, 9, 10].

In this paper, we consider these questions in the case of positive semi-definite matrices, central for example in Gaussian processes [11] and kernel methods [12]. Sampling approaches in this context are related to the so called Nyström approximation [13] and Nyström centers selection problem [11], and are widely studied both in practice [4] and in theory [5]. Our contribution is twofold. First, we propose and study BLESS, a novel algorithm for approximate leverage scores sampling. The first solution to this problem is introduced in [6], but has poor approximation guarantees and high time complexity. Improved approximations are achieved by algorithms recently proposed in [8] and [9]. In particular, the approach in [8] can obtain good accuracy and very efficient computations but only as long as distributed resources are available. Our first technical contribution is showing that our algorithm can achieve state of the art accuracy and computational complexity without requiring

*Equal contribution. Respective emails: alessandro.rudi@inria.fr, daniele.calandriello@iit.it

distributed resources. The key idea is to follow a coarse to fine strategy, alternating uniform and leverage scores sampling on sets of increasing size.

Our second, contribution is considering leverage score sampling in statistical learning with least squares. We extend the approach in [14] for efficient kernel ridge regression based on combining fast optimization algorithms (preconditioned conjugate gradient) with uniform sampling. Results in [14] showed that optimal learning bounds can be achieved with a complexity which is $\tilde{O}(n\sqrt{n})$ in time and $\tilde{O}(n)$ space. In this paper, we study the impact of replacing uniform with leverage score sampling. In particular, we prove that the derived method still achieves optimal learning bounds but the time and memory is now $\tilde{O}(nd_{\text{eff}})$, and $\tilde{O}(d_{\text{eff}}^2)$ respectively, where d_{eff} is the effective dimension which is never larger, and possibly much smaller, than $\sqrt{n}$. To the best of our knowledge this is the best currently known computational guarantees for a kernel ridge regression solver.

2 Leverage score sampling with BLESS

After introducing leverage score sampling and previous algorithms, we present our approach and first theoretical results.

2.1 Leverage score sampling

Suppose $\hat{K} \in \mathbb{R}^{n \times n}$ is symmetric and positive semidefinite. A basic question is deriving memory efficient approximation of $\hat{K}$ [4, 8] or related quantities, e.g. approximate projections on its range [9], or associated estimators, as in kernel ridge regression [15, 14]. The eigendecomposition of $\hat{K}$ offers a natural, but computationally demanding solution. Subsampling columns (or rows) is an appealing alternative. A basic approach is uniform sampling, whereas a more refined approach is leverage scores sampling. This latter procedure corresponds to sampling columns with probabilities proportional to the leverage scores

$$\ell(i, \lambda) = \left(\hat{K}(\hat{K} + \lambda n I)^{-1} \right)_{ii}, \quad i \in [n], \quad (1)$$

where $[n] = \{1, \dots, n\}$. The advantage of leverage score sampling, is that potentially very few columns can suffice for the desired approximation. Indeed, letting

$$d_{\infty}(\lambda) = n \max_{i=1, \dots, n} \ell(i, \lambda), \quad d_{\text{eff}}(\lambda) = \sum_{i=1}^n \ell(i, \lambda),$$

for $\lambda > 0$, it is easy to see that $d_{\text{eff}}(\lambda) \leq d_{\infty}(\lambda) \leq 1/\lambda$ for all λ , and previous results show that the number of columns required for accurate approximation are d_{∞} for uniform sampling and d_{eff} for leverage score sampling [5, 6]. However, it is clear from definition (1) that an exact leverage scores computation would require the same order of computations as an eigendecomposition, hence approximations are needed. The accuracy of approximate leverage scores is typically measured by $t > 0$ in multiplicative bounds of the form

$$\frac{1}{1+t} \ell(i, \lambda) \leq \tilde{\ell}(i, \lambda) \leq (1+t) \ell(i, \lambda), \quad \forall i \in [n]. \quad (2)$$

Before proposing a new improved solution, we briefly discuss relevant previous works. To provide a unified view, some preliminary discussion is useful.

2.2 Approximate leverage scores

First, we recall how a subset of columns can be used to compute approximate leverage scores. For $M \leq n$, let $J = \{j_i\}_{i=1}^M$ with $j_i \in [n]$, and $\hat{K}_{J,J} \in \mathbb{R}^{M \times M}$ with entries $(\hat{K}_{J,J})_{lm} = \hat{K}_{j_l, j_m}$. For $i \in [n]$, let $\hat{K}_{J,i} = (\hat{K}_{j_1, i}, \dots, \hat{K}_{j_M, i})$ and consider for $\lambda > 1/n$,

$$\tilde{\ell}_J(i, \lambda) = (\lambda n)^{-1} (\hat{K}_{ii} - \hat{K}_{J,i}^{\top} (\hat{K}_{J,J} + \lambda n A)^{-1} \hat{K}_{J,i}), \quad (3)$$

where $A \in \mathbb{R}^{M \times M}$ is a matrix to be specified * (see later for details). The above definition is motivated by the observation that if $J = [n]$, and $A = I$, then $\tilde{\ell}_J(i, \lambda) = \ell(i, \lambda)$, by the following

*Clearly, $\tilde{\ell}_J$ depends on the choice of the matrix A , but we omit this dependence to simplify the notation.

identity

$$\widehat{K}(\widehat{K} + \lambda n I)^{-1} = (\lambda n)^{-1}(\widehat{K} - \widehat{K}(\widehat{K} + \lambda n I)^{-1}\widehat{K}).$$

In the following, it is also useful to consider a subset of leverage scores computed as in [3]. For $M \leq R \leq n$, let $U = \{u_i\}_{i=1}^R$ with $u_i \in [n]$, and

$$L_J(U, \lambda) = \{\tilde{\ell}_J(u_1, \lambda), \dots, \tilde{\ell}_J(u_R, \lambda)\}. \quad (4)$$

Also in the following we will use the notation

$$L_J(U, \lambda) \mapsto J' \quad (5)$$

to indicate the leverage score sampling of $J' \subset U$ columns based on the leverage scores $L_J(U, \lambda)$, that is the procedure of sampling columns from U according to their leverage scores [1], computed using J , to obtain a new subset of columns J' .

We end noting that leverage score sampling [5] requires $\mathcal{O}(M^2)$ memory to store K_J , and $\mathcal{O}(M^3 + RM^2)$ time to invert K_J , and compute R leverage scores via [3].

2.3 Previous algorithms for leverage scores computations

We discuss relevant previous approaches using the above quantities.

TWO-PASS sampling [6]. This is the first approximate leverage score sampling proposed, and is based on using directly [5] as $L_{J_1}(U_2, \lambda) \mapsto J_2$, with $U_2 = [n]$ and J_1 a subset taken uniformly at random. Here we call this method TWO-PASS sampling since it requires two rounds of sampling on the whole set $[n]$, one uniform to select J_1 and one using leverage scores to select J_2 .

RECURSIVE-RLS [9]. This is a development of TWO-PASS sampling based on the idea of recursing the above construction. In our notation, let $U_1 \subset U_2 \subset U_3 = [n]$, where U_1, U_2 are uniformly sampled and have cardinalities $n/4$ and $n/2$, respectively. The idea is to start from $J_1 = U_1$, and consider first

$$L_{J_1}(U_2, \lambda) \mapsto J_2,$$

but then continue with

$$L_{J_2}(U_3, \lambda) \mapsto J_3.$$

Indeed, the above construction can be made recursive for a family of nested subsets $(U_h)_H$ of cardinalities $n/2^h$, considering $J_1 = U_1$ and

$$L_{J_h}(U_{h+1}, \lambda) \mapsto J_{h+1}. \quad (6)$$

SQUEAK [8]. This approach follows a different iterative strategy. Consider a partition U_1, U_2, U_3 of $[n]$, so that $U_j = n/3$, for $j = 1, \dots, 3$. Then, consider $J_1 = U_1$, and

$$L_{J_1 \cup U_2}(J_1 \cup U_2, \lambda) \mapsto J_2,$$

and then continue with

$$L_{J_2 \cup U_3}(J_2 \cup U_3, \lambda) \mapsto J_3.$$

Similarly to the other cases, the procedure is iterated considering H subsets $(U_h)_{h=1}^H$ each with cardinality n/H . Starting from $J_1 = U_1$ the iterations is

$$L_{J_h \cup U_{h+1}}(J_h \cup U_{h+1}, \lambda). \quad (7)$$

We note that all the above procedures require specifying the number of iteration to be performed, the weights matrix to compute the leverage scores at each iteration, and a strategy to select the subsets $(U_h)_h$. In all the above cases the selection of U_h is based on uniform sampling, while the number of iterations and weight choices arise from theoretical considerations (see [6, 8, 9] for details).

Note that TWO-PASS SAMPLING uses a set J_1 of cardinality roughly $1/\lambda$ (an upper bound on $d_\infty(\lambda)$) and incurs in a computational cost of $RM^2 = n/\lambda^2$. In comparison, RECURSIVE-RLS [9] leads to essentially the same accuracy while improving computations. In particular, the sets J_h are never larger than $d_{\text{eff}}(\lambda)$. Taking into account that at the last iteration performs leverage score sampling on $U_h = [n]$, the total computational complexity is $nd_{\text{eff}}(\lambda)^2$. SQUEAK [8] recovers the same accuracy, size of J_h , and $nd_{\text{eff}}(\lambda)^2$ time complexity when $|U_h| \simeq d_{\text{eff}}(\lambda)$, but only requires a single pass over the data. We also note that a distributed version of SQUEAK is discussed in [8], which allows to reduce the computational cost to $nd_{\text{eff}}(\lambda)^2/p$, provided p machines are available.

Algorithm 1 Bottom-up Leverage Scores Sampling (BLESS)

Input: dataset $\{x_i\}_{i=1}^n$, regularization λ , step q , starting reg. λ_0 , constants q_1, q_2 controlling the approximation level.

Output: $M_h \in [n]$ number of selected points, J_h set of indexes, A_h weights.

```
1:  $J_0 = \emptyset, A_0 = \emptyset, H = \frac{\log(\lambda_0/\lambda)}{\log q}$ 
2: for  $h = 1 \dots H$  do
3:    $\lambda_h = \lambda_{h-1}/q$ 
4:   set constant  $R_h = q_1 \min\{\kappa^2/\lambda_h, n\}$ 
5:   sample  $U_h = \{u_1, \dots, u_{R_h}\}$  i.i.d.  $u_i \sim \text{Uniform}([n])$ 
6:   compute  $\tilde{\ell}_{J_{h-1}}(x_{u_k}, \lambda_h)$  for all  $u_k \in U_h$  using Eq. 3
7:   set  $P_h = (p_{h,k})_{k=1}^{R_h}$  with  $p_{h,k} = \tilde{\ell}_{J_{h-1}}(x_{u_k}, \lambda_h) / (\sum_{u \in U_h} \tilde{\ell}_{J_{h-1}}(x_u, \lambda_h))$ 
8:   set constant  $M_h = q_2 d_h$  with  $d_h = \frac{n}{R_h} \sum_{u \in U_h} \tilde{\ell}_{J_{h-1}}(x_u, \lambda_h)$ , and
9:   sample  $J_h = \{j_1, \dots, j_{M_h}\}$  i.i.d.  $j_i \sim \text{Multinomial}(P_h, U_h)$ 
10:   $A_h = \frac{R_h M_h}{n} \text{diag}(p_{h,j_1}, \dots, p_{h,j_{M_h}})$ 
11: end for
```

2.4 Leverage score sampling with BLESS

The procedure we propose, dubbed BLESS, has similarities to the one proposed in [9] (see (6)), but also some important differences. The main difference is that, rather than a fixed λ , we consider a decreasing sequence of parameters $\lambda_0 > \lambda_1 > \dots > \lambda_H = \lambda$ resulting in different algorithmic choices. For the construction of the subsets U_h we do not use nested subsets, but rather each $(U_h)_{h=1}^H$ is sampled uniformly and independently, with a size smoothly increasing as $1/\lambda_h$. Similarly, as in [9] we proceed iteratively, but at each iteration a different decreasing parameter λ_h is used to compute the leverage scores. Using the notation introduced above, the iteration of BLESS is given by

$$L_{J_h}(U_{h+1}, \lambda_{h+1}) \mapsto J_{h+1}, \quad (8)$$

where the initial set $J_1 = U_1$ is sampled uniformly with size roughly $1/\lambda_0$.

BLESS has two main advantages. The first is computational: each of the sets U_h , including the final U_H , has cardinality smaller than $1/\lambda$. Therefore the overall runtime has a cost of only $RM^2 \leq M^2/\lambda$, which can be dramatically smaller than the nM^2 cost achieved by the methods in [9], [8] and is comparable to the distributed version of SQUEAK using $p = \lambda/n$ machines. The second advantage is that a whole *path* of leverage scores $\{\ell(i, \lambda_h)\}_{h=1}^H$ is computed at once, in the sense that at each iteration accurate approximate leverage scores at scale λ_h are computed. This is extremely useful in practice, as it can be used when cross-validating λ_h . As a comparison, for all previous method a full run of the algorithm is needed for each value of λ_h .

In the paper we consider two variations of the above general idea leading to Algorithm 1 and Algorithm 2. The main difference in the two algorithms lies in the way in which sampling is performed: with and without replacement, respectively. In particular, considering sampling without replacement (see [2]) it is possible to take the set $(U_h)_{h=1}^H$ to be nested and also to obtain slightly improved results, as shown in the next section.

The derivation of BLESS rests on some basic ideas. First, note that, since sampling uniformly a set U_λ of size $d_\infty(\lambda) \leq 1/\lambda$ allows a good approximation, then we can replace $L_{[n]}([n], \lambda) \mapsto J$ by

$$L_{U_\lambda}(U_\lambda, \lambda) \mapsto J, \quad (9)$$

where J can be taken to have cardinality $d_{\text{eff}}(\lambda)$. However, this is still costly, and the idea is to repeat and couple approximations at multiple scales. Consider $\lambda' > \lambda$, a set $U_{\lambda'}$ of size $d_\infty(\lambda') \leq 1/\lambda'$ sampled uniformly, and $L_{U_{\lambda'}}(U_{\lambda'}, \lambda') \mapsto J'$. The basic idea behind BLESS is to replace (9) by

$$L_{J'}(U_\lambda, \lambda) \mapsto \tilde{J}.$$

The key result, see [10], is that taking $\tilde{J}$ of cardinality

$$(\lambda'/\lambda)d_{\text{eff}}(\lambda) \quad (10)$$

suffice to achieve the same accuracy as J . Now, if we take λ' sufficiently large, it is easy to see that $d_{\text{eff}}(\lambda') \sim d_\infty(\lambda') \sim 1/\lambda'$, so that we can take J' uniformly at random. However, the factor (λ'/λ) in (10) becomes too big. Taking multiple scales fix this problem and leads to the iteration in [8].

Algorithm 2 Bottom-up Leverage Scores Sampling without Replacement (BLESS-R)

Input: dataset $\{x_i\}_{i=1}^n$, regularization λ , step q , starting reg. λ_0 , constant q_2 controlling the approximation level.

Output: $M_h \in [n]$ number of selected points, J_h set of indexes, A_h weights.

```

1:  $J_0 = \emptyset$ ,  $A_0 = []$ ,  $H = \frac{\log(\lambda_0/\lambda)}{\log q}$ ,
2: for  $h = 1 \dots H$  do
3:    $\lambda_h = \lambda_{h-1}/q$ 
4:   set constant  $\beta_h = \min\{q_2\kappa^2/(\lambda_h n), 1\}$ 
5:   initialize  $U_h = \emptyset$ 
6:   for  $i \in [n]$  do
7:     add  $i$  to  $U_h$  with probability  $\beta_h$ 
8:   end for
9:   for  $j \in U_h$  do
10:    compute  $p_{h,j} = \min\{q_2\tilde{\ell}_{J_{h-1}}(x_j, \lambda_{h-1}), 1\}$ 
11:    add  $j$  to  $J_h$  with probability  $p_{h,j}/\beta_h$ 
12:   end for
13:    $J_h = \{j_1, \dots, j_{M_h}\}$ , and  $A_h = \text{diag}(p_{h,j_1}, \dots, p_{h,j_{M_h}})$ .
14: end for

```

2.5 Theoretical guarantees

Our first main result establishes in a precise and quantitative way the advantages of BLESS.

Theorem 1. Let $n \in \mathbb{N}$, $\lambda > 0$ and $\delta \in (0, 1]$. Given $t > 0$, $q > 1$ and $H \in \mathbb{N}$, $(\lambda_h)_{h=1}^H$ defined as in Algorithms [1](#) and [2](#) when $(J_h, a_h)_{h=1}^H$ are computed

1. by Alg. [1](#) with parameters $\lambda_0 = \frac{\kappa^2}{\min(t,1)}$, $q_1 \geq \frac{5\kappa^2 q_2}{q(1+t)}$, $q_2 \geq 12q \frac{(2t+1)^2}{t^2} (1+t) \log \frac{12Hn}{\delta}$,
2. by Alg. [2](#) with parameters $\lambda_0 = \frac{\kappa^2}{\min(t,1)}$, $q_1 \geq 54\kappa^2 \frac{(2t+1)^2}{t^2} \log \frac{12Hn}{\delta}$,

let $\tilde{\ell}_{J_h}(i, \lambda_h)$ as in Eq. [\(3\)](#) depending on J_h, A_h , then with probability at least $1 - \delta$:

- (a) $\frac{1}{1+t} \ell(i, \lambda_h) \leq \tilde{\ell}_{J_h}(i, \lambda_h) \leq (1 + \min(t, 1)) \ell(i, \lambda_h), \quad \forall i \in [n], h \in [H],$
- (b) $|J_h| \leq q_2 d_{\text{eff}}(\lambda_h), \quad \forall h \in [H].$

The above result confirms that the subsets J_h computed by BLESS are accurate in the desired sense, see [\(2\)](#), and the size of all J_h is small and proportional to $d_{\text{eff}}(\lambda_h)$, leading to a computational cost of only $\mathcal{O}(\min(\frac{1}{\lambda}, n) d_{\text{eff}}(\lambda)^2 \log^2 \frac{1}{\lambda})$ in time and $\mathcal{O}(d_{\text{eff}}(\lambda)^2 \log^2 \frac{1}{\lambda})$ in space (for additional properties of J_h see Thm. [4](#) in appendixes). Table [1](#) compares the complexity and number of columns sampled by BLESS with other methods. The crucial point is that in most applications, the parameter λ is chosen as a decreasing function of n , e.g. $\lambda = 1/\sqrt{n}$, resulting in potentially massive computational gains. Indeed, since BLESS computes leverage scores for sets of size at most $1/\lambda$, this allows to perform leverage scores sampling on matrices with millions of rows/columns, as shown in the experiments. In the next section, we illustrate the impact of BLESS in the context of supervised statistical learning.

3 Efficient supervised learning with leverage scores

In this section, we discuss the impact of BLESS in a supervised learning. Unlike most previous results on leverage scores sampling in this context [\[6, 8, 9\]](#), we consider the setting of statistical learning, where the challenge is that inputs, as well as the outputs, are random. More precisely, given a probability space $(X \times Y, \rho)$, where $Y \subset \mathbb{R}$, and considering least squares, the problem is to solve

$$\min_{f \in \mathcal{H}} \mathcal{E}(f), \quad \mathcal{E}(f) = \int_{X \times Y} (f(x) - y)^2 d\rho(x, y), \quad (11)$$

Algorithm	Runtime	$ J $
Uniform Sampling [5]	—	$1/\lambda$
Exact RLS Sampl.	n^3	$d_{\text{eff}}(\lambda)$
Two-Pass Sampling [6]	n/λ^2	$d_{\text{eff}}(\lambda)$
Recursive RLS [9]	$nd_{\text{eff}}(\lambda)^2$	$d_{\text{eff}}(\lambda)$
SQUEAK [8]	$nd_{\text{eff}}(\lambda)^2$	$d_{\text{eff}}(\lambda)$
This work, Alg. 1 and 2	$1/\lambda d_{\text{eff}}(\lambda)^2$	$d_{\text{eff}}(\lambda)$

Table 1: The proposed algorithms are compared with the state of the art (in $\mathcal{O}$ notation), in terms of time complexity and cardinality of the set J required to satisfy the approximation condition in Eq. 2.

when ρ is known only through $(x_i, y_i)_{i=1}^n \sim \rho^n$. In the above minimization problem, $\mathcal{H}$ is a reproducing kernel Hilbert space defined by a positive definite kernel $K : X \times X \rightarrow \mathbb{R}$ [12]. Recall that the latter is defined as the completion of $\text{span}\{K(x, \cdot) \mid x \in X\}$ with the inner product $\langle K(x, \cdot), K(x', \cdot) \rangle_{\mathcal{H}} = K(x, x')$. The quality of an empirical approximate solution $\hat{f}$ is measured via probabilistic bounds on the excess risk $\mathcal{R}(\hat{f}) = \mathcal{E}(\hat{f}) - \min_{f \in \mathcal{H}} \mathcal{E}(f)$.

3.1 Learning with FALKON-BLESS

The algorithm we propose, called FALKON-BLESS, combines BLESS with FALKON [14] a state of the art algorithm to solve the least squares problem presented above. The appeal of FALKON is that it is currently the most efficient solution to achieve optimal excess risk bounds. As we discuss in the following, the combination with BLESS leads to further improvements.

We describe the derivation of the considered algorithm starting from kernel ridge regression (KRR)

$$\hat{f}_\lambda(x) = \sum_{i=1}^n K(x, x_i) c_i, \quad c = (\hat{K} + \lambda n I)^{-1} \hat{Y} \quad (12)$$

where $c = (c_1, \dots, c_n)$, $\hat{Y} = (y_1, \dots, y_n)$ and $\hat{K} \in \mathbb{R}^{n \times n}$ is the empirical kernel matrix with entries $(\hat{K})_{ij} = K(x_i, x_j)$. KRR has optimal statistical properties [16], but large $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ space requirements. FALKON can be seen as an approximate ridge regression solver combining a number of algorithmic ideas. First, sampling is used to select a subset $\{\tilde{x}_1, \dots, \tilde{x}_M\}$ of the input data uniformly at random, and to define an approximate solution

$$\hat{f}_{\lambda, M}(x) = \sum_{j=1}^M K(\tilde{x}_j, x) \alpha_j, \quad \alpha = (K_{nM}^\top K_{nM} + \lambda K_{MM})^{-1} K_{nM}^\top y, \quad (13)$$

where $\alpha = (\alpha_1, \dots, \alpha_M)$, $K_{nM} \in \mathbb{R}^{n \times M}$, has entries $(K_{nM})_{ij} = K(x_i, \tilde{x}_j)$ and $K_{MM} \in \mathbb{R}^{M \times M}$ has entries $(K_{MM})_{jj'} = K(\tilde{x}_j, \tilde{x}_{j'})$, with $i \in [n]$, $j, j' \in [M]$. We note, that the linear system in (13) can be seen to be obtained from the one in (12) by uniform column subsampling of the empirical kernel matrix. The columns selected corresponds to the inputs $\{\tilde{x}_1, \dots, \tilde{x}_M\}$. FALKON proposes to compute a solution of the linear system [13] via a preconditioned iterative solver. The preconditioner is the core of the algorithm and is defined by a matrix B such that

$$BB^\top = \left(\frac{n}{M} K_{MM}^2 + \lambda K_{MM} \right)^{-1}. \quad (14)$$

The above choice provides a computationally efficient approximation to the exact preconditioner of the linear system in (13) corresponding to B such that $BB^\top = (K_{nM}^\top K_{nM} + \lambda K_{MM})^{-1}$. The preconditioner in (14) can then be combined with conjugate gradient to solve the linear system in (13). The overall algorithm has complexity $\mathcal{O}(nMt)$ in time and $\mathcal{O}(M^2)$ in space, where t is the number of conjugate gradient iterations performed.

In this paper, we analyze a variation of FALKON where the points $\{\tilde{x}_1, \dots, \tilde{x}_M\}$ are selected via leverage score sampling using BLESS, see Algorithm 1 or Algorithm 2 so that $M = M_h$ and $\tilde{x}_k = x_{j_k}$, for $J_h = \{j_1, \dots, j_{M_h}\}$ and $k \in [M_h]$. Further, the preconditioner in (14) is replaced by

$$B_h B_h^\top = \left(\frac{n}{M} K_{J_h, J_h} A_h^{-1} K_{J_h, J_h} + \lambda_h K_{J_h, J_h} \right)^{-1}. \quad (15)$$

	Time	R-ACC	5 th / 95 th quant
BLESS	17	1.06	0.57 / 2.03
BLESS-R	17	1.06	0.73 / 1.50
SQUEAK	52	1.06	0.70 / 1.48
Uniform	-	1.09	0.22 / 3.75
RRLS	235	1.59	1.00 / 2.70

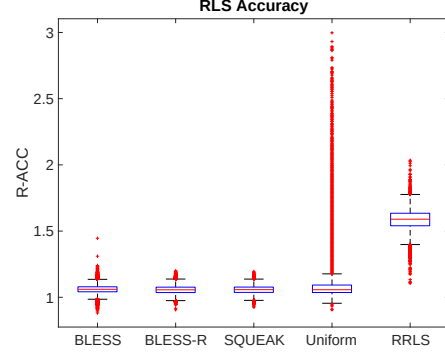


Figure 1: Leverage scores relative accuracy for $\lambda = 10^{-5}$, $n = 70\,000$, $M = 10\,000$, 10 repetitions.

This solution can lead to huge computational improvements. Indeed, the total cost of FALKON-BLESS is the sum of computing BLESS and FALKON, corresponding to

$$O(nMt + (1/\lambda)M^2 \log n + M^3) \quad O(M^2), \quad (16)$$

in time and space respectively, where M is the size of the set J_H returned by BLESS.

3.2 Statistical properties of FALKON-BLESS

In this section, we state and discuss our second main result, providing an excess risk bound for FALKON-BLESS. Here a population version of the effective dimension plays a key role. Let ρ_X be the marginal measure of ρ on X , let $C : \mathcal{H} \rightarrow \mathcal{H}$ be the linear operator defined as follows and $d_{\text{eff}}^*(\lambda)$ be the population version of $d_{\text{eff}}(\lambda)$,

$$d_{\text{eff}}^*(\lambda) = \text{Tr}(C(C + \lambda I)^{-1}), \quad \text{with} \quad (Cf)(x') = \int_X K(x', x)f(x)d\rho_X(x),$$

for any $f \in \mathcal{H}$, $x \in X$. It is possible to show that $d_{\text{eff}}^*(\lambda)$ is the limit of $d_{\text{eff}}(\lambda)$ as n goes to infinity, see Lemma [1](#) below taken from [\[15\]](#). If we assume throughout that,

$$K(x, x') \leq \kappa^2, \quad \forall x, x' \in X, \quad (17)$$

then the operator C is symmetric, positive definite and trace class, and the behavior of $d_{\text{eff}}^*(\lambda)$ can be characterized in terms of the properties of the eigenvalues $(\sigma_j)_{j \in \mathbb{N}}$ of C . Indeed as for $d_{\text{eff}}(\lambda)$, we have that $d_{\text{eff}}^*(\lambda) \leq \kappa^2/\lambda$, moreover if $\sigma_j = \mathcal{O}(j^{-\alpha})$, for $\alpha \geq 1$, we have $d_{\text{eff}}^*(\lambda) = \mathcal{O}(\lambda^{-1/\alpha})$. Then for larger α , d_{eff}^* is smaller than $1/\lambda$ and faster learning rates are possible, as shown below.

We next discuss the properties of the FALKON-BLESS solution denoted by $\hat{f}_{\lambda, n, t}$.

Theorem 2. *Let $n \in \mathbb{N}$, $\lambda > 0$ and $\delta \in (0, 1]$. Assume that $y \in [-\frac{a}{2}, \frac{a}{2}]$, almost surely, $a > 0$, and denote by $f_{\mathcal{H}}$ a minimizer of [\(11\)](#). There exists $n_0 \in \mathbb{N}$, such that for any $n \geq n_0$, if $t \geq \log n$, $\lambda \geq \frac{9\kappa^2}{n} \log \frac{n}{\delta}$, then the following holds with probability at least $1 - \delta$:*

$$\mathcal{R}(\hat{f}_{\lambda, n, t}) \leq \frac{4a}{n} + 32\|f_{\mathcal{H}}\|_{\mathcal{H}}^2 \left(\frac{a^2 \log^2 \frac{2}{\delta}}{n^2 \lambda} + \frac{a d_{\text{eff}}(\lambda) \log \frac{2}{\delta}}{n} + \lambda \right).$$

In particular, when $d_{\text{eff}}^*(\lambda) = \mathcal{O}(\lambda^{-1/\alpha})$, for $\alpha \geq 1$, by selecting $\lambda_* = n^{-\alpha/(\alpha+1)}$, we have

$$\mathcal{R}(\hat{f}_{\lambda_*, n, t}) \leq cn^{-\frac{\alpha}{\alpha+1}},$$

where c is given explicitly in the proof.

We comment on the above result discussing the statistical and computational implications.

Statistics. The above theorem provides statistical guarantees in terms of finite sample bounds on the excess risk of FALKON-BLESS. A first bound depends of the number of examples n , the regularization parameter λ and the population effective dimension $d_{\text{eff}}^*(\lambda)$. The second bound is derived optimizing λ , and is the same as the one achieved by exact kernel ridge regression which is

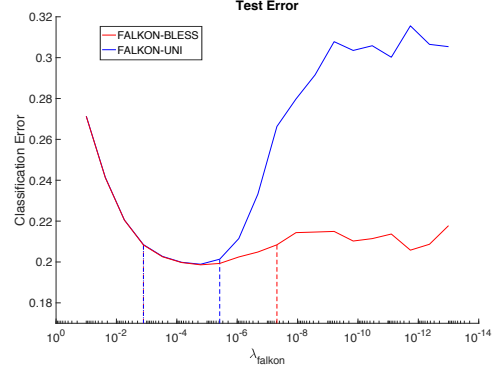
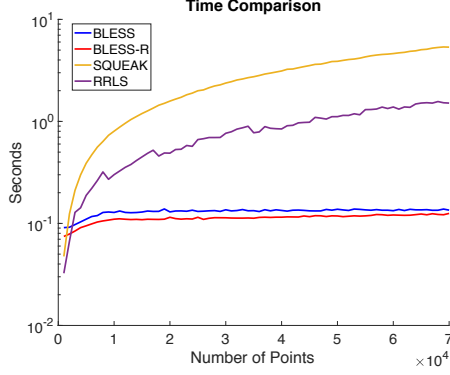


Figure 2: Runtimes with $\lambda = 10^{-3}$ and n increasing Figure 3: C-err at 5 iterations for varying λ_{falkon}

known to be optimal [16, 17, 18]. Note that improvements under further assumptions are possible and are derived in the supplementary materials, see Thm. 8. Here, we comment on the computational properties of FALKON-BLESS and compare it to previous solutions.

Computations. To discuss computational implications, we recall a result from [15] showing that the population version of the effective dimension $d_{\text{eff}}^*(\lambda)$ and the effective dimension $d_{\text{eff}}(\lambda)$ associated to the empirical kernel matrix converge up to constants.

Lemma 1. *Let $\lambda > 0$ and $\delta \in (0, 1]$. When $\lambda \geq \frac{9\kappa^2}{n} \log \frac{n}{\delta}$, then with probability at least $1 - \delta$,*

$$(1/3)d_{\text{eff}}^*(\lambda) \leq d_{\text{eff}}(\lambda) \leq 3d_{\text{eff}}^*(\lambda).$$

Recalling the complexity of FALKON-BLESS [16], using Thm 2 and Lemma 1, we derive a cost

$$\mathcal{O} \left(nd_{\text{eff}}^*(\lambda) \log n + \frac{1}{\lambda} d_{\text{eff}}^*(\lambda)^2 \log n + d_{\text{eff}}^*(\lambda)^3 \right)$$

in time and $\mathcal{O}(d_{\text{eff}}^*(\lambda)^2)$ in space, for all n, λ satisfying the assumptions in Theorem 2. These expressions can be further simplified. Indeed, it is easy to see that for all $\lambda > 0$,

$$d_{\text{eff}}^*(\lambda) \leq \kappa^2 / \lambda, \quad (18)$$

so that $d_{\text{eff}}^*(\lambda)^3 \leq \frac{\kappa^2}{\lambda} d_{\text{eff}}^*(\lambda)^2$. Moreover, if we consider the optimal choice $\lambda_* = \mathcal{O}(n^{-\frac{\alpha}{\alpha+1}})$ given in Theorem 2, and take $d_{\text{eff}}^*(\lambda) = \mathcal{O}(\lambda^{-1/\alpha})$, we have $\frac{1}{\lambda_*} d_{\text{eff}}^*(\lambda_*) \leq \mathcal{O}(n)$, and therefore $\frac{1}{\lambda} d_{\text{eff}}^*(\lambda)^2 \leq \mathcal{O}(nd_{\text{eff}}^*(\lambda))$. In summary, for the parameter choices leading to optimal learning rates, FALKON-BLESS has complexity $\tilde{\mathcal{O}}(nd_{\text{eff}}^*(\lambda_*))$, in time and $\tilde{\mathcal{O}}(d_{\text{eff}}^*(\lambda_*)^2)$ in space, ignoring log terms. We can compare this to previous results. In [14] uniform sampling is considered leading to $M \leq \mathcal{O}(1/\lambda)$ and achieving a complexity of $\tilde{\mathcal{O}}(n/\lambda)$ which is always larger than the one achieved by FALKON in view of (18). Approximate leverage scores sampling is also considered in [14] requiring $\tilde{\mathcal{O}}(nd_{\text{eff}}(\lambda)^2)$ time and reducing the time complexity of FALKON to $\tilde{\mathcal{O}}(nd_{\text{eff}}(\lambda_*))$. Clearly in this case the complexity of leverage scores sampling dominates, and our results provide BLESS as a fix.

4 Experiments

Leverage scores accuracy. We first study the accuracy of the leverage scores generated by BLESS and BLESS-R, comparing SQUEAK [8] and Recursive-RLS (RRLS) [9]. We begin by uniformly sampling a subsets of $n = 7 \times 10^4$ points from the SUSY dataset [19], and computing the exact leverage scores $\ell(i, \lambda)$ using a Gaussian Kernel with $\sigma = 4$ and $\lambda = 10^{-5}$, which is at the limit of our computational feasibility. We then run each algorithm to compute the approximate leverage scores $\tilde{\ell}_{J_H}(i, \lambda)$, and we measure the accuracy of each method using the ratio $\tilde{\ell}_{J_H}(i, \lambda) / \ell(i, \lambda)$ (R-ACC). The final results are presented in Figure 1. On the left side for each algorithm we report runtime, mean R-ACC, and the 5th and 95th quantile, each averaged over the 10 repetitions. On the right side a box-plot of the R-ACC. As shown in Figure 1 BLESS and BLESS-R achieve the same optimal accuracy

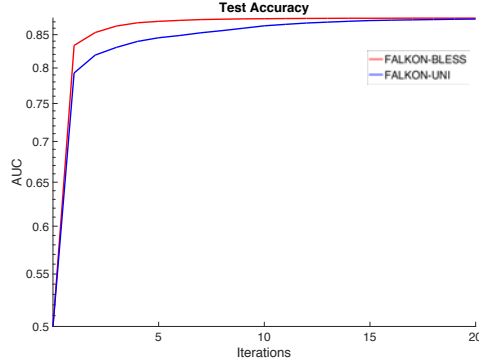


Figure 4: AUC per iteration of the SUSY dataset

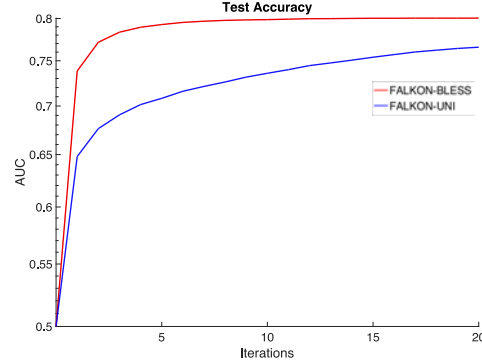


Figure 5: AUC per iteration of the HIGGS dataset

of SQUEAK with just a fraction of time. Note that despite our best efforts, we could not obtain high-accuracy results for RRLS (maybe a wrong constant in the original implementation). However note that RRLS is computationally demanding compared to BLESS, being orders of magnitude slower, as expected from the theory. Finally, although uniform sampling is the fastest approach, it suffers from much larger variance and can over or under-estimate leverage scores by an order of magnitude more than the other methods, making it more fragile for downstream applications.

In Fig. 2 we plot the runtime cost of the compared algorithms as the number of points grows from $n = 1000$ to 70000, this time for $\lambda = 10^{-3}$. We see that while previous algorithms' runtime grows near-linearly with n , BLESS and BLESS-R run in a constant $1/\lambda$ runtime, as predicted by the theory.

BLESS for supervised learning. We study the performance of FALKON-BLESS and compare it with the original FALKON [14] where an equal number of Nyström centres are sampled uniformly at random (FALKON-UNI). We take from [14] the two biggest datasets and their best hyper-parameters for the FALKON algorithm.

We noticed that it is possible to achieve the same accuracy of FALKON-UNI, by using λ_{bless} for BLESS and λ_{falcon} for FALKON with $\lambda_{\text{bless}} \gg \lambda_{\text{falcon}}$, in order to lower the d_{eff} and keep the number of Nyström centres low. For the SUSY dataset we use a Gaussian Kernel with $\sigma = 4$, $\lambda_{\text{falcon}} = 10^{-6}$, $\lambda_{\text{bless}} = 10^{-4}$ obtaining $M_H \simeq 10^4$ Nyström centres. For the HIGGS dataset we use a Gaussian Kernel with $\sigma = 22$, $\lambda_{\text{falcon}} = 10^{-8}$, $\lambda_{\text{bless}} = 10^{-6}$, obtaining $M_H \simeq 3 \times 10^4$ Nyström centres. We then sample a comparable number of centers uniformly for FALKON-UNI. Looking at the plot of their AUC at each iteration (Fig. 4, 5) we observe that FALKON-BLESS converges much faster than FALKON-UNI. For the SUSY dataset (Figure 4) 5 iterations of FALKON-BLESS (160 seconds) achieve the same accuracy of 20 iterations of FALKON-UNI (610 seconds). Since running BLESS takes just 12 secs. this corresponds to a $\sim 4\times$ speedup. For the HIGGS dataset 10 iter. of FALKON-BLESS (with BLESS requiring 1.5 minutes, for a total of 1.4 hours) achieve better accuracy of 20 iter. of FALKON-UNI (2.7 hours). Additionally we observed that FALKON-BLESS is more stable than FALKON-UNI w.r.t. $\lambda_{\text{falcon}}, \sigma$. In Figure 3 the classification error after 5 iterations of FALKON-BLESS and FALKON-UNI over the SUSY dataset ($\lambda_{\text{bless}} = 10^{-4}$). We notice that FALKON-BLESS has a wider optimal region (95% of the best error) for the regularization parameter ($[1.3 \times 10^{-3}, 4.8 \times 10^{-8}]$) w.r.t. FALKON-UNI ($[1.3 \times 10^{-3}, 3.8 \times 10^{-6}]$).

5 Conclusions

In this paper we presented two algorithms BLESS and BLESS-R to efficiently compute a small set of columns from a large symmetric positive semidefinite matrix K , useful for approximating the matrix or to compute leverage scores with a given precision. Moreover we applied the proposed algorithms in the context of statistical learning with least squares, combining BLESS with FALKON [14]. We analyzed the computational and statistical properties of the resulting algorithm, showing that it achieves optimal statistical guarantees with a cost that is $O(nd_{\text{eff}}^*(\lambda))$ in time, being currently the fastest. We can extend the proposed work in several ways: (a) combine BLESS with fast stochastic [20] or online [21] gradient algorithms and other approximation schemes (i.e. random features [22, 23, 24]), to further reduce the computational complexity for optimal rates, (b) consider the impact of BLESS in the context of multi-tasking [25, 26] or structured prediction [27, 28].

Acknowledgments.

This material is based upon work supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216, and the Italian Institute of Technology. We gratefully acknowledge the support of NVIDIA Corporation for the donation of the Titan Xp GPUs and the Tesla k40 GPU used for this research. L. R. acknowledges the support of the AFOSR projects FA9550-17-1-0390 and BAA-AFRL-AFOSR-2016-0007 (European Office of Aerospace Research and Development), and the EU H2020-MSCA-RISE project NoMADS - DLV-777826. A. R. acknowledges the support of the European Research Council (grant SEQUOIA 724063).

References

- [1] Raman Arora, Andrew Cotter, Karen Livescu, and Nathan Srebro. Stochastic optimization for pca and pls. In *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*, pages 861–868. IEEE, 2012.
- [2] David P. Woodruff. Sketching as a tool for numerical linear algebra. *arXiv preprint arXiv:1411.4357*, 2014.
- [3] Joel A Tropp. User-friendly tools for random matrices: An introduction. Technical report, CALIFORNIA INST OF TECH PASADENA DIV OF ENGINEERING AND APPLIED SCIENCE, 2012.
- [4] Christopher Williams and Matthias Seeger. Using the Nystrom method to speed up kernel machines. In *Neural Information Processing Systems*, 2001.
- [5] Francis Bach. Sharp analysis of low-rank kernel matrix approximations. In *Conference on Learning Theory*, 2013.
- [6] Ahmed El Alaoui and Michael W. Mahoney. Fast randomized kernel methods with statistical guarantees. In *Neural Information Processing Systems*, 2015.
- [7] Petros Drineas, Malik Magdon-Ismail, Michael W. Mahoney, and David P. Woodruff. Fast approximation of matrix coherence and statistical leverage. *The Journal of Machine Learning Research*, 13(1):3475–3506, 2012.
- [8] Daniele Calandriello, Alessandro Lazaric, and Michal Valko. Distributed adaptive sampling for kernel matrix approximation. In *AISTATS*, 2017.
- [9] Cameron Musco and Christopher Musco. Recursive Sampling for the Nyström Method. In *NIPS*, 2017.
- [10] Daniele Calandriello, Alessandro Lazaric, and Michal Valko. Second-order kernel online convex optimization with adaptive sketching. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 645–653, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.
- [11] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. Adaptive computation and machine learning. MIT Press, Cambridge, Mass, 2006. OCLC: ocm61285753.
- [12] Bernhard Schölkopf, Alexander J Smola, et al. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.
- [13] Alex J Smola and Bernhard Schölkopf. Sparse greedy matrix approximation for machine learning. 2000.
- [14] Alessandro Rudi, Luigi Carratino, and Lorenzo Rosasco. Falcon: An optimal large scale kernel method. In *Advances in Neural Information Processing Systems*, pages 3891–3901, 2017.
- [15] Alessandro Rudi, Raffaello Camoriano, and Lorenzo Rosasco. Less is more: Nyström computational regularization. In *Advances in Neural Information Processing Systems*, pages 1657–1665, 2015.

- [16] Andrea Caponnetto and Ernesto De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007.
- [17] Ingo Steinwart, Don R Hush, Clint Scovel, et al. Optimal rates for regularized least squares regression. In *COLT*, 2009.
- [18] Junhong Lin, Alessandro Rudi, Lorenzo Rosasco, and Volkan Cevher. Optimal rates for spectral algorithms with least-squares regression over hilbert spaces. *Applied and Computational Harmonic Analysis*, 2018.
- [19] Pierre Baldi, Peter Sadowski, and Daniel Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5:4308, 2014.
- [20] Nicolas L Roux, Mark Schmidt, and Francis R Bach. A stochastic gradient method with an exponential convergence _rate for finite training sets. In *Advances in neural information processing systems*, pages 2663–2671, 2012.
- [21] Daniele Calandriello, Alessandro Lazaric, and Michal Valko. Efficient second-order online kernel learning with adaptive embedding. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6140–6150. Curran Associates, Inc., 2017.
- [22] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.
- [23] Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. In *Advances in Neural Information Processing Systems*, pages 3215–3225, 2017.
- [24] Luigi Carratino, Alessandro Rudi, and Lorenzo Rosasco. Learning with sgd and random features. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 10213–10224. Curran Associates, Inc., 2018.
- [25] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.
- [26] Carlo Ciliberto, Alessandro Rudi, Lorenzo Rosasco, and Massimiliano Pontil. Consistent multi-task learning with nonlinear output relations. In *Advances in Neural Information Processing Systems*, pages 1986–1996, 2017.
- [27] Carlo Ciliberto, Lorenzo Rosasco, and Alessandro Rudi. A consistent regularization approach for structured prediction. *Advances in Neural Information Processing Systems 29*, pages 4412–4420, 2016.
- [28] Anna Korba, Alexandre Garcia, and Florence d’Alché Buc. A structured prediction approach for label ranking. In *Advances in Neural Information Processing Systems*, pages 9008–9018, 2018.
- [29] Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404, 1950.
- [30] Ingo Steinwart and Andreas Christmann. *Support vector machines*. Springer Science & Business Media, 2008.

Statistical and Computational Trade-Offs in Kernel K-Means

Daniele Calandriello
LCSL – IIT & MIT,
Genoa, Italy

Lorenzo Rosasco
University of Genoa,
LCSL – IIT & MIT

Abstract

We investigate the efficiency of k-means in terms of both statistical and computational requirements. More precisely, we study a Nyström approach to kernel k-means. We analyze the statistical properties of the proposed method and show that it achieves the same accuracy of exact kernel k-means with only a fraction of computations. Indeed, we prove under basic assumptions that sampling $\sqrt{n}$ Nyström landmarks allows to greatly reduce computational costs without incurring in any loss of accuracy. To the best of our knowledge this is the first result of this kind for unsupervised learning.

1 Introduction

Modern applications require machine learning algorithms to be accurate as well as computationally efficient, since data-sets are increasing in size and dimensions. Understanding the interplay and trade-offs between statistical and computational requirements is then crucial [31, 30]. In this paper, we consider this question in the context of clustering, considering a popular nonparametric approach, namely kernel k-means [33]. K-means is arguably one of most common approaches to clustering and produces clusters with piece-wise linear boundaries. Its kernel version allows to consider nonlinear boundaries, greatly improving the flexibility of the approach. Its statistical properties have been studied [15, 24, 10] and from a computational point of view it requires manipulating an empirical kernel matrix. As for other kernel methods, this latter operation becomes unfeasible for large scale problems and deriving approximate computations is subject of a large body of recent works, see for example [34, 16, 29, 35, 25] and reference therein.

In this paper we are interested into quantifying the statistical effect of computational approximations. Arguably one could expect the latter to induce some loss of accuracy. In fact, we prove that, perhaps surprisingly, there are favorable regimes where it is possible maintain optimal statistical accuracy while significantly reducing computational costs. While a similar phenomenon has been recently shown in supervised learning [31, 30, 12], we are not aware of similar results for other learning tasks.

Our approach is based on considering a Nyström approach to kernel k-means based on sampling a subset of training set points (landmarks) that can be used to approximate the kernel matrix [3, 34, 13, 14, 35, 25]. While there is a vast literature on the properties of Nyström methods for kernel approximations [25, 3], experience from supervised learning show that better results can be derived focusing on the task of interest, see discussion in [7]. The properties of Nyström approximations for k-means has been recently studied in [35, 25]. Here they focus only on the computational aspect of the problem, and provide fast methods that achieve an *empirical* cost only a multiplicative factor larger than the optimal one.

Our analysis is aimed at combining both statistical *and* computational results. Towards this end we derive a novel *additive* bound on the empirical cost that can be used to bound the true object of interest: the *expected cost*. This result can be combined with probabilistic results to show that optimal statistical accuracy can be obtained considering only $O(\sqrt{n})$ Nyström landmark points,

where n is the number of training set of points. Moreover, we show similar bounds not only for the optimal solution, which is hard to compute in general, but also for approximate solutions that can be computed efficiently using k -means++. From a computational point of view this leads to massive improvements reducing the memory complexity from $O(n^2)$ to $O(n\sqrt{n})$. Experimental results on large scale data-sets confirm and illustrate our findings.

The rest of the paper is organized as follows. We first overview kernel k -means, and introduce our approximate kernel k -means approach based on Nyström embeddings. We then prove our statistical and computational guarantees and empirically validate them. Finally, we present some limits of our analysis, and open questions.

2 Background

Notation Given an input space $\mathcal{X}$, a sampling distribution μ , and n samples $\{\mathbf{x}_i\}_{i=1}^n$ drawn i.i.d. from μ , we denote with $\mu_n(A) = (1/n) \sum_{i=1}^n \mathbb{I}\{\mathbf{x}_i \in A\}$ the *empirical* distribution. Once the data has been sampled, we use the *feature map* $\varphi(\cdot) : \mathcal{X} \rightarrow \mathcal{H}$ to maps $\mathcal{X}$ into a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ [32], that we assume separable, such that for any $\mathbf{x} \in \mathcal{X}$ we have $\phi = \varphi(\mathbf{x})$. Intuitively, in the rest of the paper the reader can assume that $\phi \in \mathbb{R}^D$ with $D \gg n$ or even infinite. Using the kernel trick [2] we also know that $\phi^\top \phi' = \mathcal{K}(\mathbf{x}, \mathbf{x}')$, where $\mathcal{K}$ is the kernel function associated with $\mathcal{H}$ and $\phi^\top \phi' = \langle \phi, \phi' \rangle_{\mathcal{H}}$ is a short-hand for the inner product in $\mathcal{H}$. With a slight abuse of notation we will also define the norm $\|\phi\|^2 = \phi^\top \phi$, and assume that $\|\varphi(\mathbf{x})\|^2 \leq \kappa^2$ for any $\mathbf{x} \in \mathcal{X}$. Using $\phi_i = \varphi(\mathbf{x}_i)$, we denote with $\mathcal{D} = \{\phi_i\}_{i=1}^n$ the input dataset. We also represent the dataset as the map $\Phi_n = [\phi_1, \dots, \phi_n] : \mathbb{R}^n \rightarrow \mathcal{H}$ with ϕ_i as its i -th column. We denote with $\mathbf{K}_n = \Phi_n^\top \Phi_n$ the empirical kernel matrix with entries $[\mathbf{K}_n]_{i,j} = k_{i,j}$. Finally, given Φ_n we denote as $\Pi_n = \Phi_n \Phi_n^\top (\Phi_n \Phi_n^\top)^+$ the orthogonal projection matrix on the span $\mathcal{H}_n = \text{Im}(\Phi_n)$ of the dataset.

k -mean's objective Given our dataset, we are interested in partitioning it into k disjoint *clusters* each characterized by its *centroid* $\mathbf{c}_j$. The Voronoi cell associated with a centroid $\mathbf{c}_j$ is defined as the set $\mathcal{C}_j := \{i : j = \arg \min_{s \in [k]} \|\phi_i - \mathbf{c}_s\|^2\}$, or in other words a point $\phi_i \in \mathcal{D}$ belongs to the j -th cluster if $\mathbf{c}_j$ is its closest centroid. Let $\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_k]$ be a collection of k centroids from $\mathcal{H}$. We can now formalize the criterion we use to measure clustering quality.

Definition 1. *The empirical and expected squared norm criterion are defined as*

$$W(\mathbf{C}, \mu_n) := \frac{1}{n} \sum_{i=1}^n \min_{j=1, \dots, k} \|\phi_i - \mathbf{c}_j\|^2, \quad W(\mathbf{C}, \mu) := \mathbb{E}_{\phi \sim \mu} \left[\min_{j=1, \dots, k} \|\phi - \mathbf{c}_j\|^2 \right].$$

The empirical risk minimizer (ERM) is defined as $\mathbf{C}_n := \arg \min_{\mathbf{C} \in \mathcal{H}^k} W(\mathbf{C}, \mu_n)$.

The sub-script n in $\mathbf{C}_n$ indicates that it minimizes $W(\mathbf{C}, \mu_n)$ for the n samples in $\mathcal{D}$. Biau et al. [10] gives us a bound on the excess risk of the empirical risk minimizer.

Proposition 1 ([10]). *The excess risk $\mathcal{E}(\mathbf{C}_n)$ of the empirical risk minimizer $\mathbf{C}_n$ satisfies*

$$\mathcal{E}(\mathbf{C}_n) := \mathbb{E}_{\mathcal{D} \sim \mu} [W(\mathbf{C}_n, \mu)] - W^*(\mu) \leq \mathcal{O}(k/\sqrt{n})$$

where $W^*(\mu) := \inf_{\mathbf{C} \in \mathcal{H}^k} W(\mathbf{C}, \mu)$ is the optimal clustering risk.

From a theoretical perspective, this result is only $\sqrt{k}$ times larger than a corresponding $\mathcal{O}(\sqrt{k/n})$ lower bound [18], and therefore shows that the ERM $\mathbf{C}_n$ achieve an excess risk optimal in n . From a computational perspective, Definition 1 cannot be directly used to compute $\mathbf{C}_n$, since the points ϕ_i in $\mathcal{H}$ cannot be directly represented. Nonetheless, due to properties of the squared norm criterion, each $\mathbf{c}_j \in \mathbf{C}_n$ must be the mean of all ϕ_i associated with that center, i.e., $\mathbf{C}_n$ belongs to $\mathcal{H}_n$. Therefore, it can be explicitly represented as a sum of the ϕ_i points included in the j -th cluster, i.e., all the points in the j -th Voronoi cell $\mathcal{C}_j$. Let $\mathcal{V}$ be the space of all possible disjoint partitions $[\mathcal{C}_1, \dots, \mathcal{C}_k]$. We can use this fact, together with the kernel trick, to reformulate the objective $W(\cdot, \mu_n)$.

Proposition 2 ([17]). *We can rewrite the objective*

$$\min_{\mathbf{C} \in \mathcal{H}} W(\mathbf{C}, \mu_n) = \frac{1}{n} \min_{\mathcal{V}} \sum_{j=1}^k \sum_{i \in \mathcal{C}_j} \left\| \phi_i - \frac{1}{|\mathcal{C}_j|} \sum_{s \in \mathcal{C}_j} \phi_s \right\|^2$$

$$\text{with } \left\| \phi_i - \frac{1}{|\mathcal{C}_j|} \sum_{s \in \mathcal{C}_j} \phi_s \right\|^2 = k_{i,i} - \frac{2}{|\mathcal{C}_j|} \sum_{s \in \mathcal{C}_j} k_{i,s} + \frac{1}{|\mathcal{C}_j|^2} \sum_{s \in \mathcal{C}_j} \sum_{s' \in \mathcal{C}_j} k_{s,s'}$$

While the combinatorial search over $\mathcal{V}$ can now be explicitly computed and optimized using the kernel matrix $\mathbf{K}_n$, it still remains highly inefficient to do so. In particular, simply constructing and storing $\mathbf{K}_n$ takes $\mathcal{O}(n^2)$ time and space and does not scale to large datasets.

3 Algorithm

A simple approach to reduce computational cost is to use approximate embeddings, which replace the map $\varphi(\cdot)$ and points $\phi_i = \varphi(\mathbf{x}_i) \in \mathcal{H}$ with a finite-dimensional approximation $\tilde{\phi}_i = \tilde{\varphi}(\mathbf{x}_i) \in \mathbb{R}^m$.

Nyström kernel k -means Given a dataset $\mathcal{D}$, we denote with $\mathcal{I} = \{\phi_j\}_{j=1}^m$ a *dictionary* (i.e., subset) of m points ϕ_j from $\mathcal{D}$, and with $\Phi_m : \mathbb{R}^m \rightarrow \mathcal{H}$ the map with these points as columns. These points acts as landmarks [36], inducing a smaller space $\mathcal{H}_m = \text{Im}(\Phi_m)$ spanned by the dictionary. As we will see in the next section, $\mathcal{I}$ should be chosen so that $\mathcal{H}_m$ is close to the whole span $\mathcal{H}_n = \text{Im}(\Phi_n)$ of the dataset.

Let $\mathbf{K}_{m,m} \in \mathbb{R}^{m \times m}$ be the empirical kernel matrix between all points in $\mathcal{I}$, and denote with

$$\mathbf{\Pi}_m = \Phi_m \Phi_m^\top (\Phi_m \Phi_m^\top)^+ = \Phi_m \mathbf{K}_{m,m}^+ \Phi_m^\top, \quad (1)$$

the orthogonal projection on $\mathcal{H}_m$. Then we can define an approximate ERM over $\mathcal{H}_m$ as

$$\mathbf{C}_{n,m} = \arg \min_{\mathbf{C} \in \mathcal{H}_m^k} \frac{1}{n} \sum_{i=1}^n \min_{j=[k]} \|\phi_i - \mathbf{c}_j\|^2 = \arg \min_{\mathbf{C} \in \mathcal{H}_m^k} \frac{1}{n} \sum_{i=1}^n \min_{j=[k]} \|\mathbf{\Pi}_m(\phi_i - \mathbf{c}_j)\|^2, \quad (2)$$

since any component outside of $\mathcal{H}_m$ is just a constant in the minimization. Note that the centroids $\mathbf{C}_{n,m}$ are still points in $\mathcal{H}_m \subset \mathcal{H}$, and we cannot directly compute them. Instead, we can use the eigen-decomposition of $\mathbf{K}_{m,m} = \mathbf{U} \mathbf{\Lambda} \mathbf{U}^\top$ to rewrite $\mathbf{\Pi}_m = \Phi_m \mathbf{U} \mathbf{\Lambda}^{-1/2} \mathbf{\Lambda}^{-1/2} \mathbf{U}^\top \Phi_m^\top$. Defining now $\tilde{\varphi}(\cdot) = \mathbf{\Lambda}^{-1/2} \mathbf{U}^\top \Phi_m^\top \varphi(\cdot)$ we have a finite-rank embedding into $\mathbb{R}^m$. Substituting in Eq. (2)

$$\|\mathbf{\Pi}_m(\phi_i - \mathbf{c}_j)\|^2 = \|\mathbf{\Lambda}^{-1/2} \mathbf{U}^\top \Phi_m^\top (\phi_i - \mathbf{c}_j)\|^2 = \|\tilde{\phi}_i - \mathbf{\Lambda}^{-1/2} \mathbf{U}^\top \Phi_m^\top \mathbf{c}_j\|^2,$$

where $\tilde{\phi}_i := \mathbf{\Lambda}^{-1/2} \mathbf{U}^\top \Phi_m^\top \phi_i$ are the embedded points. Replacing $\tilde{\mathbf{c}}_j := \mathbf{\Lambda}^{-1/2} \mathbf{U}^\top \Phi_m^\top \mathbf{c}_j$ and searching over $\tilde{\mathbf{C}} \in \mathbb{R}^{m \times k}$ instead of searching over $\mathbf{C} \in \mathcal{H}_m^k$, we obtain (similarly to Proposition 2)

$$\tilde{\mathbf{C}}_{n,m} = \arg \min_{\tilde{\mathbf{C}} \in \mathbb{R}^{m \times k}} \frac{1}{n} \sum_{i=1}^n \min_{j=[k]} \|\tilde{\phi}_i - \tilde{\mathbf{c}}_j\|^2 = \frac{1}{n} \min_{\mathcal{V}} \sum_{j=1}^k \sum_{i \in \mathcal{C}_j} \left\| \tilde{\phi}_i - \frac{1}{|\mathcal{C}_j|} \sum_{s \in \mathcal{C}_j} \tilde{\phi}_s \right\|^2, \quad (3)$$

where we do not need to resort to kernel tricks, but can use the m -dimensional embeddings $\tilde{\phi}_i$ to explicitly compute the centroid $\sum_{s \in \mathcal{C}_j} \tilde{\phi}_s$. Eq. (3) can now be solved in multiple ways. The most straightforward is to run a parametric k -means algorithm to compute $\tilde{\mathbf{C}}_{n,m}$, and then invert the relationship $\tilde{\mathbf{c}}_j = \Phi_m \mathbf{U} \mathbf{\Lambda}^{-1/2} \mathbf{c}_j$ to bring back the solution to $\mathcal{H}_m$, i.e., $\mathbf{C}_{n,m} = \Phi_m \mathbf{U}^\top \mathbf{\Lambda}^{1/2} \tilde{\mathbf{C}}_{n,m} = \Phi_m \mathbf{U} \mathbf{\Lambda}^{-1/2} \tilde{\mathbf{C}}_{n,m}$. This can be done in $\mathcal{O}(nm)$ space and $\mathcal{O}(nmkt + nm^2)$ time using t steps of Lloyd's algorithm [22] for k clusters. More in detail, computing the embeddings $\tilde{\phi}_i$ is a one-off cost taking nm^2 time. Once the m -rank Nyström embeddings $\tilde{\phi}_i$ are computed they can be stored and manipulated in nm time and space, with an n/m improvement over the n^2 time and space required to construct $\mathbf{K}_n$.

3.1 Uniform sampling for dictionary construction

Due to its derivation, the computational cost of Algorithm 1 depends on the size m of the dictionary $\mathcal{I}$. Therefore, for computational reasons we would prefer to select as small a dictionary as possible. As a conflicting goal, we also wish to optimize $W(\cdot, \mu_n)$ well, which requires a $\tilde{\varphi}(\cdot)$ and $\mathcal{I}$ rich enough to approximate $W(\cdot, \mu_n)$ well. Let $\mathbf{\Pi}_m^\perp$ be the projection orthogonal to $\mathcal{H}_m$. Then when $\mathbf{c}_i \in \mathcal{H}_m$

$$\|\phi_i - \mathbf{c}_i\|^2 = \|(\mathbf{\Pi}_m + \mathbf{\Pi}_m^\perp)(\phi_i - \mathbf{c}_i)\|^2 = \|\mathbf{\Pi}_m(\phi_i - \mathbf{c}_i)\|^2 + \|\mathbf{\Pi}_m^\perp \phi_i\|^2.$$

We will now introduce the concept of a γ -preserving dictionary $\mathcal{I}$ to control the quantity $\|\mathbf{\Pi}_m^\perp \phi_i\|^2$.

Algorithm 1 Nyström Kernel K-Means

Input: dataset $\mathcal{D} = \{\phi_i\}_{i=1}^n$, dictionary $\mathcal{I} = \{\phi_j\}_{j=1}^m$ with points from $\mathcal{D}$, number of clusters k
compute kernel matrix $\mathbf{K}_{m,m} = \Phi_m^\top \Phi_m$ between all points in $\mathcal{I}$
compute eigenvectors $\mathbf{U}$ and eigenvalues Λ of $\mathbf{K}_{m,m}$
for each point ϕ_i , compute embedding $\tilde{\phi}_i = \Lambda^{-1/2} \mathbf{U}^\top \Phi_m^\top \phi_i = \Lambda^{-1/2} \mathbf{U}^\top \mathbf{K}_{m,i} \in \mathbb{R}^m$
compute optimal centroids $\tilde{\mathbf{C}}_{n,m} \in \mathbb{R}^{m \times k}$ on the embedded dataset $\tilde{\mathcal{D}} = \{\tilde{\phi}_i\}_{i=1}^n$
compute explicit representation of centroids $\mathbf{C}_{n,m} = \Phi_m \mathbf{U} \Lambda^{-1/2} \tilde{\mathbf{C}}_{n,m}$

Definition 2. We define the subspace $\mathcal{H}_m$ and dictionary $\mathcal{I}$ as γ -preserving w.r.t. space $\mathcal{H}_n$ if

$$\Pi_m^\perp = \Pi_n - \Pi_m \preceq \frac{\gamma}{1-\varepsilon} (\Phi_n \Phi_n^\top + \gamma \Pi_n)^{-1}. \quad (4)$$

Notice that the inverse $(\Phi_n \Phi_n^\top + \gamma \Pi_n)^{-1}$ on the right-hand side of the inequality is crucial to control the error $\|\Pi_m^\perp \phi_i\|^2 \lesssim \gamma \phi_i^\top (\Phi_n \Phi_n^\top + \gamma \Pi_n)^{-1} \phi_i$. In particular, since $\phi_i \in \Phi_n$, we have that in the *worst case* the error is bounded as $\phi_i^\top (\Phi_n \Phi_n^\top + \gamma \Pi_n)^{-1} \phi_i \leq \phi_i^\top (\phi_i \phi_i^\top)^\dagger \phi_i \leq 1$. Conversely, since $\lambda_{\max}(\Phi_n \Phi_n^\top) \leq \kappa^2 n$ we know that in the best case the error can be reduced up to $1/n \leq \phi_i^\top \phi_i / \lambda_{\max}(\Phi_n \Phi_n^\top) \leq \phi_i^\top (\Phi_n \Phi_n^\top + \gamma \Pi_n)^{-1} \phi_i$. Note that the directions associated with the larger eigenvalues are the ones that occur most frequently in the data. As a consequence, Definition 2 guarantees that the overall error across the whole dataset remains small. In particular, we can control the residual $\Pi_m^\perp \Phi_n$ after the projection as $\|\Pi_m^\perp \Phi_n\|^2 \leq \gamma \|\Phi_n^\top (\Phi_n \Phi_n^\top + \gamma \Pi_n)^{-1} \Phi_n\| \leq \gamma$.

To construct γ -preserving dictionaries we focus on a uniform random sampling approach[7]. Uniform sampling is historically the first [36], and usually the simplest approach used to construct $\mathcal{I}$. Leveraging results from the literature [7, 14, 25] we can show that uniformly sampling $\tilde{\mathcal{O}}(n/\gamma)$ landmarks generates a γ -preserving dictionary with high probability.

Lemma 1. For a given γ , construct $\mathcal{I}$ by uniformly sampling $m \geq 12\kappa^2 n / \gamma \log(n/\delta) / \varepsilon^2$ landmarks from $\mathcal{D}$. Then w.p. at least $1 - \delta$ the dictionary $\mathcal{I}$ is γ -preserving.

Musco and Musco [25] obtains a similar result, but instead of considering the operator Π_n they focus on the finite-dimensional eigenvectors of $\mathbf{K}_n$. Moreover, their $\Pi_n \preceq \Pi_m + \frac{\varepsilon\gamma}{1-\varepsilon} (\Phi_n \Phi_n^\top)^\dagger$ bound is weaker and would not be sufficient to satisfy our definition of γ -accuracy. A result equivalent to Lemma 1 was obtained by Alaoui and Mahoney [3], but they also only focus on the finite-dimensional eigenvectors of $\mathbf{K}_n$, and did not investigate the implications for $\mathcal{H}$.

Proof sketch of Lemma 1. It is well known [7, 14] that uniformly sampling $\mathcal{O}(n/\gamma\varepsilon^{-2} \log(n/\delta))$ points with replacement is sufficient to obtain w.p. $1 - \delta$ the following guarantees on Φ_m

$$(1 - \varepsilon) \Phi_n \Phi_n^\top - \varepsilon \gamma \Pi_n \preceq \frac{n}{m} \Phi_m \Phi_m^\top \preceq (1 + \varepsilon) \Phi_n \Phi_n^\top + \varepsilon \gamma \Pi_n.$$

Which implies

$$\left(\frac{n}{m} \Phi_m \Phi_m^\top + \gamma \Pi_n \right)^{-1} \preceq ((1 - \varepsilon) \Phi_n \Phi_n^\top - \varepsilon \gamma \Pi_n + \gamma \Pi_n)^{-1} = \frac{1}{1 - \varepsilon} (\Phi_n \Phi_n^\top + \gamma \Pi_n)^{-1}$$

We can now rewrite Π_n as

$$\begin{aligned} \Pi_n &= \left(\frac{n}{m} \Phi_m \Phi_m^\top + \gamma \Pi_n \right) \left(\frac{n}{m} \Phi_m \Phi_m^\top + \gamma \Pi_n \right)^{-1} \\ &= \frac{n}{m} \Phi_m \Phi_m^\top \left(\frac{n}{m} \Phi_m \Phi_m^\top + \gamma \Pi_n \right)^{-1} + \gamma \left(\frac{n}{m} \Phi_m \Phi_m^\top + \gamma \Pi_n \right)^{-1} \\ &\preceq \frac{n}{m} \Phi_m \Phi_m^\top \left(\frac{n}{m} \Phi_m \Phi_m^\top \right)^\dagger + \gamma \left(\frac{n}{m} \Phi_m \Phi_m^\top + \gamma \Pi_n \right)^{-1} \\ &= \Pi_m + \gamma \left(\frac{n}{m} \Phi_m \Phi_m^\top + \gamma \Pi_n \right)^{-1} \preceq \Pi_m + \frac{\gamma}{1 - \varepsilon} (\Phi_n \Phi_n^\top + \gamma \Pi_n)^{-1} \quad \square \end{aligned}$$

In other words, using uniform sampling we can reduce the size of the search space $\mathcal{H}_m$ by a $1/\gamma$ factor (from n to $m \simeq n/\gamma$) in exchange for a γ additive error, resulting in a computation/approximation trade-off that is linear in γ .

4 Theoretical analysis

Exploiting the error bound for γ -preserving dictionaries we are now ready for the main result of this paper: showing that we can improve the computational aspect of kernel k -means using Nyström embedding, while maintaining optimal generalization guarantees.

Theorem 1. *Given a γ -preserving dictionary*

$$\mathcal{E}(\mathbf{C}_{n,m}) = W(\mathbf{C}_{n,m}, \mu) - W(\mathbf{C}_n, \mu) \leq \mathcal{O}\left(k\left(\frac{1}{\sqrt{n}} + \frac{\gamma}{n}\right)\right)$$

From a statistical point of view, Theorem 1 shows that if $\mathcal{I}$ is γ -preserving, the ERM in $\mathcal{H}_m$ achieves the same *excess risk* as the exact ERM from $\mathcal{H}_n$ up to an additional γ/n error. Therefore, choosing $\gamma = \sqrt{n}$ the solution $\mathbf{C}_{n,m}$ achieves the $\mathcal{O}(k/\sqrt{n}) + \mathcal{O}(k\sqrt{n}/n) \leq \mathcal{O}(k/\sqrt{n})$ generalization [10].

From a computational point of view, Lemma 1 shows that we can construct an $\sqrt{n}$ -preserving dictionary simply by sampling $\tilde{\mathcal{O}}(\sqrt{n})$ points uniformly¹, which greatly reduces the embedding size from n to $\sqrt{n}$, and the total required space from n^2 to $\tilde{\mathcal{O}}(n\sqrt{n})$.

Time-wise, the bottleneck becomes the construction of the embeddings $\tilde{\phi}_i$, which takes $nm^2 \leq \tilde{\mathcal{O}}(n^2)$ time, while each iterations of Lloyd's algorithm only requires $nm \leq \tilde{\mathcal{O}}(n\sqrt{n})$ time. In the full generality of our setting this is practically optimal, since computing a $\sqrt{n}$ -preserving dictionary is in general as hard as matrix multiplication [26, 9], which requires $\Omega(n^2)$ time. In other words, unlike the case of space complexity, there is no free lunch for time complexity, that in the worst case must scale as n^2 similarly to the exact case. Nonetheless embedding the points is an embarrassingly parallel problem that can be easily distributed, while in practice it is usually the execution of the Lloyd's algorithm that dominates the runtime.

Finally, when the dataset satisfies certain regularity conditions, the size of $\mathcal{I}$ can be improved, which reduces both embedding and clustering runtime. Denote with $d_{\text{eff}}^n(\gamma) = \text{Tr}(\mathbf{K}_n^\top(\mathbf{K}_n + \mathbf{I}_n)^{-1})$ the so-called *effective dimension* [3] of $\mathbf{K}_n$. Since $\text{Tr}(\mathbf{K}_n^\top(\mathbf{K}_n + \mathbf{I}_n)^{-1}) \leq \text{Tr}(\mathbf{K}_n^\top(\mathbf{K}_n)^+)$, we have that $d_{\text{eff}}^n(\gamma) \leq r := \text{Rank}(\mathbf{K}_n)$, and therefore $d_{\text{eff}}^n(\gamma)$ can be seen as a soft version of the rank. When $d_{\text{eff}}^n(\gamma) \ll \sqrt{n}$ it is possible to construct a γ -preserving dictionary with only $d_{\text{eff}}^n(\gamma)$ landmarks in $\tilde{\mathcal{O}}(nd_{\text{eff}}^n(\gamma)^2)$ time using specialized algorithms [14] (see Section 6). In this case, the embedding step would require only $\tilde{\mathcal{O}}(nd_{\text{eff}}^n(\gamma)^2) \ll \tilde{\mathcal{O}}(n^2)$, improving both time and space complexity.

Moreover, to the best of our knowledge, this is the first example of an unsupervised non-parametric problem where it is always (i.e., without assumptions on μ) possible to preserve the optimal $\mathcal{O}(1/\sqrt{n})$ risk rate while reducing the search from the whole space $\mathcal{H}$ to a smaller $\mathcal{H}_m$ subspace.

Proof sketch of Theorem 1. We can separate the distance between $W(\mathbf{C}_{n,m}, \mu) - W(\mathbf{C}_n, \mu)$ in a component that depends on how close μ is to μ_n , bounded using Proposition 1, and a component $W(\mathbf{C}_{n,m}, \mu_n) - W(\mathbf{C}_n, \mu_n)$ that depends on the distance between $\mathcal{H}_n$ and $\mathcal{H}_m$

Lemma 2. *Given a γ -preserving dictionary*

$$W(\mathbf{C}_{n,m}, \mu_n) - W(\mathbf{C}_n, \mu_n) \leq \frac{\min(k, d_{\text{eff}}^n(\gamma)) \gamma}{1 - \varepsilon} \frac{\gamma}{n}$$

To show this we can rewrite the objective as (see [17])

$$W(\mathbf{C}_{n,m}, \mu_n) = \|\Phi_n - \Pi_m \Phi_n \mathbf{S}_{n,m}\|_F^2 = \text{Tr}(\Phi_n^\top \Phi_n - \mathbf{S}_n \Phi_n^\top \Pi_m \Phi_n \mathbf{S}_n),$$

where $\mathbf{S}_n \in \mathbb{R}^{n \times n}$ is a k -rank *projection* matrix associated with the *exact* clustering $\mathbf{C}_n$. Then using Definition 2 we have $\Pi_m - \Pi_n \succeq -\frac{\gamma}{1-\varepsilon}(\Phi_n \Phi_n^\top + \gamma \Pi_n)^{-1}$ and we obtain an *additive* error bound

$$\begin{aligned} & \text{Tr}(\Phi_n^\top \Phi_n - \mathbf{S}_n \Phi_n^\top \Pi_m \Phi_n \mathbf{S}_n) \\ & \leq \text{Tr}\left(\Phi_n^\top \Phi_n - \mathbf{S}_n \Phi_n^\top \Phi_n \mathbf{S}_n + \frac{\gamma}{1-\varepsilon} \mathbf{S}_n \Phi_n^\top (\Phi_n \Phi_n^\top + \gamma \Pi_n)^{-1} \Phi_n \mathbf{S}_n\right) \\ & = W(\mathbf{C}_n, \mu_n) + \frac{\gamma}{1-\varepsilon} \text{Tr}(\mathbf{S}_n \Phi_n^\top (\Phi_n \Phi_n^\top + \gamma \Pi_n)^{-1} \Phi_n \mathbf{S}_n). \end{aligned}$$

¹ $\tilde{\mathcal{O}}$ hides logarithmic dependencies on n and m .

Since $\|\Phi_n^\top(\Phi_n\Phi_n^\top + \gamma\Pi_n)^{-1}\Phi_n\| \leq 1$, $\mathbf{S}_n$ is a projection matrix, and $\text{Tr}(\mathbf{S}_n) = k$ we have

$$\frac{\gamma}{1-\varepsilon} \text{Tr}(\mathbf{S}_n\Phi_n^\top(\Phi_n\Phi_n^\top + \gamma\Pi_n)^{-1}\Phi_n\mathbf{S}_n) \leq \frac{\gamma}{1-\varepsilon} \text{Tr}(\mathbf{S}_n\mathbf{S}_n) = \frac{\gamma k}{1-\varepsilon}.$$

Conversely, if we focus on the matrix $\Phi_n^\top(\Phi_n\Phi_n^\top + \gamma\Pi_n)^{-1}\Phi_n \preceq \Pi_n$ we have

$$\frac{\gamma}{1-\varepsilon} \text{Tr}(\mathbf{S}_n\Phi_n^\top(\Phi_n\Phi_n^\top + \gamma\Pi_n)^{-1}\Phi_n\mathbf{S}_n) \leq \frac{\gamma}{1-\varepsilon} \text{Tr}(\Phi_n^\top(\Phi_n\Phi_n^\top + \gamma\Pi_n)^{-1}\Phi_n) \leq \frac{\gamma d_{\text{eff}}^n(\gamma)}{1-\varepsilon}.$$

Since both bounds hold simultaneously, we can simply take the minimum to conclude our proof. $\square$

We now compare the theorem with previous work. Many approximate kernel k -means methods have been proposed over the years, and can be roughly split in two groups.

Low-rank *decomposition* based methods try to directly simplify the optimization problem from Proposition 2, replacing the kernel matrix $\mathbf{K}_n$ with an approximate $\tilde{\mathbf{K}}_n$ that can be stored and manipulated more efficiently. Among these methods we can mention partial decompositions [8], Nyström approximations based on uniform [36], k -means++ [27], or ridge leverage score (RLS) sampling [35, 25, 14], and random-feature approximations [6]. None of these optimization based methods focus on the underlying excess risk problem, and their analysis cannot be easily integrated in existing results, as the approximate minimum found has no clear interpretation as a statistical ERM.

Other works take the same *embedding* approach that we do, and directly replace the exact $\varphi(\cdot)$ with an approximate $\tilde{\varphi}(\cdot)$, such as Nyström embeddings [36], Gaussian projections [10], and again random-feature approximations [29]. Note that these approaches also result in approximate $\tilde{\mathbf{K}}_n$ that can be manipulated efficiently, but are simpler to analyze theoretically. Unfortunately, no existing embedding based methods can guarantee at the same time optimal excess risk rates and a reduction in the size of $\mathcal{H}_m$, and therefore a reduction in computational cost.

To the best of our knowledge, the only other result providing excess risk guarantee for approximate kernel k -means is Biau et al. [10], where the authors consider the excess risk of the ERM when the approximate $\mathcal{H}_m$ is obtained using Gaussian projections. Biau et al. [10] notes that the feature map $\varphi(\mathbf{x}) = \sum_{s=1}^D \psi_s(\mathbf{x})$ can be expressed using an expansion of basis functions $\psi_s(\mathbf{x})$, with D very large or infinite. Given a matrix $\mathbf{P} \in \mathbb{R}^{m \times D}$ where each entry is a standard Gaussian r.v., [10] proposes the following m -dimensional approximate feature map $\tilde{\varphi}(\mathbf{x}) = \mathbf{P}[\psi_1(\mathbf{x}), \dots, \psi_D(\mathbf{x})] \in \mathbb{R}^m$. Using Johnson-Lindenstrauss (JL) lemma [19], they show that if $m \geq \log(n)/\nu^2$ then a multiplicative error bound of the form $W(\mathbf{C}_{n,m}, \mu_n) \leq (1+\nu)W(\mathbf{C}_n, \mu_n)$ holds. Reformulating their bound, we obtain that $W(\mathbf{C}_{n,m}, \mu_n) - W(\mathbf{C}_n, \mu_n) \leq \nu W(\mathbf{C}_n, \mu_n) \leq \nu \kappa^2$ and $\mathcal{E}(\mathbf{C}_{n,m}) \leq \mathcal{O}(k/\sqrt{n} + \nu)$.

Note that to obtain a bound comparable to Theorem 1, and if we treat k as a constant, we need to take $\nu = \gamma/n$ which results in $m \geq (n/\gamma)^2$. This is always worse than our $\tilde{\mathcal{O}}(n/\gamma)$ result for uniform Nyström embedding. In particular, in the $1/\sqrt{n}$ risk rate setting Gaussian projections would require $\nu = 1/\sqrt{n}$ resulting in $m \geq n \log(n)$ random features, which would not bring any improvement over computing $\mathbf{K}_n$. Moreover when D is infinite, as it is usually the case in the non-parametric setting, the JL projection is not explicitly computable in general and Biau et al. [10] must assume the existence of a computational oracle capable of constructing $\tilde{\varphi}(\cdot)$. Finally note that, under the hood, traditional embedding methods such as those based on JL lemma, usually provide only bounds of the form $\Pi_n - \Pi_m \preceq \gamma \Pi_n$, and an error $\|\Pi_m^\perp \Phi_i\|^2 \leq \gamma \|\Phi_i\|^2$ (see the discussion of Definition 2). Therefore the error can be larger along multiple directions, and the overall error $\|\Pi_m^\perp \Phi_n\|^2$ across the dictionary can be as large as $n\gamma$ rather than γ .

Recent work in RLS sampling has also focused on bounding the distance $W(\mathbf{C}_{n,m}, \mu_n) - W(\mathbf{C}_n, \mu_n)$ between empirical errors. Wang et al. [35] and Musco and Musco [25] provide multiplicative error bounds of the form $W(\mathbf{C}_{n,m}, \mu_n) \leq (1+\nu)W(\mathbf{C}_n, \mu_n)$ for uniform and RLS sampling. Nonetheless, they only focus on empirical risk and do not investigate the interaction between approximation and generalization, i.e., statistics and computations. Moreover, as we already remarked for [10], to achieve the $1/\sqrt{n}$ excess risk rate using a multiplicative error bound we would require an unreasonably small ν , resulting in a large m that brings no computational improvement over the exact solution.

Finally, note that when [31] showed that a favourable trade-off was possible for kernel ridge regression (KRR), they strongly leveraged the fact that KRR is a γ -regularized problem. Therefore, all eigenvalues and eigenvectors in the $\Phi_n\Phi_n^\top$ covariance matrix smaller than the γ regularization do not influence significantly the solution. Here we show the same for kernel k -means, a problem

without regularization. This hints at a deeper geometric motivation which might be at the root of both problems, and potentially similar approaches could be leveraged in other domains.

4.1 Further results: beyond ERM

So far we provided guarantees for $\mathbf{C}_{n,m}$, that this is the ERM in $\mathcal{H}_m$. Although $\mathcal{H}_m$ is much smaller than $\mathcal{H}_n$, solving the optimization problem to find the ERM is still NP-Hard in general [4]. Nonetheless, Lloyd’s algorithm [22], when coupled with a careful k -means++ seeding, can return a good approximate solution $\mathbf{C}_{n,m}^{++}$.

Proposition 3 ([5]). *For any dataset $\mathbb{E}_{\mathcal{A}}[W(\mathbf{C}_{n,m}^{++}, \mu_n)] \leq 8(\log(k) + 2)W(\mathbf{C}_{n,m}, \mu_n)$, where $\mathcal{A}$ is the randomness deriving from the k -means++ initialization.*

Note that, similarly to [35, 25], this is a multiplicative error bound on the empirical risk, and as we discussed we cannot leverage Lemma 2 to bound the excess risk $\mathcal{E}(\mathbf{C}_{n,m}^{++})$. Nonetheless we can still leverage Lemma 2 to bound only the expected risk $W(\mathbf{C}_{n,m}^{++}, \mu)$, albeit with an extra error term appearing that scales with the optimal clustering risk $W^*(\mu)$ (see Proposition 1).

Theorem 2. *Given a γ -preserving dictionary*

$$\mathbb{E}_{\mathcal{D} \sim \mu} \left[\mathbb{E}_{\mathcal{A}} [W(\mathbf{C}_{n,m}^{++}, \mu)] \right] \leq \mathcal{O} \left(\log(k) \left(\frac{k}{\sqrt{n}} + k \frac{\gamma}{n} + W^*(\mu) \right) \right).$$

From a statistical perspective, we can once again, set $\gamma = \sqrt{n}$ to obtain a $\mathcal{O}(k/\sqrt{n})$ rate for the first part of the bound. Conversely, the optimal clustering risk $W^*(\mu)$ is a μ -dependent quantity that cannot in general be bounded in n , and captures how well our model, i.e., the choice of $\mathcal{H}$ and how well the criterion $W(\cdot, \mu)$, matches reality.

From a computational perspective, we can now bound the computational cost of finding $\mathbf{C}_{n,m}^{++}$. In particular, each iteration of Lloyd’s algorithm will take only $\tilde{\mathcal{O}}(n\sqrt{n}k)$ time. Moreover, when k -means++ initialization is used, the expected number of iterations required for Lloyd’s algorithm to converge is only logarithmic [1]. Therefore, ignoring the time required to embed the points, we can find a solution in $\tilde{\mathcal{O}}(n\sqrt{n}k)$ time and space instead of the $\tilde{\mathcal{O}}(n^2k)$ cost required by the exact method, with a strong $\mathcal{O}(\sqrt{n})$ improvement.

Finally, if the data distribution satisfies some regularity assumption the following result follows [15].

Corollary 1. *If we denote by $\mathcal{X}_\mu$ the support of the distribution μ and assume $\varphi(\mathcal{X}_\mu)$ to be a d -dimensional manifold, then $W^*(\mu) \leq dk^{-2/d}$, and given a $\sqrt{n}$ -preserving dictionary the expected cost satisfies $\mathbb{E}_{\mathcal{D} \sim \mu} [\mathbb{E}_{\mathcal{A}} [W(\mathbf{C}_{n,m}^{++}, \mu)]] \leq \mathcal{O} \left(\log(k) \left(\frac{k}{\sqrt{n}} + dk^{-2/d} \right) \right)$.*

5 Experiments

We now evaluate experimentally the claims of Theorem 1, namely that sampling $\tilde{\mathcal{O}}(n/\gamma)$ increases the excess risk by an extra γ/n factor, and that $m = \sqrt{n}$ is sufficient to recover the optimal rate. We use the `Nystroem` and `MiniBatchKmeans` classes from the `sklearn` python library [28] to implement kernel k -means with Nyström embedding (Algorithm 1) and we compute the solution $\mathbf{C}_{n,m}^{++}$.

For our experiments we follow the same approach as Wang et al. [35], and test our algorithm on two variants of the MNIST digit dataset. In particular, MNIST60K [20] is the original MNIST dataset containing pictures each with $d = 784$ pixels. We divide each pixel by 255, bringing each feature in a $[0, 1]$ interval. We split the dataset in two part, $n = 60000$ samples are used to compute the $W(\mathbf{C}_{n,m}^{++})$ centroids, and we leave out unseen 10000 samples to compute $W(\mathbf{C}_{n,m}^{++}, \mu_{test})$, as a proxy for $W(\mathbf{C}_{n,m}^{++}, \mu)$. To test the scalability of our approach we also consider the MNIST8M dataset from the infinite MNIST project [23], constructed using non-trivial transformations and corruptions of the original MNIST60K images. Here we compute $\mathbf{C}_{n,m}^{++}$ using $n = 8000000$ images, and compute $W(\mathbf{C}_{n,m}^{++}, \mu_{test})$ on 100000 unseen images. As in Wang et al. [35] we use Gaussian kernel with bandwidth $\sigma = (1/n^2) \sqrt{\sum_{i,j} \|\mathbf{x}_i - \mathbf{x}_j\|^2}$.

MNIST60K: these experiments are small enough to run in less than a minute on a single laptop with 4 cores and 8GB of RAM. The results are reported in Fig. 1. On the left we report in blue

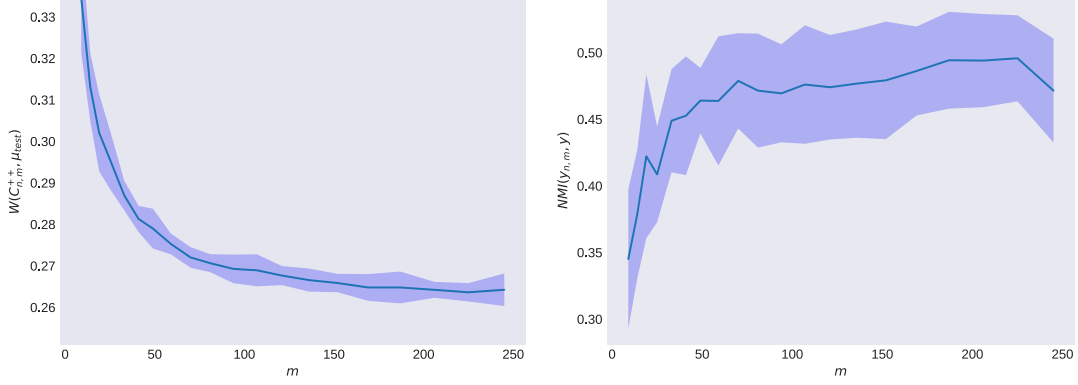


Figure 1: Results for MNIST60K

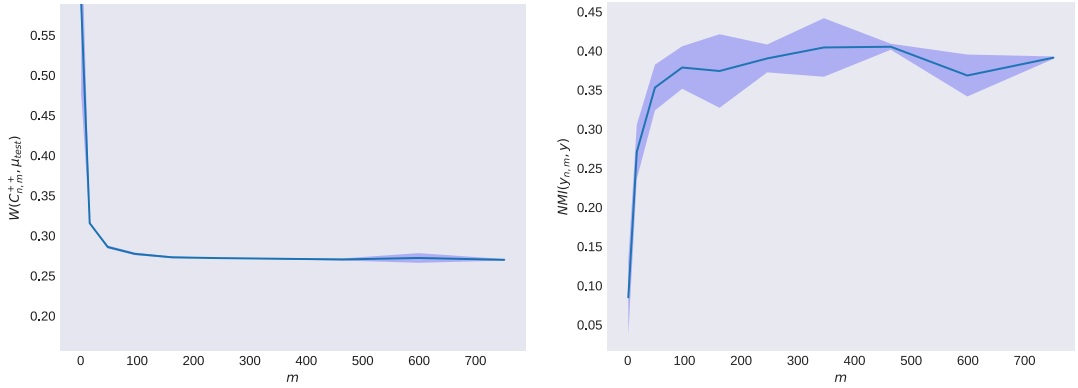


Figure 2: Results for MNIST8M

$W(\mathbf{C}_{n,m}^{++}, \mu_{test})$, where the shaded region is a 95% confidence interval for the mean over 10 runs. As predicted, the expected cost decreases as the size of $\mathcal{H}_m$ increases, and plateaus once we achieve $1/m \simeq 1/\sqrt{n}$, in line with the statistical error. Note that the normalized mutual information (NMI) between the true $[0 - 9]$ digit classes $\mathbf{y}$ and the computed cluster assignments $\mathbf{y}_{n,m}$ also plateaus around $1/\sqrt{n}$. While this is not predicted by the theory, it strengthens the intuition that beyond a certain capacity expanding $\mathcal{H}_m$ is computationally wasteful.

MNIST8M: to test the scalability of our approach, we run the same experiment on millions of points. Note that we carry out our MNIST8M experiment on a *single* 36 core machine with 128GB of RAM, much less than the setup of [35], where at minimum a cluster of 8 such nodes are used. The behaviour of $W(\mathbf{C}_{n,m}^{++}, \mu_{test})$ and NMI are similar to MNIST60K, with the increase in dataset size allowing for stronger concentration and smaller confidence intervals. Finally, note that around $m = 400$ uniformly sampled landmarks are sufficient to achieve $NMI(\mathbf{y}_{n,m}, \mathbf{y}) = 0.405$, matching the 0.406 NMI reported by [35] for a larger $m = 1600$, although smaller than the 0.423 NMI they report for $m = 1600$ when using a slower, PCA based method to compute the embeddings, and RLS sampling to select the landmarks. Nonetheless, computing $\mathbf{C}_{n,m}^{++}$ takes less than 6 minutes on a single machine, while their best solution required more than 1.5hr on a cluster of 32 machines.

6 Open questions and conclusions

Combining Lemma 1 and Lemma 2, we know that using uniform sampling we can linearly trade-off a $1/\gamma$ decrease in sub-space size m with a γ/n increase in excess risk. While this is sufficient to maintain the $\mathcal{O}(1/\sqrt{n})$ rate, it is easy to see that the same would not hold for a $\mathcal{O}(1/n)$ rate, since we would need to uniformly sample $n/1$ landmarks losing all computational improvements.

To achieve a better trade-off we must go beyond uniform sampling and use different probabilities for each sample, to capture their uniqueness and contribution to the approximation error.

Definition 3 ([3]). *The γ -ridge leverage score (RLS) of point $i \in [n]$ is defined as*

$$\tau_i(\gamma) = \phi_i^\top (\Phi_n \Phi_n^\top + \gamma \mathbf{\Pi}_n)^{-1} \phi_i = \mathbf{e}_i^\top \mathbf{K}_n (\mathbf{K}_n + \gamma \mathbf{I}_n)^{-1} \mathbf{e}_i. \quad (5)$$

The sum of the RLSs $d_{\text{eff}}^n(\gamma) = \sum_{i=1}^n \tau_i(\gamma)$ is the empirical effective dimension of the dataset.

Ridge leverage scores are closely connected to the residual $\|\mathbf{\Pi}_m^\perp \phi_i\|^2$ after the projection $\mathbf{\Pi}_m$ discussed in Definition 2. In particular, using Lemma 2 we have that the residual can be bounded as $\|\mathbf{\Pi}_m^\perp \phi_i\|^2 \leq \frac{\gamma}{1-\varepsilon} \phi_i^\top (\Phi_n \Phi_n^\top + \gamma \mathbf{\Pi}_n)^{-1} \phi_i$. It is easy to see that, up to a factor $\frac{\gamma}{1-\varepsilon}$, high-RLS points are also high-residual points. Therefore it is not surprising that sampling according to RLSs quickly selects any high-residual points and covers $\mathcal{H}_n$, generating a γ -preserving dictionary.

Lemma 3. [11] *For a given γ , construct $\mathcal{I}$ by sampling $m \geq 12\kappa^2 d_{\text{eff}}^n(\gamma) \log(n/\delta)/\varepsilon^2$ landmarks from $\mathcal{D}$ proportionally to their RLS. Then w.p. at least $1 - \delta$ the dictionary $\mathcal{I}$ is γ -preserving.*

Note there exist datasets where the RLSs are uniform, and therefore in the worst case the two sampling approaches coincide. Nonetheless, when the data is more structured $m \simeq d_{\text{eff}}^n(\gamma)$ can be much smaller than the n/γ dictionary size required by uniform sampling.

Finally, note that computing RLSs exactly also requires constructing $\mathbf{K}_n$ and $\mathcal{O}(n^2)$ time and space, but in recent years a number of fast approximate RLSs sampling methods [14] have emerged that can construct γ -preserving dictionaries of size $\tilde{\mathcal{O}}(d_{\text{eff}}^n(\gamma))$ in just $\tilde{\mathcal{O}}(nd_{\text{eff}}^n(\gamma)^2)$ time. Using this result, it is trivial to sharpen the computational aspects of Theorem 1 in special cases.

In particular, we can generate a $\sqrt{n}$ -preserving dictionary with only $d_{\text{eff}}^n(\sqrt{n})$ elements instead of the $\sqrt{n}$ required by uniform sampling. Using concentration arguments [31] we also know that w.h.p. the empirical effective dimension is at most three times $d_{\text{eff}}^n(\gamma) \leq 3d_{\text{eff}}^\mu(\gamma)$ the expected effective dimension, a μ -dependent quantity that captures the interaction between μ and the RKHS $\mathcal{H}$.

Definition 4. *Given the expected covariance operator $\mathbf{\Psi} := \mathbb{E}_{\mathbf{x} \sim \mu} [\phi(\mathbf{x})\phi(\mathbf{x})^\top]$, the expected effective dimension is defined as $d_{\text{eff}}^\mu(\gamma) = \mathbb{E}_{\mathbf{x} \sim \mu} \left[\phi(\mathbf{x}) (\mathbf{\Psi} + \gamma \mathbf{\Pi})^{-1} \phi(\mathbf{x}) \right]$. Moreover, for some constant c that depends only on $\varphi(\cdot)$ and μ , $d_{\text{eff}}^\mu(\gamma) \leq c(n/\gamma)^\eta$ with $0 < \eta \leq 1$.*

Note that $\eta = 1$ just gives us the $d_{\text{eff}}^\mu(\gamma) \leq \mathcal{O}(n/\gamma)$ worst-case upper bound that we saw for $d_{\text{eff}}^n(\gamma)$, and it is always satisfied when the kernel function is bounded. If instead we have a faster spectral decay, η can be much smaller. For example, if the eigenvalues of $\mathbf{\Psi}$ decay polynomially as $\lambda_i = i^{-\eta}$, then $d_{\text{eff}}^\mu(\gamma) \leq c(n/\gamma)^\eta$, and in our case $\gamma = \sqrt{n}$ we have $d_{\text{eff}}^\mu(\sqrt{n}) \leq cn^{\eta/2}$.

We can now better characterize the gap between statistics and computation: using RLSs sampling we can improve the computational aspect of Theorem 1 from $\sqrt{n}$ to $d_{\text{eff}}^\mu(\gamma)$, but the risk rate remains $\mathcal{O}(k/\sqrt{n})$ due to the $\mathcal{O}(k/\sqrt{n})$ component coming from Proposition 1.

Assume for a second we could generalize, with additional assumptions, Proposition 1 to a faster $\mathcal{O}(1/n)$ rate. Then applying Lemma 2 with $\gamma = 1$ we would obtain a risk $\mathcal{E}(\mathbf{C}_{n,m}) \leq \mathcal{O}(k/n) + \mathcal{O}(k/n)$. Here we see how the regularity condition on $d_{\text{eff}}^\mu(1)$ becomes crucial. In particular, if $\eta = 1$, then we have $d_{\text{eff}}^\mu(1) \sim n$ and no gain. If instead $\eta < 1$ we obtain $d_{\text{eff}}^\mu(1) \leq n^\eta$. This kind of adaptive rates were shown to be possible in supervised learning [31], but seems to still be out of reach for approximate kernel k -means.

One possible approach to fill this gap is to look at fast $\mathcal{O}(1/n)$ excess risk rates for kernel k -means.

Proposition 4 ([21], informal). *Assume that $k \geq 2$, and that μ satisfies a margin condition with radius r_0 . If $\mathbf{C}_n$ is an empirical risk minimizer, then, with probability larger than $1 - e^{-\delta}$,*

$$\mathcal{E}(\mathbf{C}_n) \leq \tilde{\mathcal{O}} \left(\frac{1}{r_0} \frac{(k + \log(|M|)) \log(1/\delta)}{n} \right),$$

where $|M|$ is the cardinality of the set of all optimal (up to a relabeling) clustering.

For more details on the margin assumption, we refer the reader to the original paper [21]. Intuitively the margin condition asks that every labeling (Voronoi grouping) associated with an optimal clustering is reflected by large separation in μ . This margin condition also acts as a counterpart of the usual margin conditions for supervised learning where μ must have lower density around the neighborhood of the critical area $\{\mathbf{x} | \mu'(Y = 1 | X = \mathbf{x}) = 1/2\}$. Unfortunately, it is not easy to integrate Proposition 4 in our analysis, as it is not clear how the margin condition translate from $\mathcal{H}_n$ to $\mathcal{H}_m$.

Acknowledgments.

This material is based upon work supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216, and the Italian Institute of Technology. We gratefully acknowledge the support of NVIDIA Corporation for the donation of the Titan Xp GPUs and the Tesla k40 GPU used for this research. L. R. acknowledges the support of the AFOSR projects FA9550-17-1-0390 and BAA-AFRL-AFOSR-2016-0007 (European Office of Aerospace Research and Development), and the EU H2020-MSCA-RISE project NoMADS - DLV-777826. A. R. acknowledges the support of the European Research Council (grant SEQUOIA 724063).

References

- [1] Nir Ailon, Ragesh Jaiswal, and Claire Monteleoni. Streaming k-means approximation. In *Advances in neural information processing systems*, pages 10–18, 2009.
- [2] M. A. Aizerman, E. A. Braverman, and L. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. In *Automation and Remote Control*, number 25 in *Automation and Remote Control*, pages 821–837, 1964.
- [3] Ahmed El Alaoui and Michael W. Mahoney. Fast randomized kernel methods with statistical guarantees. In *Neural Information Processing Systems*, 2015.
- [4] Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat. Np-hardness of euclidean sum-of-squares clustering. *Machine learning*, 75(2):245–248, 2009.
- [5] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.
- [6] Haim Avron, Michael Kapralov, Cameron Musco, Christopher Musco, Ameya Velingker, and Amir Zandieh. Random Fourier features for kernel ridge regression: Approximation bounds and statistical guarantees. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.
- [7] Francis Bach. Sharp analysis of low-rank kernel matrix approximations. In *Conference on Learning Theory*, 2013.
- [8] Francis R Bach and Michael I Jordan. Predictive low-rank decomposition for kernel methods. In *Proceedings of the 22nd international conference on Machine learning*, pages 33–40. ACM, 2005.
- [9] Arturs Backurs, Piotr Indyk, and Ludwig Schmidt. On the fine-grained complexity of empirical risk minimization: Kernel methods and neural networks. In *Advances in Neural Information Processing Systems*, 2017.
- [10] Gérard Biau, Luc Devroye, and Gábor Lugosi. On the performance of clustering in hilbert spaces. *IEEE Transactions on Information Theory*, 54(2):781–790, 2008.
- [11] Daniele Calandriello. *Efficient Sequential Learning in Structured and Constrained Environments*. PhD thesis, 2017.
- [12] Daniele Calandriello, Alessandro Lazaric, and Michal Valko. Efficient second-order online kernel learning with adaptive embedding. In *Advances in Neural Information Processing Systems*, pages 6140–6150, 2017.
- [13] Daniele Calandriello, Alessandro Lazaric, and Michal Valko. Second-order kernel online convex optimization with adaptive sketching. In *International Conference on Machine Learning*, 2017.
- [14] Daniele Calandriello, Alessandro Lazaric, and Michal Valko. Distributed adaptive sampling for kernel matrix approximation. In *AISTATS*, 2017.
- [15] Guillermo Canas, Tomaso Poggio, and Lorenzo Rosasco. Learning manifolds with k-means and k-flats. In *Advances in Neural Information Processing Systems*, pages 2465–2473, 2012.

- [16] Radha Chitta, Rong Jin, Timothy C. Havens, and Anil K. Jain. Approximate kernel k-means: Solution to large scale kernel clustering. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 895–903. ACM, 2011.
- [17] Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis. *A unified view of kernel k-means, spectral clustering and graph cuts*. Technical Report No. UTCS TR-04-25, University of Texas at Austin, 2004.
- [18] Siegfried Graf and Harald Luschgy. *Foundations of Quantization for Probability Distributions*. Lecture Notes in Mathematics. Springer-Verlag, Berlin Heidelberg, 2000. ISBN 978-3-540-67394-1.
- [19] William B Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. *Contemporary Mathematics*, 26, 1984.
- [20] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL <http://yann.lecun.com/exdb/mnist/>.
- [21] Clément Levrard et al. Nonasymptotic bounds for vector quantization in hilbert spaces. *The Annals of Statistics*, 43(2):592–619, 2015.
- [22] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.
- [23] Gaëlle Loosli, Stéphane Canu, and Léon Bottou. Training invariant support vector machines using selective sampling. In *Large Scale Kernel Machines*, pages 301–320. MIT Press, Cambridge, MA., 2007.
- [24] Andreas Maurer and Massimiliano Pontil. K -dimensional coding schemes in Hilbert spaces. *IEEE Transactions on Information Theory*, 56(11):5839–5846, 2010.
- [25] Cameron Musco and Christopher Musco. Recursive Sampling for the Nyström Method. In *NIPS*, 2017.
- [26] Cameron Musco and David Woodruff. Is input sparsity time possible for kernel low-rank approximation? In *Advances in Neural Information Processing Systems 30*. 2017.
- [27] Dino Oglic and Thomas Gärtner. Nyström method with kernel k-means++ samples as landmarks. *Journal of Machine Learning Research*, 2017.
- [28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [29] Ali Rahimi and Ben Recht. Random features for large-scale kernel machines. In *Neural Information Processing Systems*, 2007.
- [30] Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. In *Advances in Neural Information Processing Systems*, pages 3218–3228, 2017.
- [31] Alessandro Rudi, Raffaello Camoriano, and Lorenzo Rosasco. Less is more: Nyström computational regularization. In *Advances in Neural Information Processing Systems*, pages 1657–1665, 2015.
- [32] Bernhard Scholkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.
- [33] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.
- [34] Joel A Tropp, Alp Yurtsever, Madeleine Udell, and Volkan Cevher. Fixed-rank approximation of a positive-semidefinite matrix from streaming data. In *Advances in Neural Information Processing Systems*, pages 1225–1234, 2017.

Gaussian Process Optimization with Adaptive Sketching: Scalable and No Regret

Daniele Calandriello

DANIELE.CALANDRIELLO@IIT.IT

LCSL - Istituto Italiano di Tecnologia, Genova, Italy & MIT, Cambridge, USA

Luigi Carratino

LUIGI.CARRATINO@DIBRIS.UNIGE.IT

UNIGE - Università degli Studi di Genova, Genova, Italy

Alessandro Lazaric

LAZARIC@FB.COM

FAIR - Facebook AI Research, Paris, France

Michal Valko

MICHAL.VALKO@INRIA.FR

INRIA Lille - Nord Europe, SequeL team, Lille, France

Lorenzo Rosasco

LROSASCO@MIT.EDU

LCSL - Istituto Italiano di Tecnologia, Genova, Italy & MIT, Cambridge, USA

UNIGE - Università degli Studi di Genova, Genova, Italy

Editors: Alina Beygelzimer and Daniel Hsu

Abstract

Gaussian processes (GP) are a stochastic processes, used as Bayesian approach for the optimization of black-box functions. Despite their effectiveness in simple problems, GP-based algorithms hardly scale to high-dimensional functions, as their per-iteration time and space cost is at least *quadratic* in the number of dimensions d and iterations t . Given a set of A alternatives to choose from, the overall runtime $\mathcal{O}(t^3 A)$ is prohibitive. In this paper, we introduce BKB (*budgeted kernelized bandit*), a new approximate GP algorithm for optimization under bandit feedback that achieves near-optimal regret (and hence near-optimal convergence rate) with near-constant per-iteration complexity and remarkably no assumption on the input space or covariance of the GP.

We combine a kernelized linear bandit algorithm (GP-UCB) leverage score sampling as a randomized matrix sketching and prove that selecting inducing points based on their posterior variance gives an accurate low-rank approximation of the GP, preserving variance estimates and confidence intervals. As a consequence, BKB does not suffer from *variance starvation*, an important problem faced by many previous sparse GP approximations. Moreover, we show that our procedure selects at most $\tilde{\mathcal{O}}(d_{\text{eff}})$ points, where d_{eff} is the *effective* dimension of the explored space, which is typically much smaller than both d and t . This greatly reduces the dimensionality of the problem, thus leading to a $\mathcal{O}(T A d_{\text{eff}}^2)$ runtime and $\mathcal{O}(A d_{\text{eff}})$ space complexity.

Keywords: *sparse Gaussian process optimization; kernelized linear bandits; regret; sketching; Bayesian optimization; black-box optimization; variance starvation*

1. Introduction

Efficiently selecting the best alternative out of a set of alternatives is important in sequential decision making, with practical applications ranging from recommender systems (Li et al., 2010) to experimental design (Robbins, 1952). It is also the main focus of the research in bandits (Lattimore and Szepesvári, 2019) and Bayesian optimization (Mockus, 1989; Pelikan, 2005; Snoek et al., 2012),

Extended abstract. Full version appears as <https://arxiv.org/abs/1903.05594>.

that study optimization under bandit feedback. In this setting, a learning algorithm sequentially interacts with a reward or utility function f . Over T interactions, the algorithm chooses a point $\mathbf{x}_t$ and it has only access to a noisy black-box evaluation of f at $\mathbf{x}_t$. The goal of the algorithm is to minimize the cumulative regret, which compares the reward accumulated at the points selected over time, $\sum_t f(\mathbf{x}_t)$, to the reward obtained by repeatedly selecting the optimum of the function, i.e., $T \max_x f(x)$. In this paper take the Gaussian process optimization approach. In particular, we study the GP-UCB algorithm first introduced by [Srinivas et al. \(2010\)](#).

Starting from a Gaussian process prior over f , GP-UCB alternates between evaluating the function, x and using the evaluations to build a posterior of f . This posterior is composed by a mean function μ that estimates the value of f , and a variance function σ that captures the uncertainty μ . These two quantities are combined in a single upper confidence bound (UCB) that drives the selection of the evaluation points, and trades off between evaluating high-reward points (*exploitation*) and testing possibly sub-optimal points to reduce the uncertainty on the function (*exploration*). The performance of GP-UCB has been studied by [Srinivas et al. \(2010\)](#); [Valko et al. \(2013\)](#); [Chowdhury and Gopalan \(2017\)](#) to show that GP-UCB provably achieves low regret both in a Bayesian and non-Bayesian setting. However, the main limiting factor to its applicability is its computational cost. When choosing between A alternatives, GP-UCB requires $\Omega(At^2)$ time/space to select each new point, and therefore does not scale to complex functions. Several approximations of GP-UCB have been suggested ([Quinero-Candela et al., 2007](#); [Liu et al., 2018](#)) and we review them next:

Inducing points: The GP can be restricted to lie in the range of a small subset of inducing points. The subset should cover the space well for accuracy, but also be as small as possible for efficiency. Methods referred to as *sparse GPs*, have been proposed to select the inducing points and an approximation based on the subset. Popular instances of this approach are the subset of regressors (SoR, [Wahba, 1990](#)) and the deterministic training conditional (DTC, [Seeger et al., 2003](#)). While these methods are simple to interpret and efficient, they do not come with regret guarantees. Moreover, when the subset does not cover the space well, they suffer from *variance starvation* ([Wang et al., 2018](#)), as they underestimate the variance of points far away from the inducing points.

Random Fourier features: Another approach is to use explicit feature expansions to approximate the GP covariance function, and embed the points in a low-dimensional space, usually exploiting some variation of Fourier expansions ([Rahimi and Recht, 2007](#)). Among these methods, [Mutný and Krause \(2018\)](#) recently showed that discretizing the posterior on a fine grid of quadrature Fourier features (QFF) incurs a negligible approximation error. It is sufficient to prove that the maximum of the approximate posterior can be efficiently found and that it is accurate enough that Thompson sampling with quadratures provably achieves low regret. However this approach does not extend to non-stationary (or translation invariant) kernels and although its dependence on t is small, the approximation and posterior maximization procedure scales exponentially with the input dimension.

Variational inference: The next approach replaces the true GP likelihood with a variational approximation that can be optimized efficiently. Although recent methods provide guarantees on the approximate posterior mean and variance ([Huggins et al., 2019](#)), these guarantees only apply to GP regression and not to the harder optimization setting.

Linear case: In the linear bandits, there are methods that reduce the complexity of algorithms such as LinUCB have. [Kuzborskij et al. \(2019\)](#) uses the *frequent directions* (FD, [Ghashami et al., 2016](#)) to project the design matrix data to a smaller subspace. Unfortunately, the size of the subspace has to be specified in advance and when the size is not sufficiently large, the method suffers linear regret. Prior

to the FD approach, CBRAP (Yu et al., 2017) used random projections instead, but faced similar issues. This turns out to be a fundamental weakness of all approaches that do not adapt the *actual* size of the space defined by the sequence of points selected by the learning algorithm. Indeed, Ghosh et al. (2017) showed a lower bound for the linear case that shows that as soon as one single arm does not abide by the linear model we can suffer linear regret.

1.1. Contributions

In this paper, we show a way to adapt the size of the projected space online and devise the BKB (budgeted kernel bandit) algorithm achieving near-optimal regret with a computational complexity drastically smaller than GP-UCB. This is achieved *without assumptions on the complexity* of the input or on the kernel function. BKB leverages several well-known tools: a DTC approximation of the posterior variance, based on inducing points, and a confidence interval construction based on state-of-the-art self-normalized concentration inequalities (Abbasi-Yadkori et al., 2011). It also introduces two novel tools: a selection strategy to select inducing points based on ridge leverage score (RLS) sampling (Alaoui and Mahoney, 2015) that is provably accurate, and an approximate confidence interval that is not only nearly as accurate as the one of GP-UCB, but also efficient.

Ridge leverage score sampling was introduced for randomized kernel matrix approximation (Alaoui and Mahoney, 2015). In the context of GPs, RLS correspond to the posterior variance of a point, which allows adapting algorithms and their guarantees from the RLS sampling to the GP setting. This solves two problems in sparse GPs and linear bandit approximations. First, BKB constructs estimates of the variance that are provably accurate, i.e., it does not suffer from variance starvation, which results in provably accurate confidence bounds as well. The only method with comparable guarantees, Thompson sampling with quadrature FF (Mutn y and Krause, 2018), only applies to stationary kernels, and only in extremely low dimension. Moreover our approximation guarantees are qualitatively different since they do not require a corresponding *uniform* approximation bound on the GP. Second, BKB adaptively chooses the size of the inducing point set based on the *effective dimension* d_{eff} of the problem. This is crucial to achieve low regret, since fixed approximation schemes may suffer linear regret. Moreover, in a problem with A arms, using a set of $\mathcal{O}(d_{\text{eff}})$ inducing points results in an algorithm with $\mathcal{O}(Ad_{\text{eff}}^2)$ per-step runtime and $\mathcal{O}(Ad_{\text{eff}})$ space, a significant improvement over the $\mathcal{O}(At^2)$ time and $\mathcal{O}(At)$ space cost of GP-UCB.

Finally, while in our work we only address kernelized (GP) bandits, our work could be extended to more complex online learning problems, such as to recent advances in kernelized reinforcement learning (Chowdhury and Gopalan, 2019). Moreover, inducing point methods have clear interpretability and our analysis provides insight both from a bandit and Bayesian optimization perspective, making it applicable to a large amount of downstream tasks.

2. Background

Notation. We use lower-case letters a for scalars, lower-case bold letters $\mathbf{a}$ for vectors, and upper-case bold letters $\mathbf{A}$ for matrices and operators, where $[\mathbf{A}]_{ij}$ denotes its element (i, j) . We denote by $\|\mathbf{x}\|_{\mathbf{A}}^2 \triangleq \mathbf{x}^\top \mathbf{A} \mathbf{x}$, the norm with metric $\mathbf{A}$, and $\|\mathbf{x}\| \triangleq \|\mathbf{x}\|_{\mathbf{I}}$ with $\mathbf{I}$ being the identity. Finally, we denote the first T integers as $[T] \triangleq \{1, \dots, T\}$.

Online optimization under bandit feedback. Let $f : \mathcal{A} \rightarrow \mathbb{R}$ be a reward function that we wish to optimize over a set of decisions $\mathcal{A}$, also called actions or arms. For simplicity, we assume that $\mathcal{A} \triangleq \{\mathbf{x}_i\}_{i=1}^A$ is a fixed finite set of A vectors in $\mathbb{R}^d$. We discuss how to relax these assumptions in

Section 5. In optimization under bandit feedback, a learner aims to optimize f through a sequence of interactions. At each step $t \in [T]$, the learner (1) chooses an arm $\mathbf{x}_t \in \mathcal{A}$, (2) receives reward $y_t \triangleq f(\mathbf{x}_t) + \eta_t$, where η_t is a zero-mean noise, (3) updates its model of the problem.

The goal of the learner is to minimize its cumulative regret $R_T \triangleq \sum_{t=1}^T f(\mathbf{x}_*) - f(\mathbf{x}_t)$ w.r.t. the best¹ $\mathbf{x}_*$, where $\mathbf{x}_* \triangleq \arg \max_{\mathbf{x}_i \in \mathcal{A}} f(\mathbf{x}_i)$. In particular, the objective of a *no-regret* algorithm is to have R_T/T go to zero as T grows as fast as possible. Recall that the regret is related to the convergence rate and the optimization performance. In fact, let $\bar{\mathbf{x}}_T$ be an arm chosen at random from the sequence of arms $(\mathbf{x}_1, \dots, \mathbf{x}_T)$ selected by the learner, then $f(\mathbf{x}_*) - \mathbb{E}[f(\bar{\mathbf{x}}_T)] \leq R_T/T$.

Gaussian process optimization and GP-UCB

GP-UCB is popular no-regret algorithm for optimization under bandit feedback and was introduced by Srinivas et al. (2010) for Gaussian process optimization. We first give the formal definition of a Gaussian process (Rasmussen and Williams, 2006), and then briefly present GP-UCB.

A *Gaussian process* $\text{GP}(\mu, k)$ is a generalization of the Gaussian distribution to a space of functions and it is defined by a mean function $\mu(\cdot)$ and covariance function $k(\cdot, \cdot)$. We consider zero-mean $\text{GP}(0, k)$ priors and bounded covariance $k(\mathbf{x}_i, \mathbf{x}_i) \leq \kappa^2$ for all $\mathbf{x}_i \in \mathcal{A}$. An important property of Gaussian processes is that if we combine a prior $f \sim \text{GP}(0, k)$ and assume that the observation noise is zero-mean Gaussian (i.e., $\eta_t \sim \mathcal{N}(0, \xi^2)$), then the posterior distribution of f conditioned on a set of observations $\{(\mathbf{x}_s, y_s)\}_{s=1}^t$ is also a GP. More precisely, if $\mathbf{X}_t \triangleq [\mathbf{x}_1, \dots, \mathbf{x}_t]^\top \in \mathbb{R}^{t \times d}$ is the matrix with all arms selected so far and $\mathbf{y}_t \triangleq [y_1, \dots, y_t]^\top$ the corresponding observations, then the posterior is still a GP and the mean and variance of the function at a test point $\mathbf{x}$ are defined as

$$\mu_t(\mathbf{x} \mid \mathbf{X}_t, \mathbf{y}_t) = \mathbf{k}_t(\mathbf{x})^\top (\mathbf{K}_t + \lambda \mathbf{I})^{-1} \mathbf{y}_t, \quad (1)$$

$$\sigma_t^2(\mathbf{x} \mid \mathbf{X}_t) = k(\mathbf{x}, \mathbf{x}) - \mathbf{k}_t(\mathbf{x})^\top (\mathbf{K}_t + \lambda \mathbf{I})^{-1} \mathbf{k}_t(\mathbf{x}), \quad (2)$$

where $\lambda \triangleq \xi^2$, $\mathbf{K}_t \in \mathbb{R}^{t \times t}$ is the matrix $[\mathbf{K}_t]_{i,j} \triangleq k(\mathbf{x}_i, \mathbf{x}_j)$ constructed from all pairs $\mathbf{x}_i, \mathbf{x}_j$ in $\mathbf{X}_t$, and $\mathbf{k}_t(\mathbf{x}) \triangleq [k(\mathbf{x}_1, \mathbf{x}), \dots, k(\mathbf{x}_t, \mathbf{x})]^\top$. Notice that $\mathbf{k}_t(\mathbf{x})$ can be seen as an *embedding* of an arm $\mathbf{x}$ represented using by the arms $\mathbf{x}_1, \dots, \mathbf{x}_t$ observed so far.

The GP-UCB *algorithm* uses a Gaussian process $\text{GP}(0, k)$ as a prior for f . Inspired by the optimism-in-face-of-uncertainty principle, at each time step t , GP-UCB uses the posterior GP to compute the mean and variance of an arm $\mathbf{x}_i$ and obtain the score

$$u_t(\mathbf{x}_i) \triangleq \mu_t(\mathbf{x}_i) + \beta_t \sigma_t(\mathbf{x}_i), \quad (3)$$

where we use the short-hand notation $\mu_t(\cdot) \triangleq \mu(\cdot \mid \mathbf{X}_t, \mathbf{y}_t)$ and $\sigma_t(\cdot) \triangleq \sigma(\cdot \mid \mathbf{X}_t)$. Finally, GP-UCB chooses the maximizer $\mathbf{x}_{t+1} \triangleq \arg \max_{\mathbf{x}_i \in \mathcal{A}} u_t(\mathbf{x}_i)$ as the next arm to evaluate. According to the score u_t , an arm $\mathbf{x}$ is likely to be selected if it has high mean reward μ_t or high variance σ_t , i.e., its estimated reward $\mu_t(\mathbf{x})$ is very uncertain. As a result, selecting the arm $\mathbf{x}_{t+1}$ with the largest score trades off between collecting (estimated) large reward (*exploitation*) and improving the accuracy of the posterior (*exploration*). The parameter β_t balances between these two objectives and must be properly tuned to guarantee low regret. Srinivas et al. (2010) proposes different approaches for tuning β_t , depending on the assumptions on f and $\mathcal{A}$.

While GP-UCB is interpretable, simple to implement and provably achieves low regret, it is computationally expensive. In particular, computing $\sigma_t(\mathbf{x})$ has a complexity at least $\Omega(t^2)$.

1. We assume a consistent and arbitrary tie-breaking strategy.

Multiplying this complexity by T iterations and A arms results in an overall $\mathcal{O}(AT^3)$ cost, which does not scale to large number of iterations T .

3. Budgeted Kernel Bandits

In this section, we introduce the BKB (*budgeted kernel bandit*) algorithm, a novel efficient approximation of GP-UCB, and we provide guarantees for its computational complexity. The analysis in Section 4 shows that BKB can be tuned to significantly reduce the complexity of GP-UCB with a negligible impact on the regret. We begin by introducing the two major contributions of this section: an approximation of the GP-UCB scores supported only by a small subset $\mathcal{S}_t$ of *inducing points*, and a method to *incrementally and adaptively* construct an accurate subset $\mathcal{S}_t$.

3.1. The algorithm

The main computational bottleneck for getting the scores in Equation 3 is due to the fact that after t steps, the posterior GP is supported on *all* t previously seen arms. As a consequence, evaluating Equations 1 and 2 requires computing a t dimensional vector $\mathbf{k}_t(\mathbf{x})$ and $t \times t$ matrix $\mathbf{K}_t$ respectively. To avoid this dependency we restrict both $\mathbf{k}_t$ and $\mathbf{K}_t$ to be supported on a *subset* $\mathcal{S}_t$ of m arms. This approach is a case of the sparse Gaussian process approximation (Quinero-Candela et al., 2007), or equivalently, linear bandits constrained to a subspace (Kuzborskij et al., 2019).

Approximated GP-UCB scores. Consider a subset of arm $\mathcal{S}_t \triangleq \{\mathbf{x}_i\}_{i=1}^m$ and let $\mathbf{X}_{\mathcal{S}_t} \in \mathbb{R}^{m \times d}$ be the matrix with all arms in $\mathcal{S}_t$ as rows. Let $\mathbf{K}_{\mathcal{S}_t} \in \mathbb{R}^{m \times m}$ be the matrix constructed by evaluating the covariance k between any two pairs of arms in $\mathcal{S}_t$ and $\mathbf{k}_{\mathcal{S}_t}(\mathbf{x}) \triangleq [k(\mathbf{x}_1, \mathbf{x}), \dots, k(\mathbf{x}_m, \mathbf{x})]^\top$. The Nyström embedding $\mathbf{z}_t(\cdot)$ associated with subset $\mathcal{S}_t$ is defined as the mapping²

$$\mathbf{z}_t(\cdot) \triangleq \left(\mathbf{K}_{\mathcal{S}_t}^{1/2}\right)^+ \mathbf{k}_{\mathcal{S}_t}(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^m,$$

where $(\cdot)^+$ indicates the pseudo-inverse. We denote with $\mathbf{Z}_t(\mathbf{X}_t) \triangleq [\mathbf{z}_t(\mathbf{x}_1), \dots, \mathbf{z}_t(\mathbf{x}_t)]^\top \in \mathbb{R}^{t \times m}$ the associated matrix of points and we define $\mathbf{V}_t \triangleq \mathbf{Z}_t(\mathbf{X}_t)^\top \mathbf{Z}_t(\mathbf{X}_t) + \lambda \mathbf{I}$. Then, we approximate the posterior mean, variance, and UCB for the value of the function at $\mathbf{x}_i$ as

$$\begin{aligned} \tilde{\mu}_t(\mathbf{x}_i) &\triangleq \mathbf{z}_t(\mathbf{x}_i)^\top \mathbf{V}_t^{-1} \mathbf{Z}_t(\mathbf{X}_t)^\top \mathbf{y}_t, \\ \tilde{\sigma}_t^2(\mathbf{x}_i) &\triangleq \frac{1}{\lambda} \left(k(\mathbf{x}_i, \mathbf{x}_i) - \mathbf{z}_t(\mathbf{x}_i)^\top \mathbf{Z}_t(\mathbf{X}_t)^\top \mathbf{Z}_t(\mathbf{X}_t) \mathbf{V}_t^{-1} \mathbf{z}_t(\mathbf{x}_i) \right), \\ \tilde{u}_t(\mathbf{x}_i) &\triangleq \tilde{\mu}_t(\mathbf{x}_i) + \tilde{\beta}_t \tilde{\sigma}_t(\mathbf{x}_i), \end{aligned} \tag{4}$$

where $\tilde{\beta}_t$ is appropriately tuned to achieve small regret in the theoretical analysis of Section 4. Finally, at each time step t , BKB selects arm $\tilde{\mathbf{x}}_{t+1} = \arg \max_{\mathbf{x}_i \in \mathcal{A}} \tilde{u}_t(\mathbf{x}_i)$.

Notice that in general, $\tilde{\mu}_t$ and $\tilde{\sigma}_t$ do *not* correspond to any GP posterior. In fact, if we were simply replacing the $k(\mathbf{x}_i, \mathbf{x}_i)$ in the expression of $\tilde{\sigma}_t^2(\mathbf{x}_i)$ by its value in the Nyström embedding, i.e., $\mathbf{z}_t(\mathbf{x}_i)^\top \mathbf{z}_t(\mathbf{x}_i)$, then we would recover a sparse GP approximation known as the *subset of regressors*. Using $\mathbf{z}_t(\mathbf{x}_i)^\top \mathbf{z}_t(\mathbf{x}_i)$ is known to cause *variance starvation*, as it can severely underestimate the variance of a test point $\mathbf{x}_i$ when it is far from the points in $\mathcal{S}_t$. Our formulation of $\tilde{\sigma}_t$ is known in Bayesian world as the *deterministic training conditional* (DTC), where it is used as a heuristic to

2. Recall that in the exact version, $\mathbf{k}_t(\mathbf{x})$ can be seen as an embedding of any arm $\mathbf{x}$ into the space induced by all the t arms selected so far, i.e. using all selected points as inducing points.

prevent variance starvation. However, DTC does *not* correspond to a GP since it violates consistency (Quinero-Candela et al., 2007). In this work, we justify this approach rigorously and analyze it.

Choosing the inducing points. A critical for the low complexity of BKB while still controlling the regret is to carefully choose the inducing points to include in the subset $\mathcal{S}_t$. As the complexity of computing $\tilde{u}_t$ scales with the size m of $\mathcal{S}_t$, a smaller set gives a faster algorithm. Conversely, the difference between $\tilde{\mu}_t$ and $\tilde{\sigma}_t$ and their exact counterparts depends on the accuracy of the embedding $\mathbf{z}_t$, which increases with the size of $\mathcal{S}_t$. Moreover, even for a fixed m , the quality of the embedding greatly depends on *which* inducing points are included. For instance, selecting the same arm as inducing point twice, or two co-linear arms, does not improve accuracy as the embedding space does not change. Finally, we need to take into account two important aspects of sequential optimization when choosing $\mathcal{S}_t$. First, we need to focus our approximation more on regions of $\mathcal{A}$ that are relevant to the objective (i.e., high-reward arms). Second, as these regions change over time, we need to keep adapting the composition and size of $\mathcal{S}_t$. With these objectives in mind, we choose to construct $\mathcal{S}_t$ by randomly subsampling only out of the set of arms $\tilde{\mathbf{X}}_t$ evaluated so far. In particular, arms are selected for inclusion in $\mathcal{S}_t$ with a probability proportional to their posterior variance σ_t at step t . We report the selection procedure in Algorithm 1 with the complete BKB algorithm.

We initialize $\mathcal{S}_1 \triangleq \{\tilde{\mathbf{x}}_1\}$ by selecting an arm uniformly at random. At each step t , after selecting $\tilde{\mathbf{x}}_{t+1}$, we must regenerate $\mathcal{S}_t$ to reflect the changes in $\tilde{\mathbf{X}}_{t+1}$. Ideally, we would sample each arm in $\tilde{\mathbf{X}}_{t+1}$ proportionally to σ_{t+1}^2 , but this would be too computationally expensive. Therefore, we first approximate σ_{t+1}^2 with σ_t^2 . This is equivalent to ignoring the last arm and does not significantly impact the accuracy. We can then replace σ_t^2 with $\tilde{\sigma}_t^2$ that was already efficiently computed when constructing Equation 4. Finally, given a parameter $\bar{q} \geq 1$, we set our approximate inclusion probability as $\tilde{p}_{t+1,i} \triangleq \bar{q} \tilde{\sigma}_t^2(\tilde{\mathbf{x}}_i)$. The $\bar{q}$ parameter is used to increase the inclusion probability in order to boost the overall success probability of the approximation procedure at the expense of a small increase in the size of $\mathcal{S}_{t+1}$. Given $\tilde{p}_{t+1,i}$, we start from an empty $\mathcal{S}_{t+1}$ and iterate over all $\tilde{\mathbf{x}}_i$ for $i \in [t+1]$ drawing $q_{t+1,i}$ from a Bernoulli distribution with probability $\tilde{p}_{t+1,i}$. If $q_{t+1,i} = 1$, $\tilde{\mathbf{x}}_i$ is included in $\mathcal{S}_{t+1}$.

Notice that while constructing $\mathcal{S}_t$ based on σ_t^2 is a common heuristic for sparse GPs, it has not been yet rigorously justified. In the next section, we show that this approach is equivalent to λ -ridge leverage score (RLS) sampling (Alaoui and Mahoney, 2015), a well studied tool in randomized linear algebra. We leverage the results from this field to prove both accuracy and efficiency guarantees.

3.2. Complexity analysis

Let $m_t \triangleq |\mathcal{S}_t|$ be the size of the set $\mathcal{S}_t$ at step t . At each step, we first compute the embedding $\mathbf{z}_t(\mathbf{x}_i)$ of all arms in $\mathcal{O}(Am_t^2 + m_t^3)$ time, which corresponds to one inversion of $\mathbf{K}_{\mathcal{S}_t}^{1/2}$ and the matrix-vector product specific to each arm. We then rebuild the matrix $\mathbf{V}_t$ from scratch using all

Algorithm 1: BKB

Data: Arm set $\mathcal{A}$, q , $\{\beta_t\}_{t=1}^T$

Result: Arm choices $\mathcal{D}_T \leftarrow \{(\tilde{\mathbf{x}}_t, y_t)\}$

Select uniformly at random $\mathbf{x}_1$ and observe y_1 ;

Initialize $\mathcal{S}_1 \leftarrow \{\mathbf{x}_1\}$;

for $t = \{1, \dots, T-1\}$ **do**

 Compute $\tilde{\mu}_t(\mathbf{x}_i)$ and $\tilde{\sigma}_t^2(\mathbf{x}_i)$ for all $\mathbf{x}_i \in \mathcal{A}$;

 Select $\tilde{\mathbf{x}}_{t+1} \leftarrow \arg \max_{\mathbf{x}_i \in \mathcal{A}} \tilde{u}_t(\mathbf{x}_i)$ (Eq. 4);

for $i = \{1, \dots, t+1\}$ **do**

 Set $\tilde{p}_{t+1,i} \leftarrow \bar{q} \cdot \tilde{\sigma}_t^2(\tilde{\mathbf{x}}_i)$;

 Draw $q_{t+1,i} \sim \text{Bernoulli}(\tilde{p}_{t+1,i})$;

If $q_{t+1,i} = 1$ **then** include $\tilde{\mathbf{x}}_i$ in $\mathcal{S}_{t+1}$;

end

end

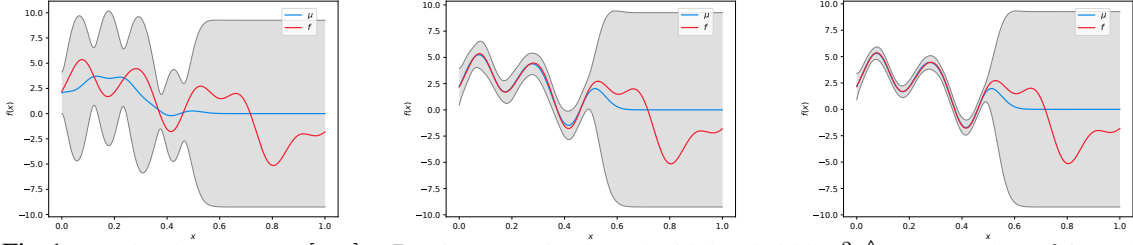


Fig. 1: We simulate a GP on $[0, 1] \in \mathbb{R}$ using Gaussian kernel with bandwidth $\sigma^2 \triangleq 100$. We draw f from the GP and give to BKB $t \in \{6, 63, 215\}$ evaluations sampled uniformly in $[0, 0.5]$. We plot f and $\tilde{\mu}_t \pm 3\tilde{\sigma}_t$.

the arms observed so far. In general, it is sufficient to record the counters of the arms pulled so far, rather than the full list of arms, so that $\mathbf{V}_t$ can be constructed in $\mathcal{O}(\min\{t, A\}m_t^2)$ time. Then, the inverse $\mathbf{V}_t^{-1}$ is computed in $\mathcal{O}(m_t^3)$ time. We can now efficiently compute $\tilde{\mu}_t$, $\tilde{\sigma}_t$, and $\tilde{u}_t$ for all arms in $\mathcal{O}(Am_t^2)$ time reusing the embeddings and $\mathbf{V}_t^{-1}$. Finally, computing all $q_{t+1,i}$ s and $\mathcal{S}_{t+1}$ takes $\mathcal{O}(\min\{t+1, A\})$ time using the estimated variances $\tilde{\sigma}_t^2$. As a result, the per-step complexity is of order $\mathcal{O}((A + \min\{t, A\})m_T^2)$.³ Space-wise, we only need to store the embedded arms and $\mathbf{V}_t$ matrix, which takes at most $\mathcal{O}(Am_T)$ space.

The size of $\mathcal{S}_T$. The size m_t of $\mathcal{S}_t$ can be expressed using the $q_{t,i}$ r.v. as the sum $m_t \triangleq \sum_{i=1}^t q_{t,i}$. In order to provide a bound on the total number of inducing points, which directly determines the computational complexity of BKB, we go through three major steps.

The first is to show that w.h.p., m_t is close to the sum $\sum_{i=1}^t \tilde{p}_{t,i} = \sum_{i=1}^t \bar{q}\tilde{\sigma}_t^2(\tilde{\mathbf{x}}_i)$, i.e., close to the sum of the probabilities we used to sample each $q_{t,i}$. However, the different $q_{t,i}$ are *not independent* and each $\tilde{p}_{t,i}$ is itself a r.v. Nonetheless all $q_{t,i}$ are conditionally independent given the previous $t-1$ steps, and this is sufficient to obtain the result.

The second and a more complex step is to guarantee that the random sum $\sum_{i=1}^t \tilde{\sigma}_t^2(\tilde{\mathbf{x}}_i)$ is close to $\sum_{i=1}^t \sigma_t^2(\tilde{\mathbf{x}}_i)$ and, at a lower level, that each individual estimate $\tilde{\sigma}_t^2(\cdot)$ is close to $\sigma_t^2(\cdot)$. To achieve this we exploit the connection between ridge leverage scores and posterior variance σ_t^2 . In particular, we show that the variance estimator $\tilde{\sigma}_t^2(\cdot)$ used by BKB is a variation of the RLS estimator of [Calandriello et al. \(2017a\)](#).

The first two steps lead to $m_t \approx \sum_{i=1}^t \sigma_i^2(\tilde{\mathbf{x}}_i)$, for which we need to derive a more explicit bound. In the GP analyses, this quantity is bounded using the maximal information gain γ_T after T rounds. For this, let $\mathbf{X}_A \in \mathbb{R}^{A \times d}$ be the matrix with all arms as rows, $\mathcal{D}$ a subset of these rows, potentially with duplicates, and $\mathbf{K}_{\mathcal{D}}$ the associated kernel matrix. Then, [Srinivas et al. \(2010\)](#) define

$$\gamma_T \triangleq \max_{\mathcal{D} \subset A: |\mathcal{D}|=T} \frac{1}{2} \log \det(\mathbf{K}_{\mathcal{D}}/\lambda + \mathbf{I}),$$

and show that $\sum_{i=1}^t \sigma_i^2(\tilde{\mathbf{x}}_i) \leq \gamma_t$, and that γ_T itself can be bounded for specific A and kernel functions, e.g., $\gamma_T \leq \mathcal{O}(\log(T)^{d+1})$ for Gaussian kernels. Using the equivalence between RLS and posterior variance σ_t^2 , we can also relate the posterior variance $\sigma_t^2(\tilde{\mathbf{x}}_i)$ of the evaluated arms to the GP's *effective dimension* d_{eff} or degrees of freedom

$$d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) \triangleq \sum_{i=1}^t \sigma_t^2(\tilde{\mathbf{x}}_i) = \text{Tr}(\mathbf{K}_T(\mathbf{K}_T + \lambda\mathbf{I})^{-1}), \quad (5)$$

using the following inequality by [Calandriello et al. \(2017b\)](#),

$$\log \det(\mathbf{K}_T/\lambda + \mathbf{I}) \leq \text{Tr}(\mathbf{K}_T(\mathbf{K}_T + \lambda\mathbf{I})^{-1}) \left(1 + \log\left(\frac{\|\mathbf{K}_T\|}{\lambda} + 1\right)\right). \quad (6)$$

3. Notice that $m_t \leq \min\{t, a\}$ and thus the complexity term $\mathcal{O}(m_t^3)$ is absorbed by the other terms.

We use both RLS and d_{eff} to describe BKB's selection. We now give the main result of this section.

Theorem 1 *For a desired $0 < \varepsilon < 1$, $0 < \delta < 1$, let $\alpha \triangleq (1 + \varepsilon)/(1 - \varepsilon)$. If we run BKB with $\bar{q} \geq 6\alpha \log(4T/\delta)/\varepsilon^2$, then with probability $1 - \delta$, for all $t \in [T]$ and for all $\mathbf{x} \in \mathcal{A}$, we have*

$$\sigma_t^2(\mathbf{x})/\alpha \leq \tilde{\sigma}_t^2(\mathbf{x}) \leq \alpha \sigma_t^2(\mathbf{x}) \quad \text{and} \quad |S_t| \leq 3(1 + \kappa^2/\lambda)\alpha \bar{q} d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_t).$$

Computational complexity. We already showed that BKB's implementation with Nyström embedding requires $\mathcal{O}(T(A + \min\{t, A\})m_T^3)$ time and $\mathcal{O}(Am_T)$ space. Combining this with Theorem 1 and the bound $m_T \leq \tilde{\mathcal{O}}(d_{\text{eff}})$, we obtain a $\tilde{\mathcal{O}}(TAd_{\text{eff}}^2 + \min\{t, A\}d_{\text{eff}}^3)$ time complexity. Whenever $d_{\text{eff}} \ll T$ and $T \ll A$, this is essentially a quadratic $\mathcal{O}(T^2)$ runtime, a large improvement over the quartic $\mathcal{O}(T^4) \leq \mathcal{O}(T^3A)$ runtime of GP-UCB.

Tuning $\bar{q}$. Note that although $\bar{q}$ must satisfy the condition of Theorem 1 for the result to hold, it is quite robust to uncertainty on the desired horizon T . In particular, the bound holds for *any* $\varepsilon > 0$, and even if we continue updating $\mathcal{S}_T$ after the T -th step, the bound still holds by implicitly increasing the parameter ε . Alternatively, after the T -th iteration the user can suspend the algorithm, increase $\bar{q}$ to suit the new desired horizon, and rerun only the subset selection on the arms selected so far.

Avoiding variance starvation. Another important consequence of Theorem 1 is that BKB's variance estimate is always close to the exact one up to a small constant factor. To the best of our knowledge, it makes BKB the first efficient and general GP algorithm that provably avoids variance starvation, which can be caused by two sources of error. The first source is the degeneracy, i.e., low-rankness of the GP approximation which causes the estimate to grow over-confident when the number of observed points grows and exceeds the degrees of freedom of the GP. BKB *adaptively chooses its degrees of freedom* as the size of $\mathcal{S}_t$ scales with the effective dimension. The second source of error arises when a point is far away from $\mathcal{S}_t$. Our use of a DTC variance estimator avoids under-estimation before we update the subset $\mathcal{S}_t$. Afterward, we can use guarantees on the quality of $\mathcal{S}_t$ to guarantee that we do not over-estimate the variance too much, exploiting a similar approach used to guarantee accuracy in RLS estimation. Both problems, and BKB's accuracy, are highlighted in Figure 1 using a benchmark experiment proposed by Wang et al. (2018).

Incremental dictionary update. At each step t , BKB recomputes the dictionary $\mathcal{S}_{t+1}$ from scratch by sampling each of the arms pulled so far with a suitable probability $\tilde{p}_{t+1,i}$. A more efficient variant would be to build $\mathcal{S}_{t+1}$ by adding the new point $\mathbf{x}_{t+1}$ with probability $\tilde{p}_{t+1,t+1}$ and including the points in $\mathcal{S}_t$ with probability $\tilde{p}_{t+1,i}/\tilde{p}_{t,i}$. This strategy is used in the streaming setting to avoid storing all points observed so far and incrementally update the dictionary (see Calandriello et al., 2017a). Nonetheless, the stream of points, although arbitrary, is assumed to be generated *independently* from the dictionary itself. On the other hand, in our bandit setting, the points $\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots$ are actually chosen by the learner depending on the dictionaries built over time, thus building a strong dependency between the stream of points and the dictionary itself. How to analyze such dependency and whether the accuracy of the inducing points is preserved in this case remains as an open question.

4. Regret Analysis

We are now ready to present the second main contribution of this paper, a bound on the regret achieved by BKB. To prove our result we additionally assume that the reward function f has a bounded norm, i.e., $\|f\|_{\mathcal{H}}^2 \triangleq \langle f, f \rangle < \infty$. We use an upper-bound $\|f\|_{\mathcal{H}} \leq F$ to properly tune β_t to the range of the rewards. If F is not known in advance, standard guess-and-double techniques apply.

Theorem 2 Assume $\|f\|_{\mathcal{H}} \leq F < \infty$. For any desired $0 < \varepsilon < 1$, $0 < \delta < 1$, $0 < \lambda$, let $\alpha \triangleq (1 + \varepsilon)/(1 - \varepsilon)$ and $\bar{q} \geq 6\alpha \log(4T/\delta)/\varepsilon^2$. If we run BKB with

$$\tilde{\beta}_t \triangleq 2\xi \sqrt{\alpha \log(\kappa^2 t) \left(\sum_{s=1}^t \tilde{\sigma}_t^2(\tilde{\mathbf{x}}_s) \right) + \log(1/\delta) + \left(1 + \frac{1}{\sqrt{1-\varepsilon}}\right) \sqrt{\lambda} F},$$

then, with probability of at least $1 - \delta$, the regret R_T of BKB is bounded as

$$R_T \leq 2(2\alpha)^{3/2} \sqrt{T} \left(\xi d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) \log(\kappa^2 T) + \sqrt{\lambda F^2 d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) \log(\kappa^2 T)} + \xi \log(1/\delta) \right).$$

Theorem 2 shows that BKB achieves exactly the same regret as (exact) GP-UCB up to small α constant and $\log(\kappa^2 T)$ multiplicative factor.⁴ For instance, setting $\varepsilon = 1/2$ results in a bound only $3 \log T$ times larger than the of GP-UCB. At the same time, the choice $\varepsilon = 1/2$ only accounts for a constant factor 12 in the per-step computational complexity, which is still dramatically reduced from $t^2 A$ to $d_{\text{eff}}^2 A$. Note also that even if we send ε to 0, in the worst case we will include all arms selected so far, i.e., $\mathcal{S}_t = \{\tilde{\mathbf{X}}_t\}$. Therefore, even in this case BKB's runtime does not grow unbounded, but BKB transforms back into exact GP-UCB. Moreover, we show that $d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) \leq \log \det(\mathbf{K}_T/\lambda + \mathbf{I})$, as in Proposition 5 in the appendix, so any bound on $\log \det(\mathbf{K}_T/\lambda + \mathbf{I})$ available for GP-UCB applies directly to BKB. This means that up to an extra $\log T$ factor, we match GP-UCB's $\tilde{\mathcal{O}}(\log(T)^{2d})$ rate for the Gaussian kernel, $\tilde{\mathcal{O}}(T^{\frac{1}{2} \frac{2\nu+3d^2}{2\nu+d^2}})$ rate for the Matérn kernel, and $\tilde{\mathcal{O}}(d\sqrt{T})$ for the linear kernel. While these bounds are not minimax optimal, they closely follow the lower bounds derived in Scarlett et al. (2017). On the other hand, in the case of linear kernel (i.e., the linear bandits) we nearly match the lower bound of Dani et al. (2008).

Another interesting aspect of BKB is that computing the trade-off parameter $\tilde{\beta}_t$ can be done efficiently. Previous methods bounded this quantity with a loose (deterministic) upper bound, e.g., $\mathcal{O}(\log(T)^d)$ for Gaussian kernels, to avoid the large cost of computing $\log \det(\mathbf{K}_T/\lambda + \mathbf{I})$. In our $\tilde{\beta}_t$, we bound the $\log \det$ by d_{eff} , which is then bounded by $\sum_{s=1}^t \tilde{\sigma}_t^2(\mathbf{x}_s)$, see Thm. 1, where all $\tilde{\sigma}_t^2$ s are already efficiently computed at each step. While this is up to $\log t$ larger than the exact $\log \det$, it is *data adaptive* and much smaller than the known worst case upper bounds.

It is crucial, that our regret guarantee is achieved without requiring an *increasing accuracy* in our approximation. One would expect that to obtain a sublinear regret the error induced by the approximation should decrease as $1/T$. Instead, in BKB, the constants ε and λ that govern the accuracy level are fixed and thus it is not possible to guarantee that $\tilde{\mu}_t$ will ever get close to μ_t everywhere. Adaptivity is the key: we can afford the same approximation level at every step because accuracy is actually increased only on a specific part of the arm set. For example, if a suboptimal arm is selected too often due to bad approximation, it will be eventually included in $\mathcal{S}_t$. After the inclusion, the approximation accuracy in the region of the suboptimal arm increases, and it would not be selected anymore. As the set of inducing points is updated *fast enough*, the impact of inaccurate approximations is limited over time, thus preventing large regret to accumulate. Note that this is a significant divergence from existing results. In particular approximation bounds that are uniformly accurate for all $\mathbf{x}_i \in \mathcal{A}$, such as those obtained with quadrature FF (Mutný and Krause, 2018), rely on packing arguments. Due to the nature of packing, this usually causes the runtime or regret to scale

4. Here we derive a *frequentist* regret bound and thus we compare with the result of Chowdhury and Gopalan (2017) rather than the original *Bayesian* analysis of Srinivas et al. (2010).

exponentially with the input dimension d , and requires kernel k to have a specific structure, e.g., to be stationary. Our new analysis avoids both of these problems.

Finally, we point out that the adaptivity of BKB allows drawing an interesting connection between learning and computational complexity. In fact, both the regret and the computation of BKB scale with the log-determinant and the effective dimension of $\mathbf{K}_T$, which is related to the effective dimension of the sequence of arms selected over time. As a result, if the problem is difficult as a learning problem, then BKB automatically adapts the set $\mathcal{S}_t$ by including many more inducing points to guarantee the level of accuracy needed to solve the problem. Conversely, if the problem is simple, then BKB greatly reduces the size of $\mathcal{S}_t$ and achieves the derived level of accuracy.

4.1. Proof sketch

We build on the GP-UCB analysis of [Chowdhury and Gopalan \(2017\)](#). Their analysis relies on a confidence interval formulation of GP-UCB that is more conveniently expressed using an explicit feature-based representation of the GP. For any GP with covariance k , there is a corresponding RKHS $\mathcal{H}$ with k as its kernel function. Furthermore, any kernel function k is associated to a non-linear feature map $\phi(\cdot) : \mathbb{R}^d \rightarrow \mathcal{H}$ such that $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x}')^\top \phi(\mathbf{x})$. As a result, any reward function $f \in \mathcal{H}$ can be written as $f(\mathbf{x}) = \phi(\mathbf{x})^\top \mathbf{w}_*$, where $\mathbf{w}_* \in \mathcal{H}$.

Confidence-interval view of GP-UCB. Let $\Phi(\mathbf{X}_t) \triangleq [\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_t)]^\top$ be the matrix $\mathbf{X}_t$ after the application of $\phi(\cdot)$ to each row. We can then define the regularized design matrix as $\mathbf{A}_t \triangleq \Phi(\mathbf{X}_t)^\top \Phi(\mathbf{X}_t) + \lambda \mathbf{I}$, and then compute regularized least-squares estimate as

$$\hat{\mathbf{w}}_t \triangleq \arg \min_{\mathbf{w} \in \mathcal{H}} \sum_{i=1}^t (y_i - \phi(\mathbf{x}_i)^\top \mathbf{w})^2 + \lambda \|\mathbf{w}\|_2^2 = \mathbf{A}_t^{-1} \Phi(\mathbf{X}_t)^\top \mathbf{y}_t.$$

We define the *confidence interval* C_t as the ellipsoid induced by $\mathbf{A}_t$ with center $\hat{\mathbf{w}}_t$ and radius β_t

$$C_t \triangleq \{\mathbf{w} : \|\mathbf{w} - \hat{\mathbf{w}}_t\|_{\mathbf{A}_t} \leq \beta_t\}, \quad \beta_t \triangleq \lambda^{1/2} F + R \sqrt{2(\log \det(\mathbf{A}_t/\lambda) + \log(1/\delta))}, \quad (7)$$

where the radius β_t is such that $\mathbf{w}_* \in C_t$ w.h.p. ([Chowdhury and Gopalan, 2017](#)). Finally, using Lagrange multipliers we reformulate the GP-UCB scores as

$$u_t(\mathbf{x}_i) = \max_{\mathbf{w} \in C_t} \phi(\mathbf{x}_i)^\top \mathbf{w} = \overbrace{\phi(\mathbf{x}_i)^\top \hat{\mathbf{w}}_t}^{\mu_t(\mathbf{x}_i)} + \beta_t \overbrace{\sqrt{\phi(\mathbf{x}_i)^\top \mathbf{A}_t^{-1} \phi(\mathbf{x}_i)}}^{\sigma_t(\mathbf{x}_i)}. \quad (8)$$

Approximating the confidence ellipsoid. Consider subset of arm $\mathcal{S}_t = \{\mathbf{x}_i\}_{i=1}^m$ chosen by BKB at each step and denote by $\mathbf{X}_{\mathcal{S}_t} \in \mathbb{R}^{m \times d}$ the matrix with all arms in $\mathcal{S}_t$ as rows. Let $\tilde{\mathcal{H}}_t \triangleq \text{Im}(\Phi(\mathbf{X}_{\mathcal{S}_t}))$ be the smaller m -rank RKHS spanned by $\Phi(\mathbf{X}_{\mathcal{S}_t})$; and by $\mathbf{P}_t$ the symmetric orthogonal projection operator on $\tilde{\mathcal{H}}_t$. We then define an *approximate* feature map $\tilde{\phi}_t(\cdot) \triangleq \mathbf{P}_t \phi(\cdot) : \mathbb{R}^d \rightarrow \tilde{\mathcal{H}}_t$ and associated approximations of $\mathbf{A}_t$ and $\hat{\mathbf{w}}_t$ as

$$\tilde{\mathbf{A}}_t \triangleq \tilde{\Phi}_t(\mathbf{X}_t)^\top \tilde{\Phi}_t(\mathbf{X}_t) + \lambda \mathbf{I}, \quad (9)$$

$$\tilde{\mathbf{w}}_t \triangleq \arg \min_{\mathbf{w} \in \mathcal{H}} \sum_{i=1}^t (y_i - \tilde{\phi}_t(\mathbf{x}_i)^\top \mathbf{w})^2 + \lambda \|\mathbf{w}\|_2^2 = \tilde{\mathbf{A}}_t^{-1} \tilde{\Phi}_t(\mathbf{X}_t)^\top \mathbf{y}_t. \quad (10)$$

This leads to an approximate confidence ellipsoid $\tilde{C}_t \triangleq \{\mathbf{w} : \|\mathbf{w} - \tilde{\mathbf{w}}_t\|_{\tilde{\mathbf{A}}_t} \leq \tilde{\beta}_t\}$. A subtle element in these definitions is that while $\tilde{\Phi}_t(\mathbf{X}_t)^\top \tilde{\Phi}_t(\mathbf{X}_t)$ and $\tilde{\mathbf{w}}_t$ are now *restricted* to $\tilde{\mathcal{H}}_t$, the identity operator $\lambda \mathbf{I}$ in the regularization of $\tilde{\mathbf{A}}_t$ still *acts over the whole* $\mathcal{H}$, and therefore $\tilde{\mathbf{A}}_t$ does not belong

to $\tilde{\mathcal{H}}_t$ and remains full-rank and invertible. This immediately leads to the usage of $k(\mathbf{x}_i, \mathbf{x}_i)$ in the definition of $\tilde{\sigma}$ in Eq. 4, instead of its approximate version using the Nyström embedding.

Bounding the regret. To find an appropriate $\tilde{\beta}_t$, we follow an approach similar to the of Abbasi-Yadkori et al. (2011). Exploiting the relationship $y_t = \tilde{\phi}(\tilde{\mathbf{x}}_t)^\top \mathbf{w}_* + \eta_t$, we bound

$$\|\mathbf{w}_* - \tilde{\mathbf{w}}_t\|_{\tilde{\mathbf{A}}_t}^2 \leq \overbrace{\lambda^{1/2} \|\mathbf{w}_*\|}^{(a)} + \overbrace{\|\tilde{\Phi}_t(\mathbf{X}_t) \boldsymbol{\eta}_t\|_{\tilde{\mathbf{A}}_t^{-1}}}^{(b)} + \overbrace{\|\Phi(\mathbf{X}_t)^\top\|_{\mathbf{I}-\mathbf{P}_t} \cdot \|\mathbf{w}_*\|}^{(c)}.$$

Both (a) and (b) are present in GP-UCB's and OFUL's analysis. First, term (a) is due to the bias introduced in the least-square estimator $\tilde{\mathbf{w}}_t$ by the regularization λ . Then, term (b) is due to the noisy rewards. Note that the same term (b) appears in GP-UCB's analysis as $\|\Phi(\mathbf{X}_t) \boldsymbol{\eta}_t\|_{\mathbf{A}_t^{-1}}$ and it is bounded by $\log \det(\mathbf{A}_t/\lambda)$ using self-normalizing concentration inequalities (Chowdhury and Gopalan, 2017). However, our $\|\tilde{\Phi}_t(\mathbf{X}_t) \boldsymbol{\eta}_t\|_{\tilde{\mathbf{A}}_t^{-1}}$ is a more complex object, since the projection $\mathbf{P}_t$ contained in $\tilde{\Phi}_t(\mathbf{X}_t) \triangleq \mathbf{P}_t \Phi(\mathbf{X}_t)$ depends on the whole process up to time t , and therefore $\tilde{\Phi}_t(\mathbf{X}_t)$ also depends on the whole process, losing its martingale structure. To avoid this, we use Sylvester's identity and the projection operator $\mathbf{P}_t$ to bound

$$\log \det(\tilde{\mathbf{A}}_t/\lambda) = \log \det\left(\frac{\Phi(\mathbf{X}_t) \mathbf{P}_t \Phi(\mathbf{X}_t)^\top}{\lambda} + \mathbf{I}\right) \leq \log \det\left(\frac{\Phi(\mathbf{X}_t) \Phi(\mathbf{X}_t)^\top}{\lambda} + \mathbf{I}\right) = \log \det(\mathbf{A}_t/\lambda).$$

In other words, restricting the problem to $\tilde{\mathcal{H}}_t$ acts as a regularization and reduces the variance of the martingale. Unfortunately, $\log \det(\mathbf{A}_t/\lambda)$ is too expensive to compute, so we first bound it with $d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_t) \log(\kappa^2 t)$, and then we bound $d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_t) \leq \alpha \sum_{s=1}^t \tilde{\sigma}_t^2(\mathbf{x}_s)$, Theorem 1, which can be computed efficiently. Finally, a new bias term (c) appears. Combining Theorem 1 with the results of Calandriello and Rosasco (2018) for projection $\mathbf{P}_t$ obtained using RLSs sampling, we show that

$$\mathbf{I} - \mathbf{P} \preceq \lambda \mathbf{A}_t^{-1} / (1 - \varepsilon).$$

The combination of (a), (b), and (c) leads to the definition of $\tilde{\beta}_t$ and the final regret bound as $R_T \leq \sqrt{\tilde{\beta}_T} \sqrt{\sum_{t=1}^T \phi(\mathbf{x}_t)^\top \tilde{\mathbf{A}}_t^{-1} \phi(\mathbf{x}_t)}$. To conclude the proof, we bound $\sum_{t=1}^T \phi(\mathbf{x}_t)^\top \tilde{\mathbf{A}}_t^{-1} \phi(\mathbf{x}_t)$ with the following corollary of Theorem 1.

Corollary 3 *Under the same conditions as Theorem 2, for all $t \in T$, we have $\mathbf{A}_t/\alpha \preceq \tilde{\mathbf{A}}_t \preceq \alpha \mathbf{A}_t$.*

Remarks. The novel bound $\|\Phi(\mathbf{X}_t)^\top\|_{\mathbf{I}-\mathbf{P}_t} \leq \frac{\lambda}{1-\varepsilon} \|\Phi(\mathbf{X}_t)^\top\|_{\mathbf{A}_t^{-1}}$ has a crucial role in controlling the bias due to the projection $\mathbf{P}_t$. Note that the second term measures the error with the same metric $\mathbf{A}_t^{-1}$ used by the variance martingale. In other words, the bias introduced by BKB's approximation can be seen as a *self-normalizing* bias. It is larger along directions that have been sampled less frequently, and smaller along directions correlated with arms selected often (e.g., the optimal arm). Our analysis bears some similarity with the one recently and independently developed by Kuzborskij et al. (2019). Nonetheless, our proof improves their result along two dimensions. First, we consider the more general (and challenging) GP optimization setting. Second, *we do not fix* the rank of our approximation in advance. While their analysis also exploits a self-normalized bias argument, this applies only to the k largest components. If the problem has an effective dimension larger than k , their radius and regret is linear. In BKB we use our adaptive sampling scheme to include all necessary directions and to achieve the same regret rate as exact GP-UCB.

5. Discussion

As the prior work in Bayesian optimization is vast, we do not compare to alternative GP acquisition functions, such as GP-EI or GP-PI, and only focus on approximation techniques with theoretical

guarantees. Similarly, we exclude scalable variational inference based methods, even when their approximate posterior is provably accurate such as pF-DTC (Huggins et al., 2019), since they only provide guarantees for GP regression and not for the more difficult optimization setting. We also do not discuss SUPKERNELUCB (Valko et al., 2013), which has a tighter analysis than GP-UCB, since the algorithm does not work well in practice.

Infinite arm sets. Looking at the proof of Theorem 1, the guarantees on $\tilde{u}_t$ hold for any $\mathcal{H}$, and in Theorem 2, we only require that the maximum $\tilde{\mathbf{x}}_{t+1} \triangleq \arg \max_{\mathbf{x} \in \mathcal{A}} \max_{\mathbf{w} \in \tilde{\mathcal{C}}_t} \phi(\mathbf{x})^\top \mathbf{w}$ is returned. Therefore, the accuracy and regret guarantees also hold also for an infinite set of arms $\mathcal{A}$. However, the search over $\mathcal{A}$ can be difficult. In the general case, maximization of a GP posterior is an NP-hard problem, with algorithms that often scale exponentially with the input dimension d and are not practical. We treated the easier case of finite sets, where the enumeration is sufficient. Note that this automatically introduces an $\Omega(A)$ runtime dependency, which could be removed if the user provides an efficient method to solve the maximization problem on a specific infinite set $\mathcal{A}$. As an example, Mutný and Krause (2018) prove that a GP posterior approximated using QFF can be optimized efficiently in low dimensions and we expect similar results hold for BKB and low *effective* dimension. Finally, note that recomputing a new set $\mathcal{S}_t$ still requires $\min\{A, t\}d_{\text{eff}}^2$ at each step. This is a bottleneck in BKB independent from the arm selection, and must be addressed in future.

Linear bandit with matrix sketching. Our analysis is related to the ones of CBRAP (Yu et al., 2017) and SOFUL (Kuzborskij et al., 2019). CBRAP uses Gaussian projections to embed all arms in a lower dimensional space. Unfortunately, their approach must either use an embedded space at least $\Omega(T)$ large, which can making it slower than the exact OFUL, or it incurs regret w.h.p. Another approach for linear bandits by Kuzborskij et al. (2019), SOFUL, uses frequent directions (Ghashami et al., 2016), a method similar to incremental PCA, to embed the arms into $\mathbb{R}^m$, where m is *fixed* in advance. To compare, we distinguish between SOFUL-UCB and SOFUL-TS, a variant based on Thompson sampling. SOFUL-UCB achieves a $\tilde{\mathcal{O}}(TAm^2)$ runtime and $\tilde{\mathcal{O}}((1 + \varepsilon_m)^{3/2}(d + m)\sqrt{T})$ regret, where ε_m is the sum of the $d - m$ smallest eigenvalues of $\mathbf{A}_T$. However, notice that if the tail do not decrease quickly, this algorithm also suffers linear regret and no adaptive way to tune m is known. On the same task BKB achieves a $\tilde{\mathcal{O}}(d\sqrt{T})$ regret, since it adaptively chooses the size of the embedding. Computationally, directly instantiating BKB to use a linear kernel would achieve a $\tilde{\mathcal{O}}(TAm_t^2)$ runtime, matching Kuzborskij et al. (2019)’s. Compared to SOFUL-TS, BKB achieves better regret, but is potentially slower. Since Thompson sampling does not need to compute all confidence intervals, but solves a simpler optimization problem, SOFUL-TS requires only $\tilde{\mathcal{O}}(TAm)$ time against BKB’s $\tilde{\mathcal{O}}(TAm_t^2)$. It is unknown if a variant of BKB can match this complexity.

Approximate GP with RFF. RFF methods are popular to transform GP optimization to finite-dimensional problems and allow for scalability. Unfortunately, GP-UCB with standard RFF is not low-regret as RFF are suffer from variance starvation (Wang et al., 2018) and unfeasibly large RFF embeddings are necessary to prevent it. Recently, Mutný and Krause (2018) proposed an alternative based on QFF, a specialized approach for stationary kernels. They achieve the same regret rate as GP-UCB and BKB, with a near-optimal $\mathcal{O}(TA \log(T)^{d+1})$ runtime and give additional variations based on Thompson sampling and exact posterior maximization. However, quadrature-based approaches apply only to stationary kernels and require to ε -cover $\mathcal{A}$, hence they cannot escape an exponential dependence on d . Conversely, BKB applies to *any* kernel and while not specifically designed for this task it achieves a close $\tilde{\mathcal{O}}(TA \log(T)^{3(d+1)})$ runtime. Moreover, in practice the size of $\mathcal{S}_T$ is less than exponential in d .

Acknowledgements This material is based upon work supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216. L. R. acknowledges the support of the AFOSR projects FA9550-17-1-0390, and BAA-AFRL-AFOSR-2016-0007 (European Office of Aerospace Research and Development), and the EU H2020-MSCA-RISE project NoMADS-DLV-777826. The research was also supported by European CHIST-ERA project DELTA, French Ministry of Higher Education and Research, Nord-Pas-de-Calais Regional Council, Inria and Otto-von-Guericke-Universität Magdeburg associated-team north-European project Allocate, and French National Research Agency project BoB (n.ANR-16-CE23-0003). This research has also benefited from the support of the FMJH Program PGMO and from the support to this program from Criteo.

References

- Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. **Improved algorithms for linear stochastic bandits**. In *Neural Information Processing Systems*, 2011.
- Ahmed El Alaoui and Michael W. Mahoney. **Fast randomized kernel methods with statistical guarantees**. In *Neural Information Processing Systems*, 2015.
- Daniele Calandriello and Lorenzo Rosasco. **Statistical and computational trade-offs in kernel k-means**. In *Neural Information Processing Systems*, 2018.
- Daniele Calandriello, Alessandro Lazaric, and Michal Valko. **Distributed adaptive sampling for kernel matrix approximation**. In *International Conference on Artificial Intelligence and Statistics*, 2017a.
- Daniele Calandriello, Alessandro Lazaric, and Michal Valko. **Second-order kernel online convex optimization with adaptive sketching**. In *International Conference on Machine Learning*, 2017b.
- Sayak Ray Chowdhury and Aditya Gopalan. **On kernelized multi-armed bandits**. In *International Conference on Machine Learning*, 2017.
- Sayak Ray Chowdhury and Aditya Gopalan. **Online learning in kernelized Markov decision processes**. In *International Conference on Artificial Intelligence and Statistics*, 2019.
- Varsha Dani, Thomas P Hayes, and Sham M Kakade. **Stochastic linear optimization under bandit feedback**. In *Conference on Learning Theory*, 2008.
- Mina Ghashami, Edo Liberty, Jeff M Phillips, and David P. Woodruff. **Frequent directions: Simple and deterministic matrix sketching**. *The SIAM Journal of Computing*, pages 1–28, 2016.
- Avishek Ghosh, Sayak Ray Chowdhury, and Aditya Gopalan. **Misspecified linear bandits**. In *AAAI Conference on Artificial Intelligence*, 2017.
- Elad Hazan, Adam Tauman Kalai, Amit Agarwal, and Satyen Kale. **Logarithmic regret algorithms for online convex optimization**. In *Conference on Learning Theory*, 2006.
- Jonathan H. Huggins, Trevor Campbell, Mikołaj Kasprzak, and Tamara Broderick. **Scalable Gaussian process inference with finite-data mean and variance guarantees**. In *International Conference on Artificial Intelligence and Statistics*, 2019.

- Ilja Kuzborskij, Leonardo Cella, and Nicolò Cesa-Bianchi. [Efficient linear bandits through matrix sketching](#). In *International Conference on Artificial Intelligence and Statistics*, 2019.
- Tor Lattimore and Csaba Szepesvári. [Bandit algorithms](#). 2019.
- Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. [A contextual-bandit approach to personalized news article recommendation](#). *International World Wide Web Conference*, 2010.
- Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. [When Gaussian process meets big data: A Review of scalable GPs](#). Technical report, 2018.
- Jonas Mockus. [Global optimization and the Bayesian approach](#). 1989.
- Mojmír Mutný and Andreas Krause. [Efficient high-dimensional Bayesian optimization with additivity and quadrature Fourier features](#). In *Neural Information Processing Systems*, 2018.
- Martin Pelikan. [Hierarchical Bayesian optimization algorithm](#). In *Studies in Fuzziness and Soft Computing*, pages 105–129. 2005.
- Joaquin Quinero-Candela, Carl Edward Rasmussen, and Christopher K. I. Williams. [Approximation methods for gaussian process regression](#). *Large-scale kernel machines*, pages 203–224, 2007.
- Ali Rahimi and Ben Recht. [Random features for large-scale kernel machines](#). In *Neural Information Processing Systems*, 2007.
- Carl Edward Rasmussen and Christopher K. I. Williams. [Gaussian processes for machine learning](#). MIT Press, 2006.
- Herbert Robbins. [Some aspects of the sequential design of experiments](#). *Bulletin of the American Mathematics Society*, 58:527–535, 1952.
- Jonathan Scarlett, Ilija Bogunovic, and Volkan Cevher. [Lower bounds on regret for noisy Gaussian process bandit optimization](#). In *Conference on Learning Theory*, 2017.
- Matthias Seeger, Christopher Williams, and Neil Lawrence. [Fast forward selection to speed up sparse Gaussian process regression](#). In *International Conference on Artificial Intelligence and Statistics*, 2003.
- Jasper Snoek, Hugo Larochelle, and Ryan P. Adams. [Practical bayesian optimization of machine learning algorithms](#). In *Neural Information Processing Systems*, 2012.
- Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias Seeger. [Gaussian process optimization in the bandit setting: No regret and experimental design](#). *International Conference on Machine Learning*, 2010.
- Joel Aaron Tropp. [An introduction to matrix concentration inequalities](#). *Foundations and Trends in Machine Learning*, 8(1-2):1–230, 2015.
- Michal Valko, Nathan Korda, Rémi Munos, Ilias Flaounas, and Nelo Cristianini. [Finite-time analysis of kernelised contextual bandits](#). In *Uncertainty in Artificial Intelligence*, 2013.

Grace Wahba. *Spline models for observational data*. Society for Industrial and Applied Mathematics, 1990.

Zi Wang, Clement Gehring, Pushmeet Kohli, and Stefanie Jegelka. **Batched Large-scale Bayesian Optimization in High-dimensional Spaces**. In *International Conference on Artificial Intelligence and Statistics*, 2018.

Xiaotian Yu, Michael R. Lyu, and Irwin King. **CBRAP: Contextual bandits with random projection**. In *AAAI Conference on Artificial Intelligence*, 2017.

Appendix A. Relaxing assumptions

In our derivations, we make several assumptions. While some are necessary, others can be relaxed.

Assumptions on the noise. Throughout the paper, we assume that the noise η_t is i.i.d. Gaussian. Since [Chowdhury and Gopalan](#)'s results hold for any ξ -sub-Gaussian noise that is measurable based with respect to the prior observations, this assumption can be easily relaxed.

Assumptions on the arms. So far we considered a set of arms that is (a) in $\mathbb{R}^d$, (b) fixed for all t , and (c) finite. Relaxing (a) is easy, since we do not make any assumption beyond boundedness on the kernel function k and there are many bounded kernel function for non-Euclidean spaces, e.g., strings or graphs. Relaxing (b) is trivial, we just need to embed the changing arm sets as they are provided, and store and re-embed previously selected arms as necessary. The per-step time complexity will now depend on the size of the set of arms available at each step. Relaxing (c) is straightforward from a theoretical perspective, but has varying computational consequences. In particular, looking at the proof of Theorem 1, the guarantees on $\tilde{u}_t$ hold for all $\mathcal{H}$ and in Theorem 2, we only require that the maximum $\tilde{\mathbf{x}}_{t+1} \triangleq \arg \max_{\mathbf{x} \in \mathcal{A}} \max_{\mathbf{w} \in \tilde{C}_t} \phi(\mathbf{x})^\top \mathbf{w}$ is returned. Therefore, at least from the regret point of view, everything holds also for infinite $\mathcal{A}$. However, while the inner maximization over $\tilde{C}_t$ can be solved in closed form for a fixed $\mathbf{x}$, the same cannot be said of the search over $\mathcal{A}$. If the designer can provide an efficient method to solve the maximization problem on an infinite $\mathcal{A}$, e.g., linear bandit optimization over compact subsets or $\mathbb{R}^d$, then all BKB guarantees apply.

Appendix B. Properties of the posterior variance

For simplicity and completeness we provide known statements regarding the posterior variance $\sigma_t^2(\cdot)$. While most of these hold for generic RLS, we will adapt them to our notation.

Proposition 4 ([Calandriello et al., 2017a](#)) *For the posterior variance, we have that*

$$\frac{1}{\kappa^2/\lambda + 1} \sigma_{t-1}^2(\tilde{\mathbf{x}}_t) \leq \frac{1}{\sigma_{t-1}^2(\tilde{\mathbf{x}}_t) + 1} \sigma_{t-1}^2(\tilde{\mathbf{x}}_t) \leq \sigma_t^2(\tilde{\mathbf{x}}_t) \leq \sigma_{t-1}^2(\tilde{\mathbf{x}}_t).$$

Proof The leftmost inequality follows from $\kappa^2/\lambda \geq \sigma_0^2(x)$ and $\sigma_a^2(x) \geq \sigma_b^2(x), \forall a \leq b$, the others are by [Calandriello et al., 2017a](#). $\blacksquare$

Proposition 5 ([Hazan et al., 2006](#); [Calandriello et al., 2017b](#)) *The effective dimension $d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T)$ is upperbounded as*

$$\begin{aligned} d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) &\triangleq \text{Tr}(\mathbf{K}_T(\mathbf{K}_T + \lambda \mathbf{I})^{-1}) = \sum_{t=1}^T \sigma_T^2(\tilde{\mathbf{x}}_t) \\ &\stackrel{(1)}{\leq} \sum_{t=1}^T \sigma_t^2(\tilde{\mathbf{x}}_t) \\ &\stackrel{(2)}{\leq} \log \det(\mathbf{K}_T/\lambda + \mathbf{I}) \\ &\stackrel{(3)}{\leq} \text{Tr}(\mathbf{K}_T(\mathbf{K}_T + \lambda \mathbf{I})^{-1}) \left(1 + \log\left(\frac{\|\mathbf{K}_T\|}{\lambda} + 1\right)\right). \end{aligned}$$

Proof Inequality (1) is due to Proposition 4, inequality (2) is due to [Hazan et al. \(2006\)](#), and Inequality (3) is due to [Calandriello et al. \(2017b\)](#). $\blacksquare$

Appendix C. Proof of Theorem 1

Let B_t be the unfavorable event where the guarantees of Theorem 1 do not hold. Our goal is to prove that B_t happens at most with probability δ uniformly for all $t \in [T]$.

C.1. Notation

In the following we refer to $\Phi(\tilde{\mathbf{X}}_t)$ as Φ_t , $\tilde{\Phi}(\tilde{\mathbf{X}}_t)$ as $\tilde{\Phi}_t$ and $\phi(\tilde{\mathbf{x}}_t)$ as ϕ_t . When the subscript is clear from the context, we omit it. Since we leverage several results of Calandriello et al. (2017b), we start with some additional notation. First we extend our notation for the subset $\mathcal{S}_t$ to include a possible reweighing of the inducing points. We denote with $\mathcal{S}_t \triangleq \{(\phi_j, s_j)\}_{j=1}^{m_t}$, a *weighted* subset, i.e., a *weighted dictionary*, of columns from Φ_t , with positive weights $s_j > 0$ that must be appropriately chosen. Now, denote with $i_j \in [t]$, the index of the sample ϕ_j as a column in Φ_t . Using a standard approach (Alaoui and Mahoney, 2015), we choose $s_j \triangleq 1/\sqrt{\tilde{p}_{t,i_j}}$, where $\tilde{p}_{t,i} \triangleq \bar{q}\tilde{\sigma}_{t-1}^2(\tilde{\mathbf{x}}_i)$ is the probability⁵ used by Algorithm 1 when sampling ϕ_{i_j} from Φ_t .

Let $\mathbf{S}_t \in \mathbb{R}^{t \times t}$ be the diagonal matrix with $q_{t,i}/\sqrt{\tilde{p}_{t,i}}$ on the diagonal, where $q_{t,i}$ are the $\{0, 1\}$ random variables selected by Algorithm 1. Then, we can see that

$$\sum_{j=1}^{m_t} \frac{1}{\tilde{p}_{t,i_j}} \phi_{i_j} \phi_{i_j}^\top = \sum_{i=1}^t \frac{q_{t,i}}{\tilde{p}_{t,i}} \phi_i \phi_i^\top = \Phi_t \mathbf{S}_t \mathbf{S}_t^\top \Phi_t^\top. \quad (11)$$

Calandriello et al. (2017a) define $\mathcal{S}_t$ to be an ε -accurate dictionary of Φ_t if it satisfies

$$(1 - \varepsilon) \Phi_t \Phi_t^\top - \varepsilon \lambda \mathbf{I} \preceq \Phi_t \mathbf{S}_t \mathbf{S}_t^\top \Phi_t^\top \preceq (1 + \varepsilon) \Phi_t \Phi_t^\top + \varepsilon \lambda \mathbf{I}. \quad (12)$$

We can also now fully define the projection operator at time t (see Section 4.1 for more details) as

$$\mathbf{P}_t \triangleq \Phi_t \mathbf{S}_t (\mathbf{S}_t^\top \Phi_t^\top \Phi_t \mathbf{S}_t)^+ \mathbf{S}_t^\top \Phi_t^\top,$$

which is the projection matrix spanned by the dictionary.

C.2. Event decomposition

We decompose Theorem 1 into an accuracy part, i.e., $\mathcal{S}_t$ must induce accurate $\tilde{\sigma}_t$, and an efficiency part, i.e., $m_t \leq d_{\text{eff}}(t)$. We also the accuracy of $\tilde{\sigma}_t$ to the definition of ε -accuracy.

Lemma 6 Let $\alpha \triangleq \frac{1+\varepsilon}{1-\varepsilon}$. If $\mathcal{S}_t$ is ε -accurate w.r.t. Φ_t , then

$$\mathbf{A}_t / \alpha \preceq \tilde{\mathbf{A}}_t \preceq \alpha \mathbf{A}_t \quad \text{and} \quad \sigma_t^2(\mathbf{x}) / \alpha \leq \min\{\tilde{\sigma}_t^2(\mathbf{x}), 1\} \leq \alpha \sigma_t^2(\mathbf{x}) \quad \text{for all } \mathbf{x} \in \mathcal{A}.$$

Proof Inverting the bound in Equation 12 and using the fact that $\mathbf{P}_t \Phi_t \mathbf{S}_t = \Phi_t \mathbf{S}_t$, we get

$$\begin{aligned} \mathbf{P}_t \Phi_t \Phi_t^\top \mathbf{P}_t &\preceq \frac{1}{1-\varepsilon} (\mathbf{P}_t \Phi_t \mathbf{S}_t \mathbf{S}_t^\top \Phi_t^\top \mathbf{P}_t + \varepsilon \lambda \mathbf{P}_t) \preceq \frac{1}{1-\varepsilon} (\Phi_t \mathbf{S}_t \mathbf{S}_t^\top \Phi_t^\top + \varepsilon \lambda \mathbf{P}_t) \\ &\preceq \frac{1}{1-\varepsilon} ((1+\varepsilon) \Phi_t \Phi_t^\top + \varepsilon \lambda \mathbf{I} + \varepsilon \lambda \mathbf{P}_t) \preceq \frac{1+\varepsilon}{1-\varepsilon} \left(\Phi_t \Phi_t^\top + \frac{2\varepsilon}{1+\varepsilon} \lambda \mathbf{I} \right). \end{aligned}$$

5. Note that $\tilde{p}_{t,i}$ might be larger than 1, but with a small abuse of notation and without the loss of generality we still refer to it as a probability.

Repeating the same process for the other side, we obtain

$$\frac{1-\varepsilon}{1+\varepsilon} \left(\Phi_t \Phi_t^\top - \frac{2\varepsilon}{1-\varepsilon} \lambda \mathbf{I} \right) \preceq \mathbf{P}_t \Phi_t \Phi_t^\top \mathbf{P}_t \preceq \frac{1+\varepsilon}{1-\varepsilon} \left(\Phi_t \Phi_t^\top + \frac{2\varepsilon}{1+\varepsilon} \lambda \mathbf{I} \right).$$

Applying the above to $\tilde{\mathbf{A}}_t$, we get

$$\tilde{\mathbf{A}}_t = \mathbf{P}_t \Phi_t \Phi_t^\top \mathbf{P}_t + \lambda \mathbf{I} \succeq \frac{1-\varepsilon}{1+\varepsilon} \left(\Phi_t \Phi_t^\top - \frac{2\varepsilon}{1-\varepsilon} \lambda \mathbf{I} \right) + \lambda \mathbf{I} = \frac{1-\varepsilon}{1+\varepsilon} (\Phi_t \Phi_t^\top + \lambda \mathbf{I}) = \frac{1-\varepsilon}{1+\varepsilon} \mathbf{A}_t,$$

which can again be applied on the other side to obtain our result. To prove the accuracy of the approximate posterior variance $\tilde{\sigma}_t^2(\mathbf{x}_i)$ we simply apply the definition to get

$$\frac{1-\varepsilon}{1+\varepsilon} \overbrace{\phi_i^\top \mathbf{A}_t \phi_i}^{\sigma_t^2(\mathbf{x}_i)} \preceq \overbrace{\phi_i^\top \tilde{\mathbf{A}}_t \phi_i}^{\tilde{\sigma}_t^2(\mathbf{x}_i)} \preceq \frac{1+\varepsilon}{1-\varepsilon} \overbrace{\phi_i^\top \mathbf{A}_t \phi_i}^{\sigma_t^2(\mathbf{x}_i)}.$$

■

Using Lemma 6, we decompose our unfavorable event $B_t \triangleq A_t \cup E_t$, where A_t is the event where $\mathcal{S}_t$ is not ε -accurate w.r.t. Φ_t and E_t is the event where m_t is much larger than $d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_t)$. We now further decompose the event A_t as

$$\begin{aligned} A_t &= (A_t \cap A_{t-1}) \cup (A_t \cap A_{t-1}^c) \\ &\subseteq A_{t-1} \cup (A_t \cap A_{t-1}^c) = A_0 \cup \left(\bigcup_{s=1}^t (A_s \cap A_{s-1}^c) \right) = \bigcup_{s=1}^t (A_s \cap A_{s-1}^c), \end{aligned}$$

where A_0 is the empty event since Φ_0 is empty and it is well approximated by the empty $\mathcal{S}_0$. Moreover, we simplify a part of the expression by noting

$$B_t = A_t \cup E_t = A_t \cup (E_t \cap A_{t-1}^c) \cup (E_t \cap A_{t-1}) \subseteq A_t \cup A_{t-1} \cup (E_t \cap A_{t-1}^c),$$

which will help us when bounding the event E_t , where we will directly act as if A_t does not hold. Putting it all together, we get

$$\begin{aligned} \bigcup_{t=1}^T B_t &= \bigcup_{t=1}^T (A_t \cup E_t) \subseteq \bigcup_{t=1}^T (A_t \cup A_{t-1} \cup (E_t \cap A_{t-1}^c)) \\ &= \left(\bigcup_{t=1}^T A_t \right) \cup \left(\bigcup_{t=1}^T (E_t \cap A_{t-1}^c) \right) = \left(\bigcup_{t=1}^T A_t \right) \cup \left(\bigcup_{t=1}^T (E_t \cap A_{t-1}^c) \right) \\ &\subseteq \left(\bigcup_{t=1}^T \left(\bigcup_{s=1}^t (A_s \cap A_{s-1}^c) \right) \right) \cup \left(\bigcup_{t=1}^T (E_t \cap A_{t-1}^c) \right) \\ &= \left(\bigcup_{t=1}^T (A_t \cap A_{t-1}^c) \right) \cup \left(\bigcup_{t=1}^T (E_t \cap A_{t-1}^c) \right). \end{aligned}$$

C.3. Bounding $\Pr(A_t \cap A_{t-1}^c)$

We now bound the probability of event $A_t \cap A_{t-1}^c$. In our first step, we formally define A_t using Equation 12. In particular, we rewrite the ε -accuracy condition as

$$\begin{aligned} (1 - \varepsilon)\Phi_t\Phi_t^\top - \varepsilon\lambda\mathbf{I} &\preceq \Phi_t\mathbf{S}_t\mathbf{S}_t^\top\Phi_t^\top \preceq (1 + \varepsilon)\Phi_t\Phi_t^\top + \varepsilon\lambda\mathbf{I} \\ \iff -\varepsilon(\Phi_t\Phi_t^\top + \lambda\mathbf{I}) &\preceq \Phi_t\mathbf{S}_t\mathbf{S}_t^\top\Phi_t^\top - \Phi_t\Phi_t^\top \preceq \varepsilon(\Phi_t\Phi_t^\top + \lambda\mathbf{I}) \\ \iff -\varepsilon\mathbf{I} &\preceq (\Phi_t\Phi_t^\top + \lambda\mathbf{I})^{-1/2}(\Phi_t\mathbf{S}_t\mathbf{S}_t^\top\Phi_t^\top - \Phi_t\Phi_t^\top)(\Phi_t\Phi_t^\top + \lambda\mathbf{I})^{-1/2} \preceq \varepsilon\mathbf{I} \\ \iff \lVert (\Phi_t\Phi_t^\top + \lambda\mathbf{I})^{-1/2}(\Phi_t\mathbf{S}_t\mathbf{S}_t^\top\Phi_t^\top - \Phi_t\Phi_t^\top)(\Phi_t\Phi_t^\top + \lambda\mathbf{I})^{-1/2} \rVert &\leq \varepsilon, \end{aligned}$$

where $\lVert \cdot \rVert$ is the spectral norm. We now focus on the last reformulation and frame it as a random matrix concentration question in RKHS $\mathcal{H}$. Let $\psi_{t,i} \triangleq (\Phi_t\Phi_t^\top + \lambda\mathbf{I})^{-\frac{1}{2}}\phi_i$ and $\Psi_t \triangleq \Phi_t(\Phi_t^\top\Phi_t + \lambda\mathbf{I})^{-\frac{1}{2}} = [\psi_{t,1}, \dots, \psi_{t,t}]^\top$, and define the operator $\mathbf{G}_{t,i} \triangleq \left(\frac{q_{t,i}}{\tilde{p}_{t,i}} - 1\right)\psi_{t,i}\psi_{t,i}^\top$. Then we rewrite ε -accuracy as

$$\left\lVert (\Phi_t\Phi_t^\top + \lambda\mathbf{I})^{-\frac{1}{2}}\Phi_t(\mathbf{S}_t\mathbf{S}_t^\top - \mathbf{I})\Phi_t^\top(\Phi_t\Phi_t^\top + \lambda\mathbf{I})^{-\frac{1}{2}} \right\rVert = \left\lVert \sum_{i=1}^t \left(\frac{q_{t,i}}{\tilde{p}_{t,i}} - 1\right)\psi_{t,i}\psi_{t,i}^\top \right\rVert = \left\lVert \sum_{i=1}^t \mathbf{G}_{t,i} \right\rVert \leq \varepsilon,$$

and the event A_t as the event where $\left\lVert \sum_{i=1}^t \mathbf{G}_{t,i} \right\rVert \geq \varepsilon$. Note that this reformulation exploits the fact that $q_{t,i} = 0$ encodes the column that are not selected in $\mathbf{S}_t$ (see Equation 11). To study this random object, we begin by defining the filtration $\mathcal{F}_t \triangleq \{q_{s,i}, \eta_s\}_{s=1}^t$ at time t containing all the randomness coming from the construction of the various $\mathcal{S}_s$ and the noise on the function η_t . In particular, note that the $\{0, 1\}$ r.v. $q_{t,i}$ used by Algorithm 1 are not necessarily Bernoulli r.v.s, since the probability $\tilde{p}_{t,i}$ used to select 0 or 1 is itself random. However, they become well defined Bernoulli when conditioned on $\mathcal{F}_{t-1}$. Let $\mathbb{I}\{\cdot\}$ indicates the indicator function of an event. We have that

$$\begin{aligned} \Pr(A_t \cap A_{t-1}^c) &= \Pr\left(\left\lVert \sum_{i=1}^t \mathbf{G}_{t,i} \right\rVert \geq \varepsilon \cap \left\lVert \sum_{i=1}^t \mathbf{G}_{t-1,i} \right\rVert \leq \varepsilon\right) \\ &= \mathbb{E}_{\mathcal{F}_t} \left[\mathbb{I}\left\{\left\lVert \sum_{i=1}^t \mathbf{G}_{t,i} \right\rVert \geq \varepsilon \cap \left\lVert \sum_{i=1}^t \mathbf{G}_{t-1,i} \right\rVert \leq \varepsilon\right\} \right] \\ &= \mathbb{E}_{\mathcal{F}_{t-1}} \left[\mathbb{E}_{\eta_t, \{q_{t,i}\}} \left[\mathbb{I}\left\{\left\lVert \sum_{i=1}^t \mathbf{G}_{t,i} \right\rVert \geq \varepsilon \cap \left\lVert \sum_{i=1}^t \mathbf{G}_{t-1,i} \right\rVert \leq \varepsilon\right\} \middle| \mathcal{F}_{t-1} \right] \right] \\ &= \mathbb{E}_{\mathcal{F}_{t-1}} \left[\mathbb{E}_{\{q_{t,i}\}} \left[\mathbb{I}\left\{\left\lVert \sum_{i=1}^t \mathbf{G}_{t,i} \right\rVert \geq \varepsilon \cap \left\lVert \sum_{i=1}^t \mathbf{G}_{t-1,i} \right\rVert \leq \varepsilon\right\} \middle| \mathcal{F}_{t-1} \right] \right], \end{aligned}$$

where the last passage is due to the fact that $\mathbf{G}_{t,i}$ is independent from η_t . Next, notice that conditioned on $\mathcal{F}_{t-1}$, the event A_{t-1}^c becomes deterministic, and we can restrict our expectations to the outcomes where $\left\lVert \sum_{i=1}^t \mathbf{G}_{t-1,i} \right\rVert \leq \varepsilon$,

$$\Pr(A_t \cap A_{t-1}^c) = \mathbb{E}_{\mathcal{F}_{t-1}: \left\lVert \sum_{i=1}^t \mathbf{G}_{t-1,i} \right\rVert \leq \varepsilon} \left[\mathbb{E}_{\{q_{t,i}\}} \left[\mathbb{I}\left\{\left\lVert \sum_{i=1}^t \mathbf{G}_{t,i} \right\rVert \geq \varepsilon\right\} \middle| \mathcal{F}_{t-1} \right] \right].$$

Moreover, conditioned on $\mathcal{F}_{t-1}$ all the $q_{t,i}$ s become independent r.v.s, and we are able to use the following result of [Tropp \(2015\)](#).

Proposition 7 Let $\mathbf{G}_1, \dots, \mathbf{G}_n$ be a sequence of independent self-adjoint random operators such that $\mathbb{E}[\mathbf{G}_i] = 0$ and $\|\mathbf{G}_i\| \leq R$ a.s. Denote $\sigma^2 = \|\sum_{i=1}^t \mathbb{E}[\mathbf{G}_i^2]\|$. Then, for any $\varepsilon \geq 0$,

$$\Pr\left(\left\|\sum_{i=1}^t \mathbf{G}_i\right\| \geq \varepsilon\right) \leq 4t \exp\left(\frac{\varepsilon^2/2}{\sigma^2 + R\varepsilon/3}\right).$$

We begin by computing the mean of $\mathbf{G}_{t,i}$,

$$\begin{aligned} \mathbb{E}_{q_{t,i}}[\mathbf{G}_{t,i} \mid \mathcal{F}_{t-1}] &= \mathbb{E}_{q_{t,i}}\left[\left(\frac{q_{t,i}}{\tilde{p}_{t,i}} - 1\right) \psi_{t,i} \psi_{t,i}^\top \mid \mathcal{F}_{t-1}\right] \\ &= \left(\frac{\mathbb{E}_{q_{t,i}}[q_{t,i} \mid \mathcal{F}_{t-1}]}{\tilde{p}_{t,i}} - 1\right) \psi_{t,i} \psi_{t,i}^\top = \left(\frac{\tilde{p}_{t,i}}{\tilde{p}_{t,i}} - 1\right) \psi_{t,i} \psi_{t,i}^\top = \mathbf{0}, \end{aligned}$$

where we use the fact that $\tilde{p}_{t,i}$ is fixed conditioned on $\mathcal{F}_{t-1}$ and it is the (conditional) expectation of $q_{t,i}$. Since $\mathbf{G}$ is zero-mean, we can use Proposition 7. First, we find R and for that, we upper bound

$$\|\mathbf{G}_{t,i}\| = \left\|\left(\frac{q_{t,i}}{\tilde{p}_{t,i}} - 1\right) \psi_{t,i} \psi_{t,i}^\top\right\| \leq \left|\left(\frac{q_{t,i}}{\tilde{p}_{t,i}} - 1\right)\right| \|\psi_{t,i} \psi_{t,i}^\top\| \leq \frac{1}{\tilde{p}_{t,i}} \|\psi_{t,i} \psi_{t,i}^\top\|.$$

Note that due to the definition of $\psi_{t,i}$,

$$\|\psi_{t,i} \psi_{t,i}^\top\| = \psi_{t,i}^\top \psi_{t,i} = \phi_i^\top (\Phi_t \Phi_t^\top + \lambda \mathbf{I})^{-1} \phi_i = \sigma_t^2(\tilde{\mathbf{x}}_i).$$

Moreover, we are only considering outcomes of $\mathcal{F}_{t-1}$ where $\|\sum_{i=1}^t \mathbf{G}_{t-1,i}\| \leq \varepsilon$, which implies that $\mathcal{S}_{t-1}$ is ε -accurate, and by Lemma 6 we have that $\tilde{\sigma}_{t-1}(\tilde{\mathbf{x}}_i) \geq \sigma_{t-1}(\tilde{\mathbf{x}}_i)/\alpha$. Finally, due to Proposition 4, we have $\sigma_{t-1}(\tilde{\mathbf{x}}_i) \geq \sigma_t(\tilde{\mathbf{x}}_i)$. Putting this all together we can bound

$$\frac{1}{\tilde{p}_{t,i}} \|\psi_{t,i} \psi_{t,i}^\top\| = \frac{1}{\tilde{q} \tilde{\sigma}_{t-1}(\tilde{\mathbf{x}}_i)} \sigma_t(\tilde{\mathbf{x}}_i) \leq \frac{\alpha}{\tilde{q}} \triangleq R.$$

For the variance term, we expand

$$\begin{aligned} \sum_{i=1}^t \mathbb{E}_{q_{t,i}}[\mathbf{G}_{t,i}^2 \mid \mathcal{F}_{t-1}] &= \sum_{i=1}^t \mathbb{E}_{q_{t,i}}\left[\left(\frac{q_{t,i}}{\tilde{p}_{t,i}} - 1\right)^2 \mid \mathcal{F}_{t-1}\right] \psi_{t,i} \psi_{t,i}^\top \psi_{t,i} \psi_{t,i}^\top \\ &= \sum_{i=1}^t \left(\mathbb{E}_{q_{t,i}}\left[\frac{q_{t,i}^2}{\tilde{p}_{t,i}^2} \mid \mathcal{F}_{t-1}\right] - \mathbb{E}_{q_{t,i}}\left[2\frac{q_{t,i}}{\tilde{p}_{t,i}} \mid \mathcal{F}_{t-1}\right] + 1\right) \psi_{t,i} \psi_{t,i}^\top \psi_{t,i} \psi_{t,i}^\top \\ &= \sum_{i=1}^t \left(\mathbb{E}_{q_{t,i}}\left[\frac{q_{t,i}}{\tilde{p}_{t,i}^2} \mid \mathcal{F}_{t-1}\right] - 1\right) \psi_{t,i} \psi_{t,i}^\top \psi_{t,i} \psi_{t,i}^\top = \sum_{i=1}^t \left(\mathbb{E}_{q_{t,i}}\left[\frac{q_{t,i}}{\tilde{p}_{t,i}^2} \mid \mathcal{F}_{t-1}\right] - 1\right) \psi_{t,i} \psi_{t,i}^\top \psi_{t,i} \psi_{t,i}^\top \\ &= \sum_{i=1}^t \left(\frac{1}{\tilde{p}_{t,i}} - 1\right) \psi_{t,i} \psi_{t,i}^\top \psi_{t,i} \psi_{t,i}^\top \leq \sum_{i=1}^t \frac{1}{\tilde{p}_{t,i}} \|\psi_{t,i} \psi_{t,i}^\top\| \|\psi_{t,i} \psi_{t,i}^\top\| \leq \sum_{i=1}^t R \psi_{t,i} \psi_{t,i}^\top, \end{aligned}$$

where we used the fact that $q_{t,i}^2 = q_{t,i}$ and $\mathbb{E}_{q_{t,i}}[q_{t,i} \mid \mathcal{F}_{t-1}] = \tilde{p}_{t,i}$. We can now bound this quantity as

$$\left\|\sum_{i=1}^t \mathbb{E}_{q_{t,i}}[\mathbf{G}_{t,i}^2 \mid \mathcal{F}_{t-1}]\right\| \leq \left\|\sum_{i=1}^t R \psi_{t,i} \psi_{t,i}^\top\right\| = R \left\|\sum_{i=1}^t \psi_{t,i} \psi_{t,i}^\top\right\| = R \|\Psi_t^\top \Psi_t\| \leq R \triangleq \sigma^2.$$

Therefore, we have $\sigma^2 = R$ and $R = 1/\tilde{q}$. Now, applying Proposition 7 and a union bound we conclude the proof.

C.4. Bounding $\Pr(E_t \cap A_{t-1}^c)$

We will use the following concentration for independent Bernoulli random variables.

Proposition 8 (Calandriello et al., 2017a, App. D.4) *Let $\{q_s\}_{s=1}^t$ be independent Bernoulli random variables, each with success probability p_s , and let $d = \sum_{s=1}^t p_s \geq 1$ be their sum. Then,⁶*

$$\mathbb{P}\left(\sum_{s=1}^t q_s \geq 3d\right) \leq \exp\{-3d(3d - (\log(3d) + 1))\} \leq \exp\{-2d\}.$$

We now rigorously define event E_t as the event where

$$\sum_{i=1}^t q_{t,i} \geq 3\alpha(1 + \kappa^2/\lambda) \log(t/\delta) \sum_{i=1}^t \sigma_t^2(\tilde{\mathbf{x}}_i) = 3\alpha(1 + \kappa^2/\lambda) d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_t) \log(t/\delta).$$

Once again, we use conditioning and in particular,

$$\Pr(E_t \cap A_t^c) = \mathbb{E}_{\mathcal{F}_{t-1}: \|\sum_{i=1}^t \mathbf{G}_{t-1,i}\| \leq \varepsilon} \left[\mathbb{E}_{\{q_{t,i}\}} \left[\mathbb{I} \left\{ \sum_{i=1}^t q_{t,i} \geq 3\alpha(1 + \kappa^2/\lambda) \log(t/\delta) \sum_{i=1}^t \sigma_t^2(\tilde{\mathbf{x}}_i) \right\} \middle| \mathcal{F}_{t-1} \right] \right].$$

Conditioned on $\mathcal{F}_{t-1}$ the r.v. $q_{t,i}$ becomes independent Bernoulli with probability $\tilde{p}_{t,i} \triangleq \bar{q} \tilde{\sigma}_{t-1}(\tilde{\mathbf{x}}_i)$. Since we restrict the outcomes to A_{t-1}^c , we can exploit Lemma 6 and the guarantees of ε -accuracy to bound $\tilde{p}_{t,i} \leq \alpha \sigma_{t-1}^2(\tilde{\mathbf{x}}_i)$. Then, we use Proposition 4 to bound $\sigma_{t-1}^2(\tilde{\mathbf{x}}_i) \leq (1 + \kappa^2/\lambda) \sigma_t^2(\tilde{\mathbf{x}}_i)$. Therefore, $q_{t,i}$ are conditionally independent Bernoulli with probability at most $\bar{q}(1 + \kappa^2/\lambda) \sigma_t^2(\tilde{\mathbf{x}}_i)$. Applying a simple stochastic dominance argument and Proposition 8 gets the needed statement.

Appendix D. Proof of Theorem 2

Following Abbasi-Yadkori et al. (2011), we divide the proof in two parts, first bounding the approximate confidence ellipsoid, and then bounding the regret.

D.1. Bounding the confidence ellipsoid

We begin by proving an intermediate result regarding the confidence ellipsoid.

Theorem 9 *Under the same assumptions as Theorem 2 with probability at least $1 - \delta$ and for all $t \geq 0$, $\mathbf{w}_\star$ lies in the set*

$$\tilde{C}_t \triangleq \left\{ \mathbf{w} : \|\mathbf{w} - \tilde{\mathbf{w}}_t\|_{\tilde{\mathbf{A}}_t} \leq \tilde{\beta}_t \right\}$$

with

$$\tilde{\beta}_t \triangleq 2\xi \sqrt{\alpha \log(\kappa^2 t) \left(\sum_{s=1}^t \tilde{\sigma}_t^2(\mathbf{x}_s) \right) + \log\left(\frac{1}{\delta}\right) + \left(1 + \frac{1}{\sqrt{1-\varepsilon}}\right) \sqrt{\lambda} F}.$$

6. This is a simple variant of the Chernoff bound where the Bernoulli random variables are not identically distributed.

Proof For simplicity, we omit the subscript t . We begin by noticing that

$$\begin{aligned}
(\tilde{\mathbf{w}} - \mathbf{w}_\star)^\top \tilde{\mathbf{A}}(\tilde{\mathbf{w}} - \mathbf{w}_\star) &= (\tilde{\mathbf{w}} - \mathbf{w}_\star)^\top \tilde{\mathbf{A}}(\tilde{\mathbf{A}}^{-1} \tilde{\Phi}^\top \mathbf{y} - \mathbf{w}_\star) \\
&= (\tilde{\mathbf{w}} - \mathbf{w}_\star)^\top \tilde{\mathbf{A}}(\tilde{\mathbf{A}}^{-1} \tilde{\Phi}^\top (\Phi \mathbf{w}_\star + \eta - \mathbf{w}_\star)) \\
&= (\tilde{\mathbf{w}} - \mathbf{w}_\star)^\top \tilde{\mathbf{A}}(\underbrace{\tilde{\mathbf{A}}^{-1} \tilde{\Phi}^\top \Phi \mathbf{w}_\star}_{\text{bias}} - \mathbf{w}_\star) + (\tilde{\mathbf{w}} - \mathbf{w}_\star)^\top \tilde{\mathbf{A}}^{1/2} \underbrace{\tilde{\mathbf{A}}^{-1/2} \tilde{\Phi}^\top \eta}_{\text{variance}}.
\end{aligned}$$

Bounding the bias. We first focus on the first term, which is difficult to analyze due to the mismatch $\tilde{\Phi}^\top \Phi$. We have that

$$\begin{aligned}
\tilde{\mathbf{A}}(\tilde{\mathbf{A}}^{-1} \tilde{\Phi}^\top \Phi \mathbf{w}_\star - \mathbf{w}_\star) &= \tilde{\Phi}^\top \Phi \mathbf{w}_\star - \tilde{\Phi}^\top \tilde{\Phi} \mathbf{w}_\star - \lambda \mathbf{w}_\star \\
&= \tilde{\Phi}^\top \Phi (\mathbf{I} - \mathbf{P}) \mathbf{w}_\star + \tilde{\Phi}^\top \Phi \mathbf{P} \mathbf{w}_\star - \tilde{\Phi}^\top \tilde{\Phi} \mathbf{w}_\star - \lambda \mathbf{w}_\star \\
&= \tilde{\Phi}^\top \Phi (\mathbf{I} - \mathbf{P}) \mathbf{w}_\star - \lambda \mathbf{w}_\star.
\end{aligned}$$

Therefore,

$$\begin{aligned}
(\tilde{\mathbf{w}} - \mathbf{w}_\star)^\top \tilde{\mathbf{A}}(\tilde{\mathbf{A}}^{-1} \tilde{\Phi}^\top \Phi \mathbf{w}_\star - \mathbf{w}_\star) &= (\tilde{\mathbf{w}} - \mathbf{w}_\star)^\top \tilde{\Phi}^\top \Phi (\mathbf{I} - \mathbf{P}) \mathbf{w}_\star - \lambda (\tilde{\mathbf{w}} - \mathbf{w}_\star)^\top \mathbf{w}_\star \\
&\leq \|\tilde{\mathbf{w}} - \mathbf{w}_\star\|_{\tilde{\mathbf{A}}} \left(\|\tilde{\mathbf{A}}^{-1/2} \tilde{\Phi}^\top \Phi (\mathbf{I} - \mathbf{P}) \mathbf{w}_\star\| + \lambda \|\mathbf{w}_\star\|_{\tilde{\mathbf{A}}^{-1}} \right) \\
&\leq \|\tilde{\mathbf{w}} - \mathbf{w}_\star\|_{\tilde{\mathbf{A}}} \left(\|\tilde{\mathbf{A}}^{-1/2} \tilde{\Phi}^\top \Phi (\mathbf{I} - \mathbf{P}) \mathbf{w}_\star\| + \frac{\lambda}{\sqrt{\lambda}} \|\mathbf{w}_\star^*\| \right).
\end{aligned}$$

Then, we have that

$$\begin{aligned}
\|\tilde{\mathbf{A}}^{-1/2} \tilde{\Phi}^\top \Phi (\mathbf{I} - \mathbf{P}) \mathbf{w}_\star\| &\leq \|\tilde{\mathbf{A}}^{-1/2} \tilde{\Phi}^\top\| \|\Phi (\mathbf{I} - \mathbf{P})\| \|\mathbf{w}_\star\| \\
&\leq \sqrt{\lambda_{\max}(\tilde{\Phi} \tilde{\mathbf{A}}^{-1} \tilde{\Phi}^\top)} \sqrt{\lambda_{\max}(\Phi (\mathbf{I} - \mathbf{P})^2 \Phi^\top)} \|\mathbf{w}_\star\|.
\end{aligned}$$

It is easy to see that

$$\lambda_{\max}(\tilde{\Phi} \tilde{\mathbf{A}}^{-1} \tilde{\Phi}^\top) = \lambda_{\max}(\tilde{\Phi}(\tilde{\Phi}^\top \tilde{\Phi} + \lambda \mathbf{I})^{-1} \tilde{\Phi}^\top) \leq 1.$$

To bound the other term we use the following result by [Calandriello and Rosasco \(2018\)](#).

Proposition 10 *If $\mathcal{S}_t$ is ε -accurate w.r.t. Φ_t , then*

$$\mathbf{I} - \mathbf{P}_t \preceq \mathbf{I} - \Phi_t \mathbf{S}_t (\mathbf{S}_t^\top \Phi_t^\top \Phi_t \mathbf{S}_t + \lambda \mathbf{I})^{-1} \mathbf{S}_t^\top \Phi_t^\top \preceq \frac{\lambda}{1 - \varepsilon} (\Phi_t \Phi_t^\top + \lambda \mathbf{I})^{-1}.$$

Since from Theorem 1, we have that $\mathcal{S}_t$ is ε -accurate, by Theorem 10, we have that

$$\Phi (\mathbf{I} - \mathbf{P})^2 \Phi^\top = \Phi (\mathbf{I} - \mathbf{P}) \Phi^\top \preceq \frac{\lambda}{1 - \varepsilon} \Phi (\Phi^\top \Phi + \lambda \mathbf{I})^{-1} \Phi^\top \preceq \frac{\lambda}{1 - \varepsilon} \mathbf{I}.$$

Putting it all together, we obtain

$$(\tilde{\mathbf{w}} - \mathbf{w}_\star)^\top \tilde{\mathbf{A}}(\tilde{\mathbf{A}}^{-1} \tilde{\Phi}^\top \Phi \mathbf{w}_\star - \mathbf{w}_\star) \leq \left(1 + \frac{1}{\sqrt{1 - \varepsilon}}\right) \|\tilde{\mathbf{w}} - \mathbf{w}_\star\|_{\tilde{\mathbf{A}}} \sqrt{\lambda} \|\mathbf{w}_\star\|.$$

Bounding the variance. We use the the following self-normalized martingale concentration inequality by [Abbasi-Yadkori et al. \(2011\)](#). It can be trivially extended to RKHSs in the case of finite sets such as our $\mathcal{A}$. Note that if the reader is interested in infinite sets, [Chowdhury and Gopalan \(2017\)](#) provide a generalization with slightly worse constants.

Proposition 11 (Abbasi-Yadkori et al., 2011) *Let $\{\mathcal{F}_t\}_{t=0}^\infty$ be a filtration, let $\{\eta_t\}_{t=1}^\infty$ be a real-valued stochastic process such that η_t is $\mathcal{F}_t$ -measurable and zero-mean ξ -subgaussian; let $\{\Phi_t\}_{t=1}^\infty$ be an $\mathcal{H}$ -valued stochastic process such that Φ_t is $\mathcal{F}_{t-1}$ -measurable, and let $\mathbf{I}$ be the identity operator on $\mathcal{H}$. For any $t \geq 1$, define*

$$\mathbf{A}_t = \Phi_t^\top \Phi_t + \lambda \mathbf{I} \quad \text{and} \quad \mathbf{V}_t = \Phi_t^\top \eta_t.$$

Then, for any $\delta > 0$, with probability at least $1 - \delta$, for all $t \geq 0$,

$$\|\mathbf{V}_t\|_{\mathbf{A}_t^{-1}}^2 \leq 2\xi^2 \log\left(\frac{\det(\mathbf{A}_t/\lambda)}{\delta}\right).$$

Recalling the definition of $\alpha \geq 1$ from Theorem 1, we reformulate

$$\begin{aligned} (\tilde{\mathbf{w}} - \mathbf{w}_*)^\top \tilde{\mathbf{A}}^{1/2} \tilde{\mathbf{A}}^{-1/2} \tilde{\Phi} \eta &\leq \|\tilde{\mathbf{w}} - \mathbf{w}_*\|_{\tilde{\mathbf{A}}} \|\tilde{\Phi} \eta\|_{\tilde{\mathbf{A}}^{-1}} \\ &= \|\tilde{\mathbf{w}} - \mathbf{w}_*\|_{\tilde{\mathbf{A}}} \|\tilde{\Phi}^\top \eta\|_{(\tilde{\Phi}^\top \tilde{\Phi} + \lambda \mathbf{I})^{-1}} \\ &= \|\tilde{\mathbf{w}} - \mathbf{w}_*\|_{\tilde{\mathbf{A}}} \|\tilde{\Phi}^\top \eta / \lambda\|_{(\tilde{\Phi}^\top \tilde{\Phi} / \lambda + \mathbf{I})^{-1}}. \end{aligned}$$

We now make a remark that requires temporal notation. Note that we cannot directly apply Theorem 11 to $\tilde{\Phi}_t \eta_t = \mathbf{P}_t \Phi_t \eta_t$. In particular, for $s < t$ we have that $\tilde{\Phi}_s \eta_s = \mathbf{P}_t \Phi_s \eta_s$ is not $\mathcal{F}_{s-1}$ measurable, since $\mathbf{P}_t$ depends on all randomness up to time t . However, since $\mathbf{P}_t$ is always a projection matrix we know that the variance of the projected process is bounded by the variance of the original process, in particular,

$$\begin{aligned} \|\tilde{\Phi}^\top \eta / \lambda\|_{(\tilde{\Phi}^\top \tilde{\Phi} / \lambda + \mathbf{I})^{-1}} &= \sqrt{\eta^\top \tilde{\Phi} (\tilde{\Phi}^\top \tilde{\Phi} / \lambda + \mathbf{I})^{-1} \tilde{\Phi}^\top \eta / \lambda} = \sqrt{\eta^\top \tilde{\Phi} \tilde{\Phi}^\top (\tilde{\Phi} \tilde{\Phi}^\top / \lambda + \mathbf{I})^{-1} \eta / \lambda} \\ &\stackrel{(a)}{=} \sqrt{\eta^\top (\mathbf{I} - \lambda (\tilde{\Phi} \tilde{\Phi}^\top / \lambda + \mathbf{I})^{-1}) \eta / \lambda} = \sqrt{\eta^\top (\mathbf{I} - \lambda (\Phi \mathbf{P} \Phi^\top / \lambda + \mathbf{I})^{-1}) \eta / \lambda} \\ &\stackrel{(b)}{\leq} \sqrt{\eta^\top (\mathbf{I} - \lambda (\Phi \Phi^\top / \lambda + \mathbf{I})^{-1}) \eta / \lambda} \stackrel{(c)}{=} \|\Phi^\top \eta / \lambda\|_{(\Phi^\top \Phi / \lambda + \mathbf{I})^{-1}}, \end{aligned}$$

where in (a) we added and subtracted $\lambda \mathbf{I}$ from $\tilde{\Phi} \tilde{\Phi}^\top$, in (b) we used the fact that $\|\mathbf{P}\| \leq 1$ for all projection matrices, and in (c) we reversed the reformulation from (a). We can finally use Theorem 11 to obtain

$$\begin{aligned} \|\Phi^\top \eta / \lambda\|_{(\Phi^\top \Phi / \lambda + \mathbf{I})^{-1}} &\leq \sqrt{2\xi^2 \log(\det(\Phi^\top \Phi / \lambda + \mathbf{I}) / \delta)} \\ &= \sqrt{2\xi^2 \log(\det(\mathbf{A} / \lambda) / \delta)}. \end{aligned}$$

While above is a valid bound on the radius of the confidence interval, it is still not satisfactory. In particular, we can use Sylvester's identity to reformulate

$$\log \det(\mathbf{A} / \lambda) = \log \det(\Phi^\top \Phi / \lambda + \mathbf{I}) = \log \det(\Phi \Phi^\top / \lambda + \mathbf{I}) = \log \det(\mathbf{K} / \lambda + \mathbf{I}).$$

Computing the radius would require constructing the matrix $\mathbf{K} \in \mathbb{R}^{t \times t}$ and this is way too expensive. Instead, we obtain a cheap but still a small enough upper bound as follows,

$$\begin{aligned}
\log \det(\mathbf{K}_t/\lambda + \mathbf{I}) &\leq \text{Tr}(\mathbf{K}_t(\mathbf{K}_t + \lambda\mathbf{I})^{-1})(1 + \log(\|\mathbf{K}_t\| + 1)) \\
&\leq \text{Tr}(\mathbf{K}_t(\mathbf{K}_t + \lambda\mathbf{I})^{-1})(1 + \log(\text{Tr } \mathbf{K}_t + 1)) \\
&\leq \text{Tr}(\mathbf{K}_t(\mathbf{K}_t + \lambda\mathbf{I})^{-1})(1 + \log(\kappa^2 t + 1)) \\
&= (1 + \log(\kappa^2 t + 1)) \sum_{s=1}^t \sigma_t^2(\mathbf{x}_s) \\
&\leq \alpha(1 + \log(\kappa^2 t + 1)) \sum_{s=1}^t \tilde{\sigma}_t^2(\mathbf{x}_s) \\
&\leq 2\alpha \log(\kappa^2 t) \sum_{s=1}^t \tilde{\sigma}_t^2(\mathbf{x}_s),
\end{aligned}$$

where $\tilde{\sigma}_t^2(\mathbf{x}_s)$ can be computed efficiently and it is actually already done by the algorithm at every step! Putting it all together, we get that

$$\begin{aligned}
\|\tilde{\mathbf{w}} - \mathbf{w}_\star\|_{\tilde{\mathbf{A}}} &\leq 2\xi \sqrt{\alpha \log(\kappa^2 t) \left(\sum_{s=1}^t \tilde{\sigma}_t^2(\mathbf{x}_s) \right) + \log(1/\delta) + \left(1 + \frac{1}{\sqrt{1-\varepsilon}} \right) \sqrt{\lambda} \|\mathbf{w}_\star\|} \\
&\leq 2\xi \sqrt{\alpha \log(\kappa^2 t) \left(\sum_{s=1}^t \tilde{\sigma}_t^2(\mathbf{x}_s) \right) + \log(1/\delta) + \left(1 + \frac{1}{\sqrt{1-\varepsilon}} \right) \sqrt{\lambda} F \triangleq \tilde{\beta}_t}.
\end{aligned}$$

■

D.2. Bounding the regret

The regret analysis is straightforward. Assume that $\mathbf{w}_\star \in \tilde{C}_t$ is satisfied (i.e., the event from Theorem 9 holds) and remember that by the definition, $\phi_t \triangleq \arg \max_{\mathbf{x}_i \in \mathcal{A}} \max_{\mathbf{w} \in \tilde{C}_t} \phi_i^\top \mathbf{w}$. We also define $\bar{\mathbf{w}}_{t,i} \triangleq \arg \max_{\mathbf{w} \in \tilde{C}_t} \phi_i^\top \mathbf{w}$ as the auxiliary vector which encodes the optimistic behaviour of the algorithm. With a slight abuse of notation, we also use $\star$ as a subscript to indicate the (unknown) index of the optimal arm, so that $\bar{\mathbf{w}}_{t,\star} \triangleq \arg \max_{\mathbf{w} \in \tilde{C}_t} \phi_\star^\top \mathbf{w}$. Since $\mathbf{w}_\star \in \tilde{C}_t$, we have that

$$\phi_t^\top \bar{\mathbf{w}}_{t,t} \geq \phi_\star^\top \bar{\mathbf{w}}_{t,\star} \geq \phi_\star^\top \mathbf{w}_\star.$$

We can now bound the instantaneous regret r_t as

$$\begin{aligned}
r_t &= \phi_\star^\top \mathbf{w}_\star - \phi_t^\top \mathbf{w}_\star \leq \phi_t^\top \bar{\mathbf{w}}_{t,t} - \phi_t^\top \mathbf{w}_\star \\
&= \phi_t^\top (\bar{\mathbf{w}}_{t,t} - \hat{\mathbf{w}}_t) + \phi_t^\top (\hat{\mathbf{w}}_t - \mathbf{w}_\star) \\
&= \phi_t^\top \tilde{\mathbf{A}}_t^{-1/2} \tilde{\mathbf{A}}_t^{1/2} (\bar{\mathbf{w}}_{t,t} - \hat{\mathbf{w}}_t) + \phi_t^\top \tilde{\mathbf{A}}_{t-1}^{-1/2} \tilde{\mathbf{A}}_t^{1/2} (\hat{\mathbf{w}}_t - \mathbf{w}_\star) \\
&\leq \sqrt{\phi_t^\top \tilde{\mathbf{A}}_t^{-1} \phi_t} \left(\|\bar{\mathbf{w}}_{t,t} - \hat{\mathbf{w}}_t\|_{\tilde{\mathbf{A}}_t} + \|\hat{\mathbf{w}}_t - \mathbf{w}_\star\|_{\tilde{\mathbf{A}}_t} \right) \\
&\leq 2\tilde{\beta}_t \sqrt{\phi_t^\top \tilde{\mathbf{A}}_t^{-1} \phi_t}.
\end{aligned}$$

Summing over t and taking the max over $\tilde{\beta}_t$, we get

$$R_t \leq 2\tilde{\beta}_T \sum_{t=1}^T \sqrt{\phi_t^\top \tilde{\mathbf{A}}_t^{-1} \phi_t} \leq 2\tilde{\beta}_T \sqrt{T} \sqrt{\sum_{t=1}^T \phi_t^\top \tilde{\mathbf{A}}_t^{-1} \phi_t} \leq 2\tilde{\beta}_T \sqrt{T} \sqrt{\alpha \sum_{t=1}^T \phi_t^\top \mathbf{A}_t^{-1} \phi_t}.$$

We can now use once again Proposition 5 to obtain

$$R_T \leq 2\tilde{\beta}_T \sqrt{\alpha T \sum_{t=1}^T \phi_t^\top \mathbf{A}_t^{-1} \phi_t} = 2\tilde{\beta}_T \sqrt{\alpha T \sum_{t=1}^T \sigma_t^2(\tilde{\mathbf{x}}_t)} \leq 2\tilde{\beta}_T \sqrt{2\alpha T d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) \log(\kappa^2 T)}.$$

We can also further upper bound $\tilde{\beta}_T$ as

$$\begin{aligned} \tilde{\beta}_T &= 2\xi \sqrt{\alpha \log(\kappa^2 T) \left(\sum_{s=1}^T \tilde{\sigma}_t^2(\mathbf{x}_s) \right) + \log(1/\delta) + \left(1 + \frac{1}{\sqrt{1-\varepsilon}} \right) \sqrt{\lambda} F} \\ &\leq 2\xi \sqrt{\alpha^2 \log(\kappa^2 T) \left(\sum_{s=1}^T \sigma_t^2(\mathbf{x}_s) \right) + \log(1/\delta) + \left(1 + \frac{1}{\sqrt{1-\varepsilon}} \right) \sqrt{\lambda} F} \\ &\leq 2\xi \alpha \sqrt{d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) \log(\kappa^2 T) + \log(1/\delta) + \left(1 + \frac{1}{\sqrt{1-\varepsilon}} \right) \sqrt{\lambda} F}. \end{aligned}$$

Putting it together, we obtain

$$\begin{aligned} R_T &\leq 2 \left(2\xi \alpha \sqrt{d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) \log(\kappa^2 T) + \log(1/\delta)} \right) \sqrt{2\alpha T d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) \log(\kappa^2 T)} \\ &\quad + 2 \left(\left(1 + \frac{1}{\sqrt{1-\varepsilon}} \right) \sqrt{\lambda} F \right) \sqrt{2\alpha T d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) \log(\kappa^2 T)} \\ &\leq 2\xi (2\alpha)^{3/2} \left(d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) \log(\kappa^2 T) + \log(1/\delta) \right) + 2 \left(2\sqrt{\alpha} \sqrt{\lambda} F \right) \sqrt{2\alpha T d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) \log(\kappa^2 T)} \\ &\leq 2(2\alpha)^{3/2} \left(\sqrt{T} \xi d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) \log(\kappa^2 T) + \sqrt{T} \log(1/\delta) + \sqrt{T \lambda F^2 d_{\text{eff}}(\lambda, \tilde{\mathbf{X}}_T) \log(\kappa^2 T)} \right). \end{aligned}$$

Implicit Regularization of Accelerated Methods in Hilbert Spaces

Nicolò Pagliana
University of Genoa
DIMA & MaLGa
pagliana@dima.unige.it

Lorenzo Rosasco
University of Genoa
DIBRIS, MaLGa, IIT & MIT
lrosasco@mit.edu

Abstract

We study learning properties of accelerated gradient descent methods for linear least-squares in Hilbert spaces. We analyze the implicit regularization properties of Nesterov acceleration and a variant of heavy-ball in terms of corresponding learning error bounds. Our results show that acceleration can provide faster bias decay than gradient descent, but also suffers of a more unstable behavior. As a result acceleration cannot be in general expected to improve learning accuracy with respect to gradient descent, but rather to achieve the same accuracy with reduced computations. Our theoretical results are validated by numerical simulations. Our analysis is based on studying suitable polynomials induced by the accelerated dynamics and combining spectral techniques with concentration inequalities.

1 Introduction

The focus on optimization is a major trend in modern machine learning, where efficiency is mandatory in large scale problems [4]. Among other solutions, first order methods have emerged as methods of choice. While these techniques are known to have potentially slow convergence guarantees, they also have low iteration costs, ideal in large scale problems. Consequently the question of accelerating first order methods while keeping their small iteration costs have received much attention, see e.g. [33]. Since machine learning solutions are typically derived minimizing an empirical objective (the training error), most theoretical studies have focused on the error estimated for this latter quantity. However, it has recently become clear that optimization can play a key role from a statistical point of view when the goal is to minimize the expected (test) error. On the one hand, iterative optimization implicitly bias the search for a solution, e.g. converging to suitable minimal norm solutions [27]. On the other hand, the number of iterations parameterize paths of solutions of different complexity [31].

The idea that optimization can implicitly perform regularization has a long history. In the context of linear inverse problems, it is known as *iterative regularization* [11]. It is also an old trick for training neural networks where it is called early stopping [15]. The question of understanding the generalization properties of deep learning applications has recently sparked a lot of attention on this approach, which has been referred to as implicit regularization, see e.g. [13]. Establishing the regularization properties of iterative optimization requires the study of the corresponding expected error by combining optimization and statistical tools. First results in this sense focused on linear least squares with gradient descent and go back to [6, 31], see also [25] and references therein for improvements. Subsequent works have started considering other loss functions [16], multi-linear models [13] and other optimization methods, e.g. stochastic approaches [26, 18, 14].

In this paper, we consider the implicit regularization properties of acceleration. We focus on linear least squares in Hilbert space, because this setting allows to derive sharp results and working in infinite dimension magnify the role of regularization. Unlike in finite dimension learning bounds are

possible only if some form of regularization is considered. In particular, we consider two of the most popular accelerated gradient approaches, based on Nesterov acceleration [22] and (a variant of) the heavy-ball method [24]. Both methods achieve acceleration by exploiting a so called momentum term, which uses not only the previous, but the previous two iterations at each step. Considering a suitable bias-variance decomposition, our results show that accelerated methods have a behavior qualitatively different from basic gradient descent. While the bias decays faster with the number of iterations, the variance increases faster too. The two effect balance out, showing that accelerated methods achieve the same optimal statistical accuracy of gradient descent but they can indeed do this with less computations. Our analysis takes advantage of the linear structures induced by least squares to exploit tools from spectral theory. Indeed, the characterization of both convergence and stability rely on the study of suitable spectral polynomials defined by the iterates. While the idea that accelerated methods can be more unstable, this has been pointed out in [10] in a pure optimization context. Our results quantify this effect from a statistical point of view. Close to our results is the study in [9], where a stability approach is considered to analyze gradient methods for different loss functions [5].

2 Learning with (accelerated) gradient methods

Let the input space $\mathcal{X}$ be a separable Hilbert space (with scalar product $\langle \cdot, \cdot \rangle$ and induced norm $\|\cdot\|$) and the output space be $\mathbb{R}$ ¹. Let ρ be a unknown probability measure on the input-output space $\mathcal{X} \times \mathbb{R}$, $\rho_{\mathcal{X}}$ the induced marginal probability on $\mathcal{X}$, and $\rho(\cdot|x)$ the conditional probability measure on $\mathbb{R}$ given $x \in \mathcal{X}$. We make the following standard assumption: there exist $\kappa > 0$ such that

$$\langle x, x' \rangle \leq \kappa^2 \quad \forall x, x' \in \mathcal{X}, \quad \rho_{\mathcal{X}}\text{-almost surely.} \quad (1)$$

The goal of least-squares linear regression is to solve the *expected risk minimization* problem

$$\inf_{w \in \mathcal{X}} \mathcal{E}(w), \quad \mathcal{E}(w) = \int_{\mathcal{X} \times \mathbb{R}} (\langle w, x \rangle - y)^2 d\rho(x, y), \quad (2)$$

where ρ is known only through the n i.i.d. samples $(x_1, y_1), \dots, (x_n, y_n)$. In the following, we measure the quality of an approximate solution $\hat{w}$ with the excess risk

$$\mathcal{E}(\hat{w}) - \inf_{\mathcal{X}} \mathcal{E}.$$

The search of a solution is often based on replacing (2) with the empirical risk minimization (ERM)

$$\min_{w \in \mathcal{X}} \hat{\mathcal{E}}(w), \quad \hat{\mathcal{E}}(w) = \frac{1}{n} \sum_{i=1}^n (\langle w, x_i \rangle - y_i)^2. \quad (3)$$

For least squares an ERM solution can be computed in closed form using a direct solver. However, for large problems, iterative solvers are preferable and we next describe the approaches we consider.

First, it is useful to rewrite the ERM with vectors notation. Let $\mathbf{y} \in \mathbb{R}^n$ with $(\mathbf{y})_i = y_i$ and $X : \mathcal{X} \rightarrow \mathbb{R}^n$ s.t. $(Xw)_i = \langle w, x_i \rangle$ for $i = 1 \dots, n$. Here the norm $\|\cdot\|_n$ is norm in $\mathbb{R}^n$ multiplied by $1/\sqrt{n}$. Let $X^* : \mathbb{R}^n \rightarrow \mathcal{X}$ be the adjoint of X defined by $X^* \mathbf{y} = \frac{1}{n} \sum_{i=1}^n x_i y_i$. Then, ERM becomes

$$\min_{w \in \mathcal{X}} \hat{\mathcal{E}}(w) = \|Xw - \mathbf{y}\|_n^2. \quad (4)$$

2.1 Gradient descent and accelerated methods

Gradient descent serves as a reference approach throughout the paper. For problem (4) it becomes

$$\hat{w}_{t+1} = \hat{w}_t - \alpha X^* (X \hat{w}_t - \mathbf{y}) \quad (5)$$

with initial point $\hat{w}_0 = 0$ and the step-size α that satisfy $\alpha < \frac{1}{\kappa^2}$ ². The progress made by gradient descent at each iteration can be slow and the idea behind acceleration is to use the information of the previous directions in order to improves the convergence rate of the algorithm.

¹As shown in Appendix this choice allows to recover nonparametric kernel learning as a special case.

² The step-size α is the step-size at the t -th iteration and satisfies the condition $0 < \alpha \|X\|_{op}^2 < 1$, where $\|\cdot\|_{op}$ denotes the operatorial norm. Since the operator X is bounded by κ (which means $\|X\|_{op} \leq \kappa$) it is sufficient to assume $\alpha < \frac{1}{\kappa^2}$.

Heavy-ball

Heavy-ball is a popular accelerated method that adds the *momentum* $\hat{w}_t - \hat{w}_{t-1}$ at each iteration

$$\hat{w}_{t+1} = \hat{w}_t - \alpha X^* (X \hat{w}_t - \mathbf{y}) + \beta(\hat{w}_t - \hat{w}_{t-1}) \quad (6)$$

with $\alpha, \beta \geq 0$, the case $\beta = 0$ reduces to gradient descent. In the quadratic case we consider it is also called Chebyshev iterative method. The optimization properties of heavy-ball have been studied extensively [24, 32]. Here, we consider the following variant. Let $\nu > 1$, consider the varying parameter heavy-ball replacing α, β in (6) with $\alpha_{t+1}, \beta_{t+1}$ defined as:

$$\alpha_t = \frac{4}{\kappa^2} \frac{(2t + 2\nu - 1)(t + \nu - 1)}{(t + 2\nu - 1)(2t + 4\nu - 1)} \quad \beta_t = \frac{(t - 1)(2t - 3)(2t + 2\nu - 1)}{(t + 2\nu - 1)(2t + 4\nu - 1)(2t + 2\nu - 3)},$$

for $t > 0$ and with initialization $\hat{w}_{-1} = \hat{w}_0 = 0$, $\alpha_1 = \frac{1}{\kappa^2} \frac{4\nu+2}{4\nu+1}$, $\beta_1 = 0$. With this choice and considering the least-squares problem this algorithm is known as ν -method in the inverse problem literature (see e.g. [11]). This seemingly complex parameters' choice allows to relates the approach to suitable orthogonal polynomials recursion as we discuss later.

Nesterov acceleration

The second form of gradient acceleration we consider is the popular Nesterov acceleration [22]. In our setting, it corresponds to the iteration

$$\hat{w}_{t+1} = \hat{v}_t - \alpha X^* (X \hat{v}_t - \mathbf{y}), \quad \hat{v}_t = \hat{w}_t + \beta_t (\hat{w}_t - \hat{w}_{t-1}) \quad (7)$$

with the two initial points $\hat{w}_{-1} = \hat{w}_0 = 0$, and the sequence β_t chosen as

$$\beta_t = \frac{t - 1}{t + \beta}, \quad \beta \geq 1. \quad (8)$$

Differently from heavy-ball, Nesterov acceleration uses the momentum term also in the evaluation of the gradient. Also in this case optimization results are well known [1, 29].

Here, as above, optimization results refer to solving ERM (3), (4), whereas in the following we study to which extent the above iterations can be used to minimize the expected error (2). In the next section, we discuss a spectral approach which will be instrumental towards this goal.

3 Spectral filtering for accelerated methods

Least squares allows to consider spectral approaches to study the properties of gradient methods for learning. We illustrate these ideas for gradient descent before considering accelerated methods.

Gradient descent as spectral filtering

Note that by a simple (and classical) induction argument, gradient descent can be written as

$$\hat{w}_t = \alpha \sum_{j=0}^{t-1} (I - \alpha \hat{\Sigma})^j X^* \mathbf{y}.$$

Equivalently using spectral calculus

$$\hat{w}_t = g_t(\hat{\Sigma}) X^* \mathbf{y}, \quad \text{with } \hat{\Sigma} = X^* X,$$

where g_t are the polynomials $g_t(\sigma) = \alpha \sum_{j=0}^{t-1} (I - \alpha \sigma)^j$ for all $\sigma \in (0, \kappa^2]$ and $t \in \mathbb{N}$. Note that, the polynomials g_t are bounded by αt . A first observation is that $g_t(\sigma)\sigma$ converges to 1 as $t \rightarrow \infty$, since $g_t(\sigma)$ converges to $\frac{1}{\sigma}$. A second observation is that the *residual polynomials* $r_t(\sigma) = 1 - \sigma g_t(\sigma)$, which are all bounded by 1, control ERM convergence since,

$$\|X \hat{w}_t - \mathbf{y}\|_n = \|X g_t(\hat{\Sigma}) X^* \mathbf{y} - \mathbf{y}\|_n = \|g_t(\hat{\Sigma}) \hat{\Sigma} \mathbf{y} - \mathbf{y}\|_n = \|r_t(\hat{\Sigma}) \mathbf{y}\|_n \leq \|r_t(\hat{\Sigma})\|_{op} \|\mathbf{y}\|_n.$$

In particular, if $\mathbf{y}$ is in the range of $\hat{\Sigma}^r$ for some $r > 0$ (source condition on $\mathbf{y}$) improved convergence rates can be derived noting that by an easy calculation

$$|r_t(\sigma)\sigma^q| \leq \left(\frac{q}{\alpha}\right)^q \left(\frac{1}{t}\right)^q.$$

As we show in Section 4, considering the polynomials g_t and r_t allows to study not only ERM but also expected risk minimization (2), by relating gradient methods to their infinite sample limit. Further, we show how similar reasoning hold for accelerated methods. In order to do so, it useful to first define the characterizing properties of g_t and r_t .

3.1 Spectral filtering

The following definition abstracts the key properties of the function g_t and r_t often called spectral filtering function [2]. Following the classical definition we replace t with a generic parameter λ .

Definition 1.

The family $\{g_\lambda\}_{\lambda \in (0,1]}$ is called *spectral filtering function* if the following conditions hold:

- (i) There exist a constant $E < +\infty$ such that, for any $\lambda \in (0, 1]$

$$\sup_{\sigma \in (0, \kappa^2]} |g_\lambda(\sigma)| \leq \frac{E}{\lambda}. \quad (9)$$

- (ii) Let $r_\lambda(\sigma) = 1 - \sigma g_\lambda(\sigma)$ there exist a constant F_0 such that, for any $\lambda \in (0, 1]$

$$\sup_{\sigma \in (0, \kappa^2]} |r_\lambda(\sigma)| \leq F_0. \quad (10)$$

Definition 2. (Qualification)

The qualification of the spectral filtering function $\{g_\lambda\}_\lambda$ is the maximum parameter q such that for any $\lambda \in (0, 1]$ there exist a constant F_q such that

$$\sup_{\sigma \in (0, \kappa^2]} |r_\lambda(\sigma)\sigma^q| \leq F_q \lambda^q. \quad (11)$$

Moreover we say that a filtering function has qualification ∞ if (11) holds for every $q > 0$.

Methods with finite qualification might have slow convergence rates in certain regimes. The smallest the qualification the worse the rates can be.

The discussion in the previous section shows that gradient descent defines a spectral filtering function where $\lambda = 1/t$. More precisely, the following holds.

Proposition 1. Assume $\lambda = \frac{1}{t}$ for $t \in \mathbb{N}$, then the polynomials g_t related to the gradient descent iterates, defined in (5), are a filtering function with parameters $E = \alpha$ and $F_0 = 1$. Moreover it has qualification ∞ with parameters $F_q = (q/\alpha)^q$.

The above result is classical and we report a proof in the appendix for completeness. Next, we discuss analogous results for accelerate methods and then compare the different spectral filtering functions.

3.2 Spectral filtering for accelerated methods

For the heavy-ball (6) the following result holds

Proposition 2. Assume $\kappa \leq 1$, let $\nu > 0$ and $\lambda = \frac{1}{t^2}$ for $t \in \mathbb{N}$, then the polynomials g_t related to heavy-ball method (6) are a filtering function with parameters $E = 2$ and $F_0 = 1$. Moreover there exist a positive constant $c_\nu < +\infty$ such that the ν -method has qualification ν .

The proof of the above proposition follows combining several intermediate results from [11]. The key idea is to show that the residual polynomials defined by heavy-ball iteration form a sequence of orthogonal polynomials with respect to the weight function

$$\omega_\nu(\sigma) = \frac{\sigma^{2\nu}}{\sigma^{\frac{1}{2}}(1-\sigma)^{\frac{1}{2}}},$$

which is a so called shifted Jacobi weight. Results from orthogonal polynomials can then be used to characterize the corresponding spectral filtering function.

The following proposition considers Nesterov acceleration.

Proposition 3. Assume $\lambda = 1/t^2$, then the polynomials g_t related to Nesterov iterates (7) are a filtering function with constants $E = 2\alpha$ and $F_0 = 1$. Moreover the qualification of this method is at least $1/2$ with constants $F_q = \left(\frac{\beta^2}{\alpha}\right)^q$.

Filtering properties of the Nesterov iteration (7) have been studied recently in the context of inverse problems [23]. In the appendix 7.3 we provide a simplified proof based on studying the properties of suitable discrete dynamical systems defined by the Nesterov iteration (7).

3.3 Comparing the different filter functions

We summarize the properties of the spectral filtering function of the various methods for $\kappa = 1$.

Method	E	F ₀	F _q	Qualification
Gradient descent	1	1	q^q	∞
Heavy-ball	2	1	$c_\nu \quad (q = \nu)$	ν
Nesterov	2	1	β^{2q}	$\geq 1/2$

The main observation is that the properties of the spectral filtering functions corresponding to the different iterations depend on $\lambda = 1/t$ for gradient descent, but on $\lambda = 1/t^2$ for the accelerated methods. As we see in the next section this leads to substantially different learning properties. Further we can see that gradient descent is the only algorithm with qualification ∞ , even if the parameter $F_q = q^q$ can be very large. The accelerated methods seem to have smaller qualification. In particular, the heavy-ball method can attain a high qualification, depending on ν , but the constant c_ν is unknown and could be large. For Nesterov accelerated method, the qualification is at least $1/2$ and it's an open question whether this bound is tight or higher qualification can be attained.

In the next section, we show how the properties of the spectral filtering functions can be exploited to study the excess risk of the corresponding iterations.

4 Learning properties for accelerated methods

We first consider a basic scenario and then a more refined analysis leading to a more general setting and potentially faster learning rates.

4.1 Attainable case

Consider the following basic assumption.

Assumption 1. Assume there exist $M > 0$ such that $|y| < M$ ρ -almost surely and $w^* \in \mathcal{X}$ such that $\mathcal{E}(w^*) = \inf_{\mathcal{X}} \mathcal{E}$.

Then the following result can be derived.

Theorem 1. Under Assumption 1, let $\hat{w}_t^{GD}$ and $\hat{w}_t^{acc}$ be the t -th iterations respectively of gradient descent (5) and an accelerated version given by (6) or (7). Assuming the sample-size n to be large enough and let $\delta \in (0, 1/2)$ then there exist two positive constant C_1 and C_2 such that with probability at least $1 - \delta$

$$\begin{aligned} \mathcal{E}(\hat{w}_t^{GD}) - \inf_{\mathcal{H}} \mathcal{E} &\leq C_1 \left(\frac{1}{t} + \frac{t}{n} \right) \log^2 \frac{2}{\delta} \\ \mathcal{E}(\hat{w}_t^{acc}) - \inf_{\mathcal{H}} \mathcal{E} &\leq C_2 \left(\frac{1}{t^2} + \frac{t^2}{n} \right) \log^2 \frac{2}{\delta}. \end{aligned}$$

where the constants C_1 and C_2 do not depend on n, t, δ , but depend on the chosen optimization method.

Moreover by choosing the stopping rules $t^{GD} = O(n^{1/2})$ and $t^{acc} = O(n^{1/4})$ both algorithms have learning rate of order $\frac{1}{\sqrt{n}}$.

The proof of the above results is given in the appendix and the novel part is the one concerning accelerated methods, particularly Nesterov acceleration. The result shows how the number of iteration

controls the learning properties both for gradient descent and accelerated gradient. In this sense implicit regularization occurs in all these approaches. For any t the error is split in two contributions. Inspecting the proof it is easy to see that, the first term in the bound comes from the convergence properties of the algorithm with infinite data. Hence the optimization error translates into a bias term. The decay for accelerated method is much faster than for gradient descent. The second term arises from comparing the empirical iterates with their infinite sample (population) limit. It is a variance term depending on the sampling in the data and hence decreases with the sample size. For all methods, this term increases with the number of iterations, indicating that the empirical and population iterations are increasingly different. However, the behavior is markedly worse for accelerated methods. The benefit of acceleration seems to be balanced out by this more unstable behavior. In fact, the benefit of acceleration is apparent balancing the error terms to obtain a final bound. The obtained bound is the same for gradient descent and accelerated methods, and is indeed optimal since it matches corresponding lower bounds [3, 7]. However, the number of iterations needed by accelerated methods is the square root of those needed by gradient descent, indicating a substantial computational gain can be attained. Next we show how these results can be generalized to a more general setting, considering both weaker and stronger assumptions, corresponding to harder or easier learning problems.

4.2 More refined result

Theorem 1 is a simplified version of the more general result that we discuss in this section. We are interested in covering also the non-attainable case, that is when there is no $w^* \in \mathcal{X}$ such that $\mathcal{E}(w^*) = \inf_{\mathcal{X}} \mathcal{E}$. In order to cover this case we have to introduce several more definitions and notations. In Appendix 8.2 we give a more detailed description of the general setting. Consider the space $L^2_{\rho_{\mathcal{X}}}$ of the square integrable functions with the norm $\|f\|_{\rho_{\mathcal{X}}}^2 = \int_{\mathcal{X}} f(x)^2 d\rho_{\mathcal{X}}(x)$ and extend the expected risk to $L^2_{\rho_{\mathcal{X}}}$ defining $\mathcal{E}(f) = \int_{\mathcal{X} \times \mathbb{R}} (f(x) - y)^2 d\rho(x, y)$. Let $\mathcal{H} \subseteq L^2_{\rho_{\mathcal{X}}}$ be the hypothesis space of functions such that $f(x) = \langle w, x \rangle \rho_{\mathcal{X}}$ almost surely. Recall that, the minimizer of the expected risk over $L^2_{\rho_{\mathcal{X}}}$ is the regression function $f_{\rho} = \int_{\mathcal{X}} y d\rho(y|x)$. The projection $f_{\mathcal{H}}$ over the closure of the hypothesis space $\mathcal{H}$ is defined as

$$f_{\mathcal{H}} = \arg \min_{g \in \overline{\mathcal{H}}} \|g - f_{\rho}\|_{\rho_{\mathcal{X}}} .$$

Let $L : L^2_{\rho_{\mathcal{X}}} \rightarrow L^2_{\rho_{\mathcal{X}}}$ be the integral operator

$$Lf(x) = \int_{\mathcal{X}} f(x') \langle x, x' \rangle d\rho_{\mathcal{X}}(x') .$$

The first assumption we consider concern the moments of the output variable and is more general than assuming the output variable y to be bounded as assumed before.

Assumption 2. *There exist positive constant Q and M such that for all $\mathbb{N} \ni l \geq 2$,*

$$\int_{\mathbb{R}} |y|^l d\rho(y|x) \leq \frac{1}{2} l! M^{l-2} Q^2 \quad \rho_{\mathcal{X}} \text{ almost surely.}$$

This assumption is standard and satisfied in classification or regression with well behaved noise. Under this assumption the regression function f_{ρ} is bounded almost surely

$$|f_{\rho}(x)| \leq \int_{\mathbb{R}} |y| d\rho(y|x) \leq \left(\int_{\mathbb{R}} |y|^2 d\rho(y|x) \right)^{1/2} \leq Q . \quad (12)$$

The next assumptions are related to the regularity of the target function $f_{\mathcal{H}}$.

Assumption 3.

There exist a positive constant B such that the target function $f_{\mathcal{H}}$ satisfy

$$\int_{\mathcal{X}} (f_{\mathcal{H}}(x) - f_{\rho}(x))^2 x \otimes x d\rho_{\mathcal{X}}(x) \preceq B^2 \Sigma .$$

This assumption is needed to deal with the misspecification of the model. The last assumptions quantify the regularity of $f_{\mathcal{H}}$ and the size (capacity) of the space $\mathcal{H}$.

Assumption 4.

There exist $g_0 \in L_{\rho_X}^2$ and $r > 0$ such that

$$f_{\mathcal{H}} = L^r g_0, \quad \text{with } \|g_0\|_{\rho_X} \leq R.$$

Moreover we assume that there exist $\gamma \geq 1$ and a positive constant c_γ such that the effective dimension

$$\mathcal{N}(\lambda) = \text{Tr} \left(L (L + \lambda I)^{-1} \right) \leq c_\gamma \lambda^{-\frac{1}{\gamma}}.$$

The assumption on $\mathcal{N}(\lambda)$ is always true for $\gamma = 1$ and $c_1 = \kappa^2$ and it's satisfied when the eigenvalues σ_i of L decay as $i^{-\gamma}$. We recall that, the space $\mathcal{H}$ can be characterized in terms of the operator L , indeed

$$\mathcal{H} = L^{1/2} (L_{\rho_X}^2).$$

Hence, the non-attainable corresponds to considering $r < 1/2$.

Theorem 2. Under Assumption 2, 3, 4, let $\hat{w}_t^{GD}$ and $\hat{w}_t^{acc}$ be the t -th iterations of gradient descent (5) and an accelerated version given by (6) or (7) respectively. Assuming the sample-size n to be large enough, let $\delta \in (0, 1/2)$ and assuming r to be smaller than the qualification of the considered algorithm (and equal to $1/2$ in the case of Nesterov accelerated methods), then there exist two positive constant C_1 and C_2 such that with probability at least $1 - \delta$

$$\begin{aligned} \mathcal{E}(\hat{w}_t^{GD}) - \inf_{\mathcal{H}} \mathcal{E} &\leq C_1 \left(\frac{1}{t^{2r}} + \frac{t^{\frac{1}{\gamma}}}{n} \right) \log^2 \frac{2}{\delta} \\ \mathcal{E}(\hat{w}_t^{acc}) - \inf_{\mathcal{H}} \mathcal{E} &\leq C_2 \left(\frac{1}{t^{4r}} + \frac{t^{\frac{2}{\gamma}}}{n} \right) \log^2 \frac{2}{\delta}. \end{aligned}$$

where the constants C_1 and C_2 do not depend on n, t, δ , but depend on the chosen optimization.

Choosing the stopping rules $t^{GD} = O(n^{\frac{\gamma}{2\gamma r+1}})$ and $t^{acc} = O(n^{\frac{\gamma}{4\gamma r+2}})$ both gradient descent and accelerated methods achieve a learning rate of order $O\left(n^{\frac{-2\gamma r}{2\gamma r+1}}\right)$.

The only reason why we do not consider $r < 1/2$ in the analysis of Nesterov accelerated methods is that our proof require the qualification of the method to be larger than 1 for technical reasons. However we think that our result can be extended to that case, furthermore we think Nesterov qualification to be larger than 1, however it's an open question whether higher qualification can be attained. The proof of the above result is given in the appendix. The general structure of the bound is the same as in the basic setting, which is now recovered as a special case. However, in this more general form, the various terms in the bound depend now on the regularity assumptions on the problem. In particular, the variance depends on the effective dimension behavior, e.g. on the eigenvalue decay, while the bias depend on the regularity assumption on $f_{\mathcal{H}}$. The general comparison between gradient descent and accelerated methods follows the same line as in the previous section. Faster bias decay of accelerated methods is contrasted by a more unstable behavior. As before, the benefit of accelerated methods becomes clear when deriving optimal stopping time and corresponding learning bound: they achieve the accuracy of gradient methods but in considerable less time. While heavy-ball and Nesterov have again similar behaviors, here a subtle difference resides in their different qualifications, which in principle lead to different behavior for easy problems, that is for large r and γ . In this regime, gradient descent could work better since it has infinite qualification. For problems in which $r < 1/2$ and $\gamma = 1$ the rates are worse than in the basic setting, hence these problems are hard.

4.3 Related work

In the convex optimization framework a similar phenomenon was pointed out in [10] where they introduce the notion of inexact first-order oracle and study the behaviour of several first-order methods of smooth convex optimization with such oracle. In particular they show that the superiority of accelerated methods over standard gradient descent is no longer absolute when an inexact oracle is used. This because acceleration suffer from the accumulation of the errors committed by the inexact oracle. A relevant result on the generalization properties of learning algorithm is [5] in which they introduce the notion of *uniform* stability and use it to obtain generalization error bounds

for regularization based learning algorithms. Recently, to show the effectiveness of commonly used optimization algorithms in many large-scale learning problems, algorithmic stability has been established for stochastic gradient methods [14], as well as for any algorithm in situations where global minima are approximately achieved [8]. For Nesterov’s accelerated gradient descent and heavy-ball method, [9] provide stability upper bounds for quadratic loss function in a finite dimensional setting. All these approaches focus on the definition of uniform stability given in [5]. Our approach to the stability of a learning algorithm is based on the study of filtering properties of accelerated methods together with concentration inequalities, we obtain upper bounds on the generalization error for quadratic loss in a infinite dimensional Hilbert space and generalize the bounds obtained in [9] by considering different regularity assumptions and by relaxing the hypothesis of the existence of a minimizer of the expected risk on the hypothesis space.

5 Numerical simulation

In this section we show some numerical simulations to validate our results. We want to simulate the case in which the eigenvalues σ_i of the operator L are $\sigma_i = i^{-\gamma}$ for some $\gamma \leq 1$ and the non-attainable case $r < 1/2$. In order to do this we observe that if we consider the kernel setting over a finite space $\mathcal{Z} = \{z_1, \dots, z_n\}$ of size N with the uniform probability distribution $\rho_{\mathcal{Z}}$, then the space $L^2(\mathcal{Z}, \rho_{\mathcal{Z}})$ becomes $\mathbb{R}^N$ with the usual scalar product multiplied by $1/N$. the operator L becomes a $N \times N$ matrix which entries are $L_{i,j} = K(z_i, z_j)$ for every $i, j \in \{1, \dots, N\}$, where K is the kernel, which is fixed by the choice of the matrix L . We build the matrix $L = UDU^T$ with $U \in \mathbb{R}^{N \times N}$ orthogonal matrix and D diagonal matrix with entries $D_{i,i} = i^{-\gamma}$. The source condition becomes $f_{\mathcal{H}} = L^r g_0$ for some $g_0 \in \mathbb{R}^N, r > 0$. We simulate the observed output as $y = f_{\mathcal{H}} + \mathcal{N}(0, \sigma)$ where $\mathcal{N}(0, \sigma)$ is the standard normal distribution of variance σ^2 . The sampling operation can be seen as extracting n indices $i_1, \dots, i_n$ and building the kernel matrix $\hat{K}_{j,k} = K(z_{i_j}, z_{i_k})$ and the noisy labels $\hat{y}_j = y_{i_j}$ for every $j, k \in \{1, \dots, n\}$. The Representer Theorem ensure that we can built our estimator $\hat{f} \in \mathbb{R}^N$ as $\hat{f}(z) = \sum_{j=1}^n K(z, z_{i_j}) c_j$ where the vector c depends on the chosen optimization algorithm and takes the form $c = g_t(\hat{K})y$. The excess risk of the estimator $\hat{f}$ is given by $\|\hat{f} - f_{\mathcal{H}}\|_{L_{\mathcal{Z}}^2}^2$.

For every algorithm considered, we run 50 repetitions, in which we sample the data-space and compute the error $\|\hat{f}_t - f_{\mathcal{H}}\|_{L_{\mathcal{Z}}^2}^2$, where $\hat{f}_t$ represents the estimator related to the t -th iteration of one of the considered algorithms, and in the end we compute the mean and the variance of those errors. In Figure 1 we simulate the error of all the algorithms considered for both attainable and non-attainable case. We observe that both heavy-ball and Nesterov acceleration provides faster convergence rates with respect to gradient descent method, but the learning accuracy is not improved. We observe that the accelerated methods considered show similar behavior and that for “easy problem” (large r) that gradient descent can exploits its higher qualification and perform similarly to the accelerated methods.

In Figure 2 we show the test error related to the real dataset *pumadyn8nh* (available at <https://www.dcc.fc.up.pt/ltorgo/Regression/puma.html>). Even in this case we can observe the behaviors shown in our theoretical results.

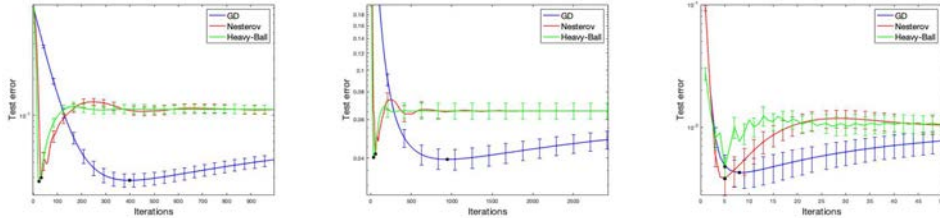


Fig. 1: Mean and variance of error $\|\hat{f}_t - f_{\mathcal{H}}\|_N^2$ for the t -th iteration of gradient descent (GD), Nesterov accelerated algorithm and heavy-ball ($\nu = 1$). Black dots shows the absolute minimum of the curves. The parameters are chosen $N = 10^4, n = 10^2, \gamma = 1, \sigma = 0.5$. We show the attainable case ($r = 1/2$) in the left, the “hard case” ($r = 0.1 < 1/2$) in the center and the “easy case” ($r > 1/2$) in the right.

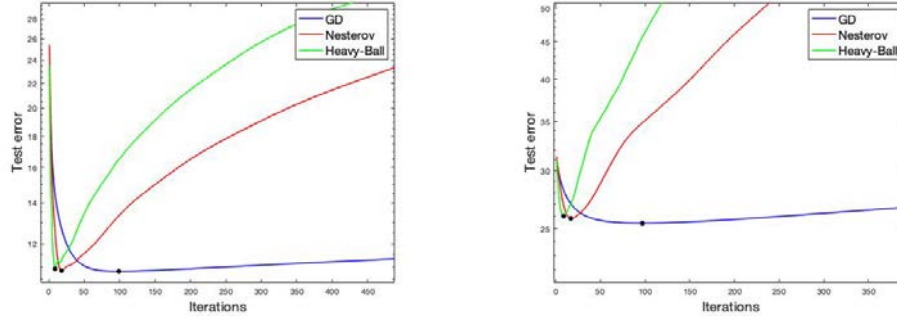


Fig. 2: Test error on the real dataset *pumadyn8nh* using gradient descent (GD), Nesterov accelerated algorithm and heavy-ball. In the left we use a gaussian kernel with $\sigma = 1.2$ and in the right a polynomial kernel of degree 9.

6 Conclusion

In this paper, we have considered the implicit regularization properties of accelerated gradient methods for least squares in Hilbert space. Using spectral calculus we have characterized the properties of the different iterations in terms of suitable polynomials. Using the latter, we have derived error bounds in terms of suitable bias and variance terms. The main conclusion is that under the considered assumptions accelerated methods have smaller bias but also larger variance. As a byproduct they achieve the same accuracy of vanilla gradient descent but with much fewer iterations. Our study opens a number of potential theoretical and empirical research directions. From a theory point of view, it would be interesting to consider other learning regimes, for examples classification problems, different loss functions or other regularity assumptions beyond classical nonparametric assumptions, e.g. misspecified models and fast eigenvalues decays (Gaussian kernel). From an empirical point of view it would be interesting to do a more thorough investigation on a larger number of simulated and real data-sets of varying dimension.

Acknowledgments

This material is based upon work supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216, and the Italian Institute of Technology. We gratefully acknowledge the support of NVIDIA Corporation for the donation of the Titan Xp GPUs and the Tesla k40 GPU used for this research. L. R. acknowledges the financial support of the AFOSR projects FA9550-17-1-0390 and BAA-AFRL-AFOSR-2016-0007 (European Office of Aerospace Research and Development), and the EU H2020-MSCA-RISE project NoMADS - DLV-777826. N.P. would like to thank Murata Tomoya for the useful observations.

References

- [1] Hedy Attouch, Zaki Chbani, and Hassan Riahi. Rate of convergence of the nesterov accelerated gradient method in the subcritical case $\alpha \leq 3$. *ESAIM: Control, Optimisation and Calculus of Variations*, 25:2, 2019.
- [2] Luca Baldassarre, Lorenzo Rosasco, Annalisa Barla, and Alessandro Verri. Multi-output learning via spectral filtering. *Machine learning*, 87(3):259–301, 2012.
- [3] Gilles Blanchard and Nicole Mücke. Optimal rates for regularization of statistical inverse learning problems. *Foundations of Computational Mathematics*, 18(4):971–1013, Aug 2018.
- [4] Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Advances in neural information processing systems*, pages 161–168, 2008.
- [5] Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of machine learning research*, 2(Mar):499–526, 2002.

- [6] Peter Bühlmann and Bin Yu. Boosting with the l_2 loss: regression and classification. Journal of the American Statistical Association, 98(462):324–339, 2003.
- [7] Andrea Caponnetto and Ernesto De Vito. Optimal rates for the regularized least-squares algorithm. Foundations of Computational Mathematics, 7(3):331–368, 2007.
- [8] Zachary Charles and Dimitris Papailiopoulos. Stability and generalization of learning algorithms that converge to global optima. arXiv preprint arXiv:1710.08402, 2017.
- [9] Yuansi Chen, Chi Jin, and Bin Yu. Stability and convergence trade-off of iterative optimization algorithms. arXiv preprint arXiv:1804.01619, 2018.
- [10] Olivier Devolder, François Glineur, and Yurii Nesterov. First-order methods of smooth convex optimization with inexact oracle. Mathematical Programming, 146(1-2):37–75, 2014.
- [11] Heinz Werner Engl, Martin Hanke, and Andreas Neubauer. Regularization of inverse problems, volume 375. Springer Science & Business Media, 1996.
- [12] Junichi Fujii, Masatoshi Fujii, Takayuki Furuta, and Ritsuo Nakamoto. Norm inequalities equivalent to heinz inequality. Proceedings of the American Mathematical Society, 118(3):827–830, 1993.
- [13] Suriya Gunasekar, Jason D Lee, Daniel Soudry, and Nati Srebro. Implicit bias of gradient descent on linear convolutional networks. In Advances in Neural Information Processing Systems, pages 9461–9471, 2018.
- [14] Moritz Hardt, Benjamin Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. arXiv preprint arXiv:1509.01240, 2015.
- [15] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In Neural networks: Tricks of the trade, pages 9–48. Springer, 2012.
- [16] Junhong Lin, Raffaello Camoriano, and Lorenzo Rosasco. Generalization properties and implicit regularization for multiple passes sgm. In International Conference on Machine Learning, pages 2340–2348, 2016.
- [17] Junhong Lin and Volkan Cevher. Optimal convergence for distributed learning with stochastic gradient methods and spectral-regularization algorithms. stat, 1050:22, 2018.
- [18] Junhong Lin and Lorenzo Rosasco. Optimal learning for multi-pass stochastic gradient methods. In Advances in Neural Information Processing Systems, pages 4556–4564, 2016.
- [19] Junhong Lin, Alessandro Rudi, Lorenzo Rosasco, and Volkan Cevher. Optimal rates for spectral algorithms with least-squares regression over hilbert spaces. Applied and Computational Harmonic Analysis, 2018.
- [20] Peter Mathé and Sergei Pereverzev. Regularization of some linear ill-posed problems with discretized random noisy data. Mathematics of Computation, 75(256):1913–1929, 2006.
- [21] Peter Mathé and Sergei V Pereverzev. Moduli of continuity for operator valued functions. Numerical Functional Analysis and Optimization, 23(5-6):623–631, 2002.
- [22] Yurii E Nesterov. A method for solving the convex programming problem with convergence rate $o(1/k^2)$. In Dokl. akad. nauk Sssr, volume 269, pages 543–547, 1983.
- [23] Andreas Neubauer. On nesterov acceleration for landweber iteration of linear ill-posed problems. Journal of Inverse and Ill-posed Problems, 25(3):381–390, 2017.
- [24] Boris T Polyak. Introduction to optimization. Technical report, 1987.

- [25] Garvesh Raskutti, Martin J Wainwright, and Bin Yu. Early stopping and non-parametric regression: an optimal data-dependent stopping rule. The Journal of Machine Learning Research, 15(1):335–366, 2014.
- [26] Lorenzo Rosasco and Silvia Villa. Learning with incremental iterative regularization. In Advances in Neural Information Processing Systems, pages 1630–1638, 2015.
- [27] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. The Journal of Machine Learning Research, 19(1):2822–2878, 2018.
- [28] Ingo Steinwart and Andreas Christmann. Support vector machines. Springer Science & Business Media, 2008.
- [29] Weijie Su, Stephen Boyd, and Emmanuel Candes. A differential equation for modeling nesterov’s accelerated gradient method: Theory and insights. In Advances in Neural Information Processing Systems, pages 2510–2518, 2014.
- [30] Gabor Szeg. Orthogonal polynomials, volume 23. American Mathematical Soc., 1939.
- [31] Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. On early stopping in gradient descent learning. Constructive Approximation, 26(2):289–315, 2007.
- [32] SK Zavriev and FV Kostyuk. Heavy-ball method in nonconvex optimization problems. Computational Mathematics and Modeling, 4(4):336–341, 1993.
- [33] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. arXiv preprint arXiv:1611.03530, 2016.

Interactive Data Collection for Deep Learning Object Detectors on Humanoid Robots

Elisa Maiettini^{1,2,3}, Giulia Pasquale^{1,2}, Lorenzo Rosasco^{1,2,3} and Lorenzo Natale¹

Abstract—Deep Learning (DL) methods are notoriously data hungry. Their adoption in robotics is challenging due to the cost associated with data acquisition and labeling. In this paper we focus on the problem of object detection, i.e. the simultaneous localization and recognition of objects in the scene, for which various DL architectures have been proposed in the literature. We propose to use an automatic annotation procedure, which leverages on human-robot interaction and depth-based segmentation, for the acquisition and labeling of training examples. We fine-tune the Faster R-CNN [36] network with these data acquired by the robot autonomously. We measure the performance on the same dataset and investigate the generalization abilities of the network on different settings and in absence of explicit segmentation, showing good detection performance. Experiments on the iCub humanoid robot [25] show that the proposed strategy is effective and can be used to deploy deep object detection algorithms on a robot.

I. INTRODUCTION

The ability to localize and recognize objects in the scene is crucial for autonomous robots to act in unconstrained environments [20]. Detecting objects from the visual input is essential, and often it represents the first step in the interaction of the robot with the environment.

While planning actions requires full pose estimation in 6D, a common approach adopted in the literature is to split the problem in different stages. The first step is the localization of objects in the image plane. In Computer Vision this task is called 2D object detection and consists in predicting the label and location (e.g., in the form of 2D bounding box coordinates) for each object represented in the image (see, e.g., [37]). In this work, we focus on this task, as a prerequisite for more complex operations in scene understanding.

We consider recent Deep Learning (DL) methods [21], [36], [39], motivated by the remarkable performance they obtain on difficult tasks such as the ImageNet Large-Scale Visual Recognition [37], the MS COCO [23] and Pascal VOC [12] challenges. One of the problems in the adoption of these methods in robotics, is that they require a large dataset of images carefully annotated. Image annotation is particularly demanding for training object detection systems, because this process requires not only object labels but also bounding boxes. In addition, it implies an off-line process, which is unfeasible for a system that learns on-line.

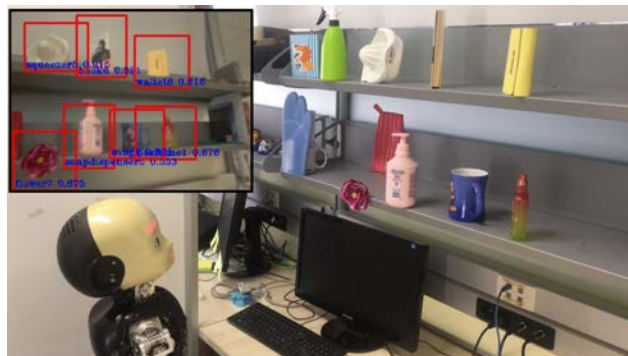


Fig. 1: The iCub robot observes some of the objects learned using the proposed pipeline and detects them on a shelf. A video showing the system running has been made available as Supplementary Material.

We propose an approach to train DL methods for object detection that overcomes the lack of manual annotations by exploiting the interaction with a human teacher. Learning happens in a natural, semi-controlled setting, in which objects are presented to the robot by the teacher and the problem of figure-ground segmentation is greatly simplified. This acquisition procedure was validated in our previous work [28], where we showed how this approach can be adopted to acquire large-scale annotated image datasets (i.e. the ICUBWORLD TRANSFORMATIONS¹). Some example images in the dataset are represented in Fig. 2; annotations are in terms of the label of the object and a surrounding bounding box computed with the depth estimation and segmentation procedure presented in [29].

The contribution of this work is to assess whether this approach can be adopted to effectively train deep object detectors as the recent Faster Region-CNN [36]. More specifically, we first investigate whether such models can learn to predict accurate bounding boxes from imperfect ground-truth. In fact, there are clearly multiple sources of noise in the automatic annotation procedure, which negatively affect the quality of supervision with respect to manual annotations.

In addition, our learning scenario is rather constrained: images depicts isolated objects, well separated from the background, and almost always centered in the visual field because the robot is tracking it with the eyes. The next question, is then, to investigate whether the learned detector can generalize to other, less constrained, scenarios, i.e. when multiple objects are present in the visual scene and figure-

¹ iCub Facility, Istituto Italiano di Tecnologia, Genoa, IT

² Laboratory for Computational and Statistical Learning, Istituto Italiano di Tecnologia and Massachusetts Institute of Technology, Cambridge, MA

³ Dipartimento di Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi, University of Genoa, Genoa, IT

¹<https://robotology.github.io/iCubWorld/>

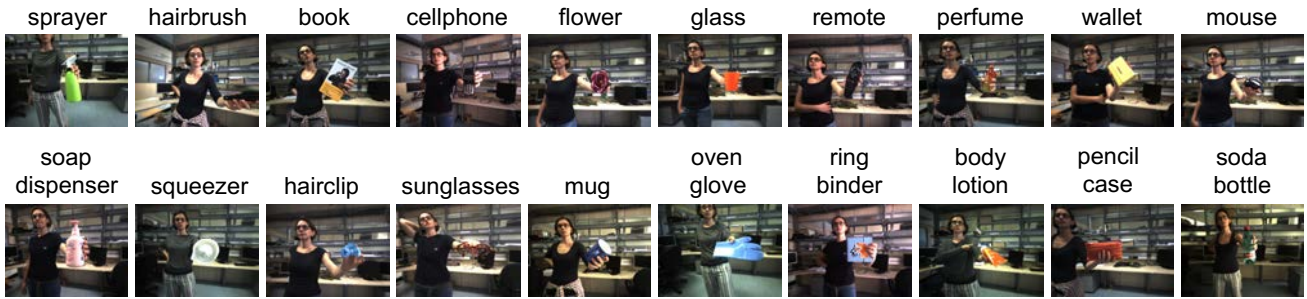


Fig. 2: Examples images from ICWT, depicting 20 objects shown to the robot by a human teacher.

ground segmentation with depth cues is more challenging.

In order to quantitatively assess the effectiveness of this procedure, we i) manually annotate a subset of the ICUB-WORLD TRANSFORMATIONS dataset, on which we measure the performance of the detectors trained with automatic annotations, and ii) collect and manually annotate three additional image sequences, representing a subset of the objects in the dataset randomly placed in different indoor settings. Experiments demonstrate that the proposed learning strategy is effective: DL detectors learn to predict bounding boxes which are more accurate than those available in the training set, and, more importantly, they can generalize well to different settings, providing good detection in absence of explicit segmentation.

The paper is organized as follows: Sec. II reviews related work; Sec. III describes the proposed pipeline, focusing in particular on the data acquisition procedure (Sec. III-A and on the deep architecture adopted for our experiments (Sec. III-B). Sec. IV reports on the experimental results obtained in our study and finally Sec. V draws conclusions and outlines future work.

II. RELATED WORK

In this Section we review the latest architectures for object detection and illustrate conventional training strategies. We motivate the choice of the particular detection method we adopted and relate our contribution to other work in the literature that proposes methods to reduce or automatically compute image annotations.

Deep Architectures for Object Detection and training strategies. Deep Learning methods have advanced the state-of-the-art on object detection. All approaches have at their core a deep Convolutional Neural Network (CNN) [7], [21], [46] often indicated as “feature extractor CNN”. This network is then integrated into a more complex “meta-architecture” which relies on extracted features for the localization and recognition (i.e. detection) of the objects in the image [17]. A possible approach is to partition the image using a grid and perform detection in parallel on all the areas (e.g., SSD (Single-Shot MultiBox Detector) [24] and YOLO (You Only Look Once) [34], [35]). Another approach is to perform detection only on a set of “candidate” regions selected with a separate process (e.g., Region-CNN

(R-CNN) [16] and its optimizations Fast R-CNN [15], Faster R-CNN [36] and Region-FCN [9]). Approaches in the first group are in general faster, because they do not need a per-proposal processing for detection; on the other hand, approaches in the second group proved to be generally more accurate [17]. Among the various solutions, Faster R-CNN [36] seems to be the most suitable for robotics, as it provides good accuracy while preserving efficiency and real-time performance. In this work we employ this method, however it is fair to say that our pipeline is general and could be applied to other models (e.g. SSD).

The parameters of the feature extractor CNN and those of the additional components for the bounding box prediction must be learned from data. These models perform well when trained on large-scale annotated datasets, which include bounding boxes of the objects (e.g. MS COCO [23] and Pascal VOC2007, 2012 [12]). A common approach to reduce the need for the bounding boxes is to first train the feature extractor CNN – which holds the majority of parameters – on a large-scale image classification task (e.g., the ImageNet Large-Scale Visual Recognition Challenge [37]), and then fine-tune it on the target detection task. Strategies have been proposed to speed up the annotation procedure [26], [27]. Still, these have to be performed off-line and still require a considerable manual effort to provide bounding boxes for a sufficiently large number of objects.

Weakly Supervised Approaches. Researchers have proposed alternative methods, which allow training an object detector from images annotated only with object’s presence or absence [2]–[4], [8], [11], [19], [38], [40]–[42], [45]. In some cases adoption of deep CNNs has led to remarkable progress [2]–[4], [8], [19], [41], [42], [45]. However, learning from weak supervision is a difficult task, which leads to performance that are much below those that can be obtained with full supervision.

Exploiting the Context: Learning Features by Training on a Pretext Task. For this reason the literature describes methods that explore ways to extract useful forms of supervision from the contextual information available in real world scenarios (e.g., the spatio-temporal coherence on a sequence of images). Among these, one approach is to exploit the natural structure of the visual data by training a

CNN to solve “alternative” visual tasks [1], [18], [30], [32], [33]. These methods demonstrate that the CNN, which is trained with implicit supervision, can provide good features that can be subsequently fine-tuned on supervised detection tasks.

Exploiting the Context for Computing Annotations. In this paper we follow a similar direction, however, we do not focus on the pre-training of the feature extractor CNN (for which we rely on available models, as it will be detailed in Sec. III-B), but rather exploit the contextual information to compute automatic image annotations (in terms of objects’ labels and locations) to fine-tune the architecture for detection. We rely on the interaction between the robot and a human teacher. Object labels are obtained through a speech interface, while motion and depth cues allow the robot to segment the objects and automatically assign bounding boxes. We show that training data acquired in this way is sufficiently accurate to train an object detection model. Interestingly, we show that the trained model is able to generalize to novel scenarios, allowing detection of objects that are static and in presence of clutter, for which motion and depth cues would not be sufficient.

To our knowledge, there are very few works which address this problem (e.g., [31]) and this is the first attempt in the robotics literature to implement an autonomous learning system for object detection. While the acquisition application is unchanged with respect to our previous work [28], the task is made more difficult because it now uses bounding boxes computed from depth cues to train a predictor that detect objects in the full image.

III. METHODS

The pipeline proposed in this work is an automatic procedure for extracting and labeling example images for training an object detection network. The robot used for our experiments is the iCub humanoid [25]. In the following, we describe the two main steps in the pipeline: i) data acquisition and ii) model training.

A. Data Acquisition Method

For the first step of data acquisition we used the method employed for the ICUBWORLD TRANSFORMATIONS dataset² [28] (shortened to ICWT for simplicity in the following). This method exploits depth information and human-robot interaction to collect labeled images. The procedure is the following: the teacher shows the object in front of the cameras of the iCub. A tracking routine [29], uses stereo vision [14], selecting the pixels from the depth map that are closer to the robot, thus segmenting the object from the background. A bounding box is estimated to surround it, and it is stored as annotation jointly with the label of the objects which is provided verbally by the teacher.

B. Faster Region-CNN Architecture

We chose the region-based method Faster R-CNN [36] as a representative architecture among the ones recently proposed in the DL literature for the same task. This model specifically differs from previous similar approaches (R-CNN [16] and Fast R-CNN [15]) because, instead of employing an external method to provide the object detection network with candidate Regions of Interest (RoIs) –as, e.g., the Selective Search algorithm [44]–, it uses a shallow Convolutional Neural Network called Region Proposal Network (RPN). The RPN, sharing the convolutional layers with the detection network, leads to real-time performance at inference time and allows an efficient training procedure.

As feature extraction CNN, we evaluated and compared two models, whose integration in the Faster R-CNN meta-architecture is publicly available³:

- the ZF network proposed by [46],
- the VGG_CNN_M_1024 network proposed by [7].

For both networks, we initialized the training process by adopting for the shared convolutional layers the weights trained on the image classification task of the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) 2012 [10]. We then fine-tuned by following the 4-Steps Alternating Training pipeline proposed by [36]. This pipeline alternates the optimization of the RPN and of the detection network, thus enabling to learn shared features. In the first two steps, the RPN and the detection network are fine-tuned, by initializing the weights of the shared layers with the ImageNet pre-trained model and training the remaining ones from scratch. In the two latter steps, the shared convolutional layers are kept frozen, and the RPN and the detection network are fine-tuned on the detection task. For fine-tuning, we set the number of training epochs to 6 for steps one and three and to 2 for the other two phases. We left all other parameters unchanged with respect to [36], to which we also refer the reader for a more detailed explanation of the architecture.

IV. EXPERIMENTS

In this Section we report on the experiments that we performed to quantitatively assess the effectiveness of our approach.

As explained in Sec. III-B, we adopt Faster R-CNN [36] with either the ZF or the VGG_CNN_M_1024 feature extractor CNNs. We address an object identification task among 20 objects, by randomly choosing one object instance for each of the 20 categories available in the ICWT dataset presented in Sec. III-A. Figure 2 shows an example image for each selected object. As training set for the considered task, we used the union of the 4 image sequences available in ICWT for each object, corresponding to the 2D ROT,

²iCubWorld website: <https://robotology.github.io/iCubWorld/>

³https://github.com/rbgirshick/py-faster-rcnn/tree/master/models/pascal_voc/ZF

	ovenglove	hairclip	hairbrush	book	ringbinder	flower	cellphone	glass	mouse	wallet	body lotion	pencilcase	soapdispenser	sprayer	sodabottle	mug	sunglasses	perfume	remote	squeezer	mAP
ZF	0.59	0.78	0.71	0.59	0.52	0.86	0.69	0.83	0.92	0.79	0.84	0.41	0.81	0.61	0.79	0.85	0.61	0.75	0.59	0.77	0.72
VGG_CNN_M_1024	0.53	0.76	0.72	0.55	0.51	0.86	0.62	0.84	0.92	0.77	0.83	0.44	0.75	0.63	0.78	0.85	0.64	0.77	0.64	0.79	0.71

TABLE I: AP for each class and mAP (last column) reported by the two Faster-RCNN models considered in this work when tested on the MIX sequences of ICWT. The reference ground truth is the automatic bounding box provided by the depth segmentation routine.

	ZF	VGG_CNN_M_1024
depth's ground truth	0.71	0.69
manual ground truth	0.75	0.73

TABLE II: mAP reported by the two models on a 3K subset of images sampled from the test set of Table I and manually annotated. It can be noticed that the mAP with respect to the manual ground truth is even higher than the one with respect to the depth's automatic ground truth.

3D ROT, BKG and SCALE viewpoint transformations⁴. We considered the two available acquisition days, both for training and testing, and only images from the left camera, overall leading to a training set of $\sim 27K$ images.

We evaluated the system on two settings, respectively in Sec. IV-A and IV-B:

- 1) First, we assessed the effectiveness of the approach in the same HRI setting: a human holds the object to be detected in the hand. For this test, we could use the MIX sequence available in ICWT, leading to a test set of $\sim 13k$ images.
- 2) Then, we evaluated the ability of the detection system to generalize to a different setting. To this end, we acquired and manually annotated three new image sequences, representing the considered 20 objects randomly positioned on the floor, on a table or on a shelf. We made these sequences already available at the same dataset website.

A. Evaluation I: Object detection in the same setting

In this experiment we evaluate the detection performance of the network on a setting similar to the one used for training. Because the training data is segmented and labelled automatically by the robot it inevitably contains errors in the bounding box. This evaluation is therefore important to determine to what extent the detection network is robust to the noise in the training data.

To this end, we considered the two ZF and VGG_CNN_M_1024 models, trained to detect 20 objects from ICWT as described in Sec. III and Sec. IV, and started testing them in the same HRI setting, using the MIX sequences of the considered 20 objects. Testing on these sequences is the best way to evaluate the robustness of the

⁴In ICWT the objects are acquired in a way that separate different transformations: planar 2D rotation (2D ROT), generic rotation (3D ROT), translation with changing background (BKG) and scale (SCALE) and, finally, a sequence that contains all transformations (MIX)

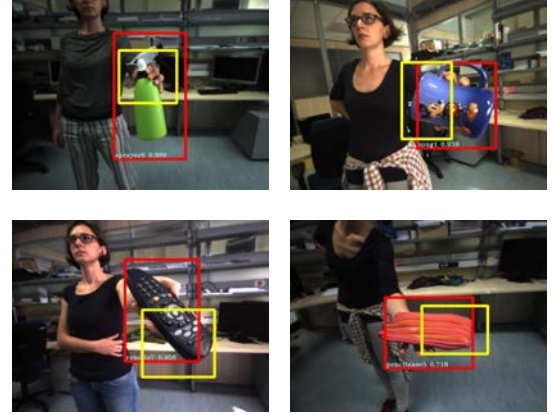


Fig. 3: Example frames where the trained detector (either ZF or VGG_CNN_M_1024) outputs a correct bounding box around the object (red), while the depth segmentation fails (yellow).

predictions, since, we recall, the object is shown naturally to the robot and can appear in any configuration.

In Table I we report, for the two network models, the Average Precision (AP) for each object, with the mean AP (mAP) over all objects (last column). Performance is computed against the ground truth bounding boxes provided by the depth segmentation routine.

It can be first noticed that the reported performance is overall good, in line with the state of the art of deep learning detection systems on other benchmarks (see, e.g., results achieved by Faster RCNN [36] on the Pascal VOC Dataset [12], which consists as well in a 20-class discrimination task). This result is a first important achievement because it shows that these network models, trained with the proposed method, succeeds in localizing and identifying objects in RGB images. Notice that the testing scenario is more challenging, because the object is now localized without the use of depth information. As we will also see in Sec. IV-B, this approach expands the range of possible applications also to settings where the depth cannot be used to localize objects, or not available at all.

In this first test we use as ground-truth the bounding boxes computed automatically using depth information. Because this bounding boxes may contain errors, we further evaluated the system against bounding boxes computed with manual annotation on a subset of the images. We manually annotated this subset of images from the MIX sequences used in the



Fig. 4: Example images from the three sequences (respectively FLOOR (4a), TABLE (4b) and SHELF (4c)) collected and manually annotated to evaluate the generalization performance of the learned detectors with respect to different indoor settings.

previous test. We adopted the *labelImg* tool⁵ and fixed an annotating policy such that an object must be annotated if at least a 50-25% of its total shape is visible (i.e. not cut out from the image or occluded). We annotated 150 frames from each MIX sequence, gathering a test set of 3K images for all the 20 objects that we made available at the ICWT website.

We therefore evaluated the two models on this test set, computing their performance both against the depth’s ground truth and the manual ground truth. In fact, a high AP against the depth’s ground truth implies that the model learned to predict bounding boxes which are “similar” to the ones provided by the automatic annotation procedure, which, however, may contain noise, be biased or less precise with respect to “ideal” bounding boxes provided by a human supervisor. In Table II, we report the mAP of the predicted bounding boxes against automatic ones (first row) and manually annotated ones (second row). Since the mAP evaluated on the manual ground truth is even higher, it can be inferred that, not only the automatic annotations are sufficient to train good detectors, but these networks also learned to “average out” possible noise in the ground truth, performing thus better than the depth segmentation procedure. In Figure 3 we show some example frames where this effect is evident: while the depth segmentation routine fails to segment the object (yellow), the prediction of the model (red) provides a substantially correct bounding box around it.

B. Evaluation II: Detecting objects in other settings

In the evaluation of the previous Section, training and testing took place in similar settings. We performed a further evaluation to determine to what extent the learning system generalizes to a different scenario. This is important because the training uses images that have (i) the constant presence of a human, holding the object in the hand, possibly generating occlusions, and (ii) the presence of a single object of interest, mostly centered (because the robot was tracking it). In this Section we investigate if this bias [43] affects the generalization properties of the network.

We ask therefore whether the networks learned to detect the objects only in these conditions, or are able to generalize to other settings. To this end, we collected and manually annotated three new image sequences, representing three scenes where the objects are randomly positioned respectively on a table, on the floor and on a shelf. These sequences remarkably differ from the ones in ICWT because (i) there is no human presence in the scene and (ii) they contain a variable number of objects, at multiple locations in the image. Finally, also the light and background are different from the ones represented in the dataset. Figure 4 shows an example frame for each sequence (comprising ~ 300 frames):

- The FLOOR sequence depicts 14 out of the 20 objects, lying on the floor.
- The TABLE sequence shows 11 objects on a table with others that are not part of the dataset (like a laptop or a monitor), and hence not to be detected.
- In the SHELF sequence 10 objects are placed on two shelves. The one below is partially shadowed by the one above and, as in the previous sequence, it may contain objects not to be detected.

Table III reports the performance of the two models tested on the three sequences separately, while Fig. 5 shows the predictions of ZF (we obtained similar results with the VGG_CNN_M_1024 model) for two randomly sampled frames from each sequence.

These results show that the performance on these testing sequences remains good, with an average mAPs over the three sequences of 0.58 for the ZF model and 0.52 for VGG_CNN_M_1024. This indicates that the networks succeeded in learning to detect the objects also when these are not hand-held and, as it can be noticed also from the video in the Supplementary Material, which show the predictions of the ZF model running on the iCub while the robot is looking around in these table-top or shelf settings, the proposed approach is a feasible solution to quickly obtain robust and accurate enough object detectors to be used on a general indoor setting.

⁵<https://github.com/tzutalin/labelImg>

	ZF	VGG_CNN_M_1024
FLOOR	0.55	0.47
TABLE	0.66	0.32
SHELF	0.53	0.78

TABLE III: Performance (mAP) of the two models considered in this work when tested on the three image sequences described in Se. IV-B.

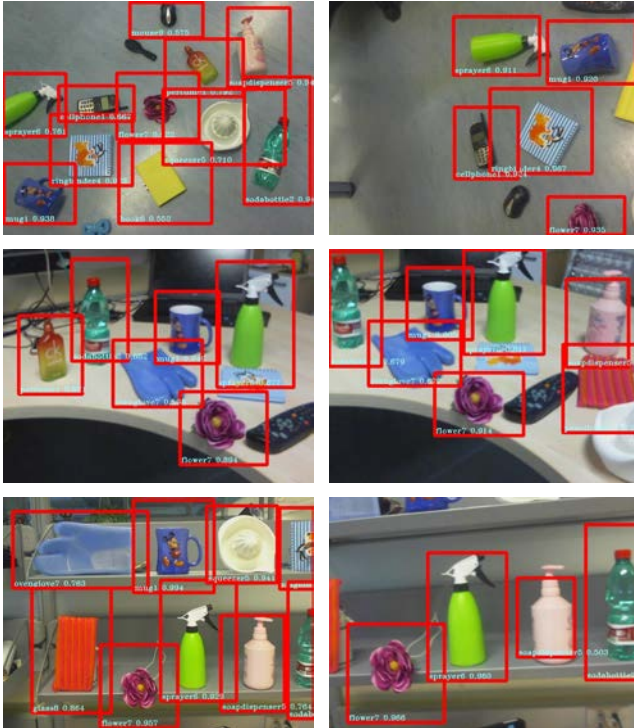


Fig. 5: Example frames, randomly sampled from each sequence, showing the predictions reported by the trained detector (using ZF as feature extractor).

V. DISCUSSION AND CONCLUSIONS

In this work we propose a procedure to train object detection models in a robotic setting with automatically collected annotated data. The presented pipeline adopts the acquisition setup used for the ICUBWORLD TRANSFORMATIONS dataset [28], which leverages on the interaction of the robot with a human teacher and contextual information on the depth of the scene for segmenting and labeling objects observed by the robot, held by the teacher. We demonstrate that images collected in this way are sufficient for fine-tuning a deep architecture as [36] to successfully detect the learned objects without using any depth information at inference time.

We also observed that the noise in the bounding boxes can be averaged out by the training, and that the learned detector generalizes well to different indoor settings. Overall, our experiments suggest that this strategy can be effectively used to deploy object detection systems to realistic robotic applications.

Current work is focusing on further exploring the capa-

bilities of the models trained with the proposed pipeline, considering other real world settings. We are also working on improving the detection performance, e.g., by exploiting the temporal coherence at test time to refine and stabilize the predicted bounding boxes. In addition we are also assessing the employment of other DL architectures for object detection (e.g., SSD [24]) within the same pipeline.

As future work, we plan to expand the proposed acquisition scenario by including information from the motion of the object (which can be done by incorporating, for instance, the work of [13], [22]). Moreover, human gestures like pointing could also be used in order to focus the robot’s attention on objects which are not hand-held. This could be achieved by leveraging on recent improvements in human skeleton detection [6].

Finally, we plan to investigate possible strategies to speedup the current training algorithm in order to provide the robot with the ability of learning on the fly to detect more objects, while interacting with the human. A possible approach which we will evaluate is to start from recent work on incremental object recognition [5] and apply a similar approach to the object detection problem. This would allow to incrementally update the detector, rather than training it from scratch every time new image examples are collected by the robot.

ACKNOWLEDGMENTS

The authors would like to thank NVIDIA Corporation for the donation of a Tesla K40 GPU used for this research. This work is funded by the Air Force project FA9550-17-1- 0390 (European Office of Aerospace Research and Development) and by the FIRB project RBFR12M3AC (Italian Ministry of Education, University and Research).

REFERENCES

- [1] P. Agrawal, J. Carreira, and J. Malik. Learning to see by moving. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 37–45, Dec 2015.
- [2] Hakan Bilen, Marco Pedersoli, and Tinne Tuytelaars. Weakly supervised detection with posterior regularization. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014.
- [3] Hakan Bilen, Marco Pedersoli, and Tinne Tuytelaars. Weakly supervised object detection with convex clustering. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- [4] Hakan Bilen and Andrea Vedaldi. Weakly supervised deep detection networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [5] R.* Camoriano, G.* Pasquale, C. Ciliberto, L. Natale, L. Rosasco, and G. Metta. Incremental robot learning of new objects with fixed update time. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017.
- [6] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017.
- [7] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *British Machine Vision Conference*, 2014.
- [8] Ramazan Gokberk Cinbis, Jakob Verbeek, and Cordelia Schmid. Multi-fold mil training for weakly supervised object localization. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR '14*, pages 2409–2416, Washington, DC, USA, 2014. IEEE Computer Society.

- [9] jifeng dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 379–387. Curran Associates, Inc., 2016.
- [10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.
- [11] Thomas Deselaers, Bogdan Alexe, and Vittorio Ferrari. Localizing objects while learning their appearance. In *Proceedings of the 11th European Conference on Computer Vision: Part IV, ECCV’10*, pages 452–466, Berlin, Heidelberg, 2010. Springer-Verlag.
- [12] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.
- [13] Sean Ryan Fanello, Carlo Ciliberto, Lorenzo Natale, and Giorgio Metta. Weakly supervised strategies for natural object recognition in robotics. *IEEE International Conference on Robotics and Automation*, pages 4223–4229, 5 2013.
- [14] Andreas Geiger, Martin Roser, and Raquel Urtasun. Efficient large-scale stereo matching. In *Asian Conference on Computer Vision (ACCV)*, 2010.
- [15] Ross Girshick. Fast R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015.
- [16] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.
- [17] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. *CoRR*, abs/1611.10012, 2016.
- [18] Dinesh Jayaraman and Kristen Grauman. Learning image representations tied to ego-motion. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- [19] V. Kantorov, M. Oquab, Cho M., and I. Laptev. Contextlocnet: Context-aware deep network models for weakly supervised localization. In *Proc. European Conference on Computer Vision (ECCV)*, IEEE, 2016, 2016.
- [20] C. C. Kemp, A. Edsinger, and E. Torres-Jara. Challenges for robot manipulation in human environments [grand challenges of robotics]. *IEEE Robotics Automation Magazine*, 14(1):20–29, March 2007.
- [21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.
- [22] S. Kumar, F. Odone, N. Noceti, and L. Natale. Object segmentation using independent motion detection. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 94–100, Nov 2015.
- [23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, Zürich, 2014. Oral.
- [24] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, and Scott E. Reed. Ssd: Single shot multibox detector. *CoRR*, abs/1512.02325, 2015.
- [25] Giorgio Metta, Lorenzo Natale, Francesco Nori, Giulio Sandini, David Vernon, Luciano Fadiga, Claes von Hofsten, Kerstin Rosander, Manuel Lopes, José Santos-Victor, Alexandre Bernardino, and Luis Montesano. The icub humanoid robot: an open-systems platform for research in cognitive development. *Neural networks : the official journal of the International Neural Network Society*, 23(8-9):1125–34, 1 2010.
- [26] Dim P. Papadopoulos, Jasper R. R. Uijlings, Frank Keller, and Vittorio Ferrari. We don’t need no bounding-boxes: Training object class detectors using only human verification. *CoRR*, abs/1602.08405, 2016.
- [27] Dim P. Papadopoulos, Jasper R. R. Uijlings, Frank Keller, and Vittorio Ferrari. Training object class detectors with click supervision. *CoRR*, abs/1704.06189, 2017.
- [28] G. Pasquale, C. Ciliberto, L. Rosasco, and L. Natale. Object identification from few examples by improving the invariance of a deep convolutional neural network. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4904–4911, Oct 2016.
- [29] Giulia Pasquale, Tanis Mar, Carlo Ciliberto, Lorenzo Rosasco, and Lorenzo Natale. Enabling depth-driven visual attention on the icub humanoid robot: Instructions for use and new perspectives. *Frontiers in Robotics and AI*, 3:35, 2016.
- [30] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *CVPR*, 2017.
- [31] Sudeep Pillai and John J. Leonard. Monocular SLAM supported object recognition. *CoRR*, abs/1506.01732, 2015.
- [32] L. Pinto and A. Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3406–3413, May 2016.
- [33] Lerrel Pinto, Dhiraj Gandhi, Yuanfeng Han, Yong-Lae Park, and Abhinav Gupta. *The Curious Robot: Learning Visual Representations via Physical Interactions*, pages 3–18. Springer International Publishing, Cham, 2016.
- [34] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [35] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*, 2016.
- [36] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*, 2015.
- [37] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.
- [38] Olga Russakovsky, Yuanqing Lin, Kai Yu, and Li Fei-Fei. Object-centric spatial pooling for image classification. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part II, ECCV’12*, pages 1–15, Berlin, Heidelberg, 2012. Springer-Verlag.
- [39] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Robert Fergus, and Yann Lecun. *Overfeat: Integrated recognition, localization and detection using convolutional networks*. 2014.
- [40] Parthipan Siva and Tao Xiang. Weakly supervised object detector learning with model drift detection. *2011 International Conference on Computer Vision*, pages 343–350, 2011.
- [41] Hyun Oh Song, Ross Girshick, Stefanie Jegelka, Julien Mairal, Zaid Harchaoui, and Trevor Darrell. On learning to localize objects with minimal supervision. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1611–1619, Beijing, China, 22–24 Jun 2014. PMLR.
- [42] Hyun Oh Song, Yong Jae Lee, Stefanie Jegelka, and Trevor Darrell. Weakly-supervised discovery of visual pattern configurations. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 1637–1645. Curran Associates, Inc., 2014.
- [43] Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1521–1528, 2011.
- [44] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.
- [45] Chong Wang, Kaiqi Huang, Weiqiang Ren, Junge Zhang, and Steve Maybank. Large-scale weakly supervised object localization via latent category learning. *IEEE Transactions on Image Processing*, 24(4):1371–1385, 2015.
- [46] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013.

Speeding-up Object Detection Training for Robotics with FALKON

Elisa Maietti^{1,2,3}, Giulia Pasquale^{1,2}, Lorenzo Rosasco^{2,3} and Lorenzo Natale¹

Abstract—Latest deep learning methods for object detection provide remarkable performance, but have limits when used in robotic applications. One of the most relevant issues is the long training time, which is due to the large size and imbalance of the associated training sets, characterized by few positive and a large number of negative examples (i.e. background). Proposed approaches are based on end-to-end learning by back-propagation [22] or kernel methods trained with Hard Negatives Mining on top of deep features [8]. These solutions are effective, but prohibitively slow for on-line applications.

In this paper we propose a novel pipeline for object detection that overcomes this problem and provides comparable performance, with a 60x training speedup. Our pipeline combines (i) the Region Proposal Network and the deep feature extractor from [22] to efficiently select candidate RoIs and encode them into powerful representations, with (ii) the FALKON [23] algorithm, a novel kernel-based method that allows fast training on large scale problems (millions of points). We address the size and imbalance of training data by exploiting the stochastic subsampling intrinsic into the method and a novel, fast, bootstrapping approach.

We assess the effectiveness of the approach on a standard Computer Vision dataset (PASCAL VOC 2007 [5]) and demonstrate its applicability to a real robotic scenario with the iCubWorld Transformations [18] dataset.

I. INTRODUCTION

Visual recognition and localization of objects is crucial for robotic platforms. This problem is known in Computer Vision as object detection, where the objects represented in an image are localized with a bounding box and then associated to a label. Deep learning has remarkably boosted the performance of the state of the art [4], [24], [14], [26], [8], [22], [15], [1] but training such systems require large amount of data and it is slow. For robotic applications it is desirable to use methods that allow robots to quickly adapt to the environment.

In previous work [18], [19] we proposed an interactive method that allows the robot to automatically acquire annotated images by interacting with a human and demonstrated that data acquired in this way allows training an object detection algorithm (i.e. Faster R-CNN [22]) with good accuracy [16].

However, in [16] training was performed off-line as in [22], i.e., by fine-tuning the model on the task at hand. Most latest architectures for object detection are learned end-to-end by gradient descent with back-propagation [22], [15], [1], [21]. This is achieved by formulating a single

optimization problem, to learn, jointly, the three stages of the classical detection pipeline: extraction of candidate regions, feature encoding and regions classification (see [10] for a comprehensive overview). This training protocol requires massive computational resources and hours/days of training. In this paper we propose an object detection pipeline that can be trained on-line, (i.e. in seconds), without sacrificing detection performance.

We adopt an approach that is similar to Region-CNN [8], in which each stage of the pipeline is trained independently. This approach is more suitable for an on-line learning because the region proposals and the deep representation can be kept fixed while the final region classification is trained. For instance, Region-CNN uses Selective Search [31] and a pre-trained CNN to get candidate regions and encode them into features, on top of which Support Vector Machines (SVMs) are trained.

Training neural models for object detection is computationally demanding. This is due to the large number of candidate regions that are extracted from the training images, especially from the background. Typically a time consuming process is applied to extract a small number of negative examples on which to train the system. Such process aims at selecting a subset of negative examples that are meaningful (i.e. ‘hard’) and whose size is comparable to the available positive examples, thus re-balancing the dataset. As an example training in Region-CNN uses the Hard Negative Mining procedure of [30] and it requires hours to complete.

We compose our pipeline by first adopting the Region Proposal Network (RPN) presented in Faster R-CNN [22] for learning the candidate regions: this is a double-layer Convolutional Neural Network (CNN) that can be trained off-line in relatively short time and, since it is object-agnostic, can be used for multiple object detection tasks. Using another deep CNN we encode each region into a set of feature vectors. These feature vectors are then fed to a kernel-based classifier which is trained on-line. In doing so, for the first off-line part, we basically adopt the architecture of Faster R-CNN [22], except that the RPN and the feature extractor CNN are learned on one task, and then re-used on different tasks.

This choice is motivated by the literature suggesting that deep CNNs provide powerful and general features which can be used for multiple tasks [27], [3], [11], [8], [18].

For region classification we adopt FALKON [23], a recent kernel-based method for large-scale datasets. We leverage on the stochastic sampling of the kernel centers performed by FALKON and propose an approximated version of the Hard Negative Mining procedure, to efficiently re-balance the

¹ iCub Facility, Istituto Italiano di Tecnologia, Genoa, Italy

² Laboratory for Computational and Statistical Learning, Istituto Italiano di Tecnologia and Massachusetts Institute of Technology, Cambridge, MA

³ Dipartimento di Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi, University of Genoa, Genoa, Italy

training set. We show that depending on the desired computation time, this procedure can be tuned to subsample more or less extensively the training set without compromising performance.

We validate the pipeline on a subset of the PASCAL VOC 2007 [5], by comparing performance and training time with Faster R-CNN. Finally, we experiment with a real-world robotic application to demonstrate that the proposed approach allows learning a novel task (10 classes $\times$ 10k images) in few seconds. To this end, we use the iCubWorld Transformations Dataset [18], a robotic benchmark for object recognition and detection.

II. RELATED WORK

The problem of object detection can be naturally decomposed into two subtasks: (1) objects localization and (2) image classification. Historically many approaches have been proposed which address these tasks in different ways. They can be grouped as follows:

Grid-based object detectors. In this case a classifier is applied on a dense image grid, obtained using a Sliding Window paradigm [12], [6] or a fixed stride. The work of LeCun et al. [12] is one of the first where convolutional neural networks were applied in a Sliding Window fashion. More recently, other grid-based approaches have been proposed like SSD (Single-Shot MultiBox Detector) [15] and YOLO (You Only Look Once) [20], [21].

Region-based object detectors. Algorithms belonging to this group perform detection only on a set of “candidate” Regions of Interest (RoIs), selected with a separate process (see e.g., Region-CNN (R-CNN) [8] and its optimizations Fast R-CNN [7], Faster R-CNN [22], Region-FCN [1] and Mask R-CNN [9]). These architectures employ different methods for predicting RoIs. For example Region-CNN usually relies on Selective Search [31], while Faster R-CNN uses a dedicated CNN, called Region Proposal Network (RPN) [22]. The RPN is a double-layer Convolutional Neural Network, which is trained on the data to discriminate background from potential objects. The main advantage of this approach is that it provides a limited number of predicted RoIs with low computation time [22].

Among these architectures we can distinguish between those which are trained end-to-end [1], [9], [22] and those which need a multi-stage learning process [8].

In this work we propose a pipeline that exploits the benefits of an RPN for RoIs prediction while relying on a two-stage training procedure to allow fast model adaptation to new tasks.

The problem of background selection. The problem with grid-based and region-based detectors is that they produce a large number of RoIs, most of which originate from the background. Because negative examples are computed from these RoIs, this leads to a training set that is large and highly unbalanced. The size of the training set makes

learning computationally unfeasible, and the fact that positive examples are underrepresented inevitably bias the result of the training.

In the literature, methods have been proposed to deal with these issues. The first solution to be introduced was based on bootstrapping [30] (now often named as Hard Negative Mining [8]). This approach performs an iterative training, which selects a set of ‘hard’ negatives, i.e. negative examples that are difficult to classify. This solution is still currently adopted in the state of the art like R-CNN [8], and it has been adapted recently to be used during back-propagation [28], [7].

More recently, a new object detector has been proposed (i.e. RetinaNet [13]). It is a CNN based approach in which a novel loss function, called Focal Loss, is adopted for training end-to-end and deal with class imbalance. This new function is designed to down-weight easy negative examples such that their contribution to the total loss is small even if their number is large.

These solutions are effective and produce impressive results. However they are intrinsically slow because they rely on back-propagation, while bootstrapping requires to iteratively visit all negative examples in the training set. Training such systems takes hours even for medium scale datasets of few thousands of images, on servers equipped with powerful GPUs as the one used for the experimental evaluation of this work (an NVIDIA(R) Tesla P100 GPU).

In this work we propose a novel approach to (i) select hard negatives, by implementing an approximated and faster bootstrapping procedure, and (ii) account for the imbalance between positive and negative regions by relying on a Nyström based kernel method (namely FALKON [23]).

III. METHODS

We considered the scenario in which a human teaches the robot to detect a set of novel object instances (TARGET-TASK in the following). We suppose that the visual system of the robot has been previously trained on a different set of objects (PREV-TASK in the following), and that the convolutional weights are detained for future use.

We propose an object detection method that can be trained on-the-fly (i.e. at run time) on the TARGET-TASK, by relying on some of the components previously trained on PREV-TASK. Referring to a typical detection pipeline, composed of (i) a region proposal stage, (ii) a feature extraction stage and (iii) a final classification stage, we propose an approach which learns (i) the region proposals and (ii) the feature extractor, on PREV-TASK, such that, when the robot is required to learn the TARGET-TASK, only the final classification stage is re-trained on-line. Our major contribution consists in an efficient approach to perform the latter step, which remarkably reduce training time while retaining performance. Specifically, we propose to use a set of FALKON classifiers, Regularized Least Squares (RLS) regressors for bounding boxes refinement, and an approximated method for selecting positive and negative samples.

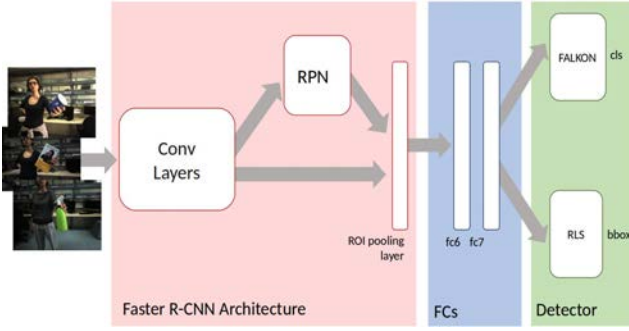


Fig. 1: Overview of the proposed on-line object detection pipeline. For a detailed description see Sec. III.

In this Section, we describe the main steps of the proposed pipeline. We first motivate the choices of the region proposal and feature extraction method (Sec. III-A). Then, we focus on the classifier algorithm (Sec. III-C), and describe how it addresses the large imbalance between positive and negative examples. Finally we describe the procedure for selecting background samples (Sec. III-D).

A. Pipeline Overview

We refer to Fig. 1 for a pictorial representation of each stage of the pipeline. For the first stages, we adopt the architecture of Faster R-CNN [22]. We exploit the class-agnostic Region Proposal Network (RPN) to predict a set of candidate RoIs. Each of them is then associated to a deep feature map by means of the so-called RoI pooling layer [7]. This allows to encode each proposed region into a deep representation, using only one forward pass of the convolutional feature extraction layers. Specifically, we adopt the CNN model proposed in [33] as convolutional backbone of the algorithm (whose integration in Faster R-CNN is publicly available¹).

Finally, in Faster R-CNN, the pooled features from the region proposals are processed by the so-called detection network, which is composed of two fully-connected layers and two final output layers for class prediction and bounding box refinement. In our pipeline, we keep the two fully connected layers, but replace the two output layers respectively with the FALKON classifiers and Regularized Least Squares (RLS) regressors for the refinement of the bounding boxes.

B. Training

In the considered scenario, we train Faster R-CNN on the PREV-TASK in order to learn the RPN and the convolutional and fully connected layers (CNN feature extractor). When learning the TARGET-TASK, we use these components to extract and encode region proposals into deep features, and we train on-line only the FALKON classifiers and RLS regressors.

To learn the RPN and the CNN feature extractor, we adopted the 4-Steps Alternating Training method proposed

in [22]. This method alternates the optimization of the RPN and the detection network, thus learning shared convolutional features. In the first two steps, respectively the RPN and the Detector are learned from scratch, while the shared convolutional layers and the fully-connected layers are fine-tuned, starting from weights previously trained on ImageNet (the image classification task of Large-Scale Visual Recognition Challenge (ILSVRC) 2012 [2]). In the two latter steps, the shared convolutional layers are frozen, and the RPN and the detection network are fine-tuned on the target detection task (in our section PREV-TASK). We refer the reader to [22] for a detailed explanation of the architecture and this training procedure.

In the following section, we describe more in detail the algorithm and learning of the remaining part of the pipeline, which is the core of the proposed approach.

C. FALKON: efficient regions classification

The candidate RoIs extracted by the RPN represent potential object locations, which need to be classified as belonging to either one of the considered classes (namely, the object instances of our TARGET-TASK), or rejected as background.

We address a multiclass classification task by learning a set of binary classifiers, one for each class, in a one-vs-all fashion. For each category, we collect the training set by selecting and labeling candidate RoIs as either positive samples (i.e. belonging to the class, indicated as P in the following) or negative ones (i.e. background or other classes, indicated as N in the following). This results in a large dataset.

Training standard kernel methods for classification on large datasets can be prohibitive, because it requires to solve the linear system (also known as Kernel Ridge Regression or KRR): $(K_{nn} + \lambda nI) \alpha = \hat{y}$, where n is the number of training points $\{(x_1, y_1), \dots, (x_n, y_n)\}$, $(K_{nn})_{ij} = K(x_i, x_j)$ is the Kernel matrix and λ is a regularization parameter.

It can be easily observed that this problem does not scale well with the number of samples n . Just storing K requires $O(n^2)$ in memory space while computing and inverting K_{nn} (i.e. learning phase) requires $O(n^2c + n^3)$ in time (where c is the kernel evaluation cost). FALKON [23] approximates the KRR problem using a Nystrom method [32], [29] by stochastically sampling a subset of $M \ll n$ training points as Kernel centers. In addition it uses the conjugate gradient method [25] associated with a preliminary preconditioning, for an iterative and faster solution of the associated linear system.

FALKON requires $O(M^2)$ in memory space and $O(nMt + M^3)$ for the kernel computation and inversion, where t is the number of iterations required. Since it has been shown that choices of $M \ll n$ preserve statistical properties [23], accuracy is retained, while we gain a significant boost in computation performance.

We refer the reader to [23] for a detailed description of the method. In our pipeline, we adopted the publicly available FALKON implementation².

¹https://github.com/rbgirshick/py-faster-rcnn/tree/master/models/pascal_voc/ZF

²https://github.com/LCSL/FALKON_paper

D. Background samples selection

We propose two strategies in order to (i) select hard negatives and (ii) account for the imbalance between positive and negative regions. As it will be shown in the experimental Sec. IV, adopting these strategies in combination with FALKON proved to be fundamental in order to reduce the training time without losing accuracy.

Approximated Hard Negatives Mining. A first component of our approach is an approximation of the Hard Negatives Mining method adopted in [8] and proposed in [30]. Core idea behind the original method is to gradually grow (bootstrap) the set of negative examples by repeatedly training and testing the detector and by including in the training set only those samples which are hard to classify correctly for the detector.

This idea is implemented with an iterative procedure which, for each class, visits all images in the training set and, for each image i :

- 1) tests the model trained at the previous iteration ($model_{i-1}$) on *all* negative regions in the i^{th} image, to select the hard ones, maintaining a number of N_i^H ;
- 2) learns a new model ($model_i$) on the train set composed by the union of the P positives (which are fixed) and the hard negatives collected so far ($N_{chosen,i-1} \cup N_i^H$);
- 3) tests $model_i$ on the negatives on which it has been trained in order to prune the easy ones maintaining a number of $N_{chosen,i}$.

The output of this procedure is a number of $N_{chosen,final}$ (hard) negatives examples, which, jointly with the P positive regions, are used to train the final version of the model. The procedure is repeated for all the binary classifiers that are trained on the multiclass problem.

Such an approach is clearly time consuming, because it iterates over all images in the training set and processes *all* negative regions proposed by the RPN. Therefore, in our pipeline we propose to use an approximation, which consists in (i) considering a random subset of *all* negative regions extracted by the RPN from all training images and (ii) dividing the sampled regions into a number n_B mini-batches (of size B). Finally we select a number of hard negatives by applying the Hard Negatives Mining method, described above, on each batch.

In Sec. IV we compare three variants of the proposed approach:

FALKON + MINI BOOTSTRAP ($n_B \times B$): This is the approximated procedure we outlined above, when both the size of the mini-batches B and the number of bootstrapping iterations n_B are parameters that can be varied to tune the procedure.

FALKON + FULL BOOTSTRAP: This is the strategy adopted by [8]. It can be seen as performing our approximated procedure when considering all negative regions in the training set, setting each mini-batch to contain all the negatives from each of the n images of the dataset (i.e.,

$n_B = n$ and B set as the number of negatives in each image).

FALKON + RANDOM BKG: In this case we do not perform any type of hard negatives selection, but we randomly sample a subset from all the negative regions proposed by the RPN. This can be seen as performing our approximated procedure with $n_B = 0$ and B set as the size of the selected subset.

Rebalancing Nystrom centers. The second component of our approach is the stochastic sampling of the Nystrom centers performed by FALKON, to account for the positive-negative imbalance. Specifically, we propose to condition the stochastic sampling of the M centers in the algorithm, at each iteration of the approximated bootstrapping procedure described above, such that we take a number of P' positives with $P' = \min(P, \frac{M}{2})$, while we randomly choose the remaining $(M - P')$ centers among the $N_{chosen,i-1} \cup N_i^H$ negatives obtained at the i^{th} iteration. This step is fundamental because, when $P \ll N$, randomly sampling the M centers among the union of the two sets might lead to further reducing the number of positives with respect to the number of negatives and, in the worst case, discarding all positives from the sampled Nystrom centers.

The fundamental parameters of the proposed approach are (i) the number of selected Nystrom centers (M), (ii) the number of bootstrapping iterations (n_B), (iii) the size of the mini-batches of negatives (B), and (iv) FALKON's Gaussian kernel parameters λ and σ . We cross-validated these latter two using a one-fold cross-validation strategy, considering as validation set a subset of 20% of the training set. In Sec. IV, we provide experimental evaluation of the other parameters which allow to tune the procedure to subsample more or less extensively the training set, depending on the desired computation time.

IV. EXPERIMENTS

In this section we provide experimental evaluation of the proposed on-line object detection pipeline. We compare the three different training protocols described in Sec. III-D using Faster R-CNN as baseline.

In Sec. IV-B we test the pipeline in a robotic setting using the ICUBWORLD TRANSFORMATIONS dataset [18], considering the scenario described in Sec. III.

All experiments reported in this paper have been performed on a machine equipped with Intel(R) Xeon(R) E5-2690 v4 CPUs @2.60GHz, and a single NVIDIA(R) Tesla P100 GPU. We set FALKON to not use more than 10GB of RAM.

A. Experimental Validation on PASCAL VOC 2007

We validate our method on a subset of PASCAL VOC 2007 [5], a standard computer vision dataset. We report performance in terms of (i) mAP (mean Average Precision), as defined for PASCAL VOC 2007, and (ii) training time. Specifically, we considered 7 classes among the 20 that

Method	mAP [%]	Train Time
Faster R-CNN	51,9	~25 min
FALKON + Full Bootstrap ($\sim 1K \times 1000$)	51,5	~8 min
FALKON + Random BKG (0×7000)	47,7	~25 sec

TABLE I: Performance comparison on a subset of 7 classes of the PASCAL VOC 2007. See the text (Sec. IV-A) for details about the task and Sec. III-D for details about the methods.

are available (*aeroplane, bicycle, bird, boat, bottle, bus, car*). As train and test set we selected, from the VOC 2007 *trainval* and *test* sets, all the images that depict at least one instance which belongs to one of the 7 classes. Overall we collected a training set of $\sim 1K$ images and a test set of $\sim 2K$.

Approximated Hard Negatives Mining. As a first validation, we explored different configurations of the FALKON + MINI BOOTSTRAP approach proposed in Sec. III-D. We vary the number of bootstrapping iterations (n_B) between 0 (i.e., no bootstrapping) and 1K (i.e., full bootstrapping, by visiting all training images one by one), with varying size of background batches (B). Coherently, we observed an improvement in mAP (with an increasing training time) when progressively performing a more extensive bootstrap.

We report, in Table I, the results provided by two “extreme” training conditions, comparing them with the baseline represented by fine-tuning Faster R-CNN’s last layers (which we consider the “upper bound” for the expected mAP). We consider (i) FALKON + RANDOM BKG, for which we did not perform bootstrapping and set the number of randomly sampled background regions to 7000 (higher values did not improve performance), and (ii) FALKON + FULL BOOTSTRAP. In this latter case, as explained in Sec. III-D, we performed as many bootstrapping iterations as the number of training images (1K), processing a batch of ~ 1000 background regions for each visited image. In both experiments, in this case, we set the number of Nystrom centers equal to the number of training points (this parameter is investigated in more details in the next paragraph).

When fine-tuning Faster R-CNN, we froze all layers (RPN and CNN’s layers up to *fc7*), except the last fully connected layers for classification and bounding box regression (namely, *cls* and *bbbox-reg*) which we trained from scratch. For this step we set the number of iterations to 15K.

As it can be observed from Table I, we could train a detection model in ~ 25 seconds, with a 4% gap in performance with respect to the mAP provided by fine-tuning Faster R-CNN’s last layers (which however requires ~ 25 minutes). Moreover, we were able to reproduce state of the art performance (up to 0.4%) in 8 minutes. In Sec. IV-B, we show that, on the robot, it is possible to find a bootstrapping configuration that remarkably reduce training time while retaining performance.

Rebalancing Nystrom centers. As a second validation, in Fig. 2 we report performance on the same task when varying

the number of Nystrom centers (M parameter in Sec. III-C). For this experiment, we considered the FALKON + RANDOM BKG configuration of Table I, and progressively decreased M , by applying, for each value, the rebalancing approach described in Sec. III-D. For each value of M , we report the mAP (Left), the training time (Center) and the testing time (Right). It can be observed that mAP performance degrades only when using very few centers, and that a value of $M = \sim 500$ is sufficient to achieve optimal performance, fast training (~ 12 seconds) and testing time (around ~ 10 FPS, obtained by dividing the number of tested images, $\sim 2K$, by the reported testing time for $M = 500$, i.e., ~ 225 seconds).

B. Experiments on the ICUBWORLD TRANSFORMATION dataset

To evaluate the effectiveness of the proposed pipeline in a robotic scenario, we consider the ICUBWORLD TRANSFORMATIONS dataset (ICWT in the following).

iCubWorld Transformations. This dataset has been automatically acquired in a natural teacher-learner setting by using the iCub robot [17]. The teacher shows an object in front of the robot, which moves the eyes to fixate and track it. Using its stereo system, the robot segments the object by selecting those points which are closest to the cameras³. The bounding box that contains the segmented object, is then stored jointly with the label of the object which is provided verbally by the teacher. We refer to [19] for a complete description of the acquisition setup and to [18] for details about the dataset, which is publicly available⁴.

Experimental Setup. For our experiments we considered the scenario described in Sec. III and we defined the two tasks (PREV-TASK and TARGET-TASK) as two object identification tasks among 10 object instances. We randomly chose one object instance from each of the 20 different categories in the dataset, and split them into two sets of 10 objects each. We show an example image for each object considered in both tasks in Fig. 3.

For each task, we considered, as training set for each object, the union of the 4 image sequences available in ICWT, corresponding to the 2D ROT, 3D ROT, BKG and SCALE viewpoint transformations, taking into account both acquisition days. Overall this leads to a set of $\sim 10K$ images. As a test set for each object we used the remaining MIX sequence, in both acquisition days. We refer to [18] for details about the dataset’s sequences.

We trained Faster R-CNN end-to-end on PREV-TASK, setting the number of iterations to 40K when learning the RPN and to 20K when learning the CNN detector.

We finally trained the FALKON classifiers and evaluated performance on the test set of TARGET-TASK.

³This approach assumes that the object of interest is the closest entity to the robot in the scene, which in practice holds in our scenario

⁴<https://robotology.github.io/iCubWorld/>

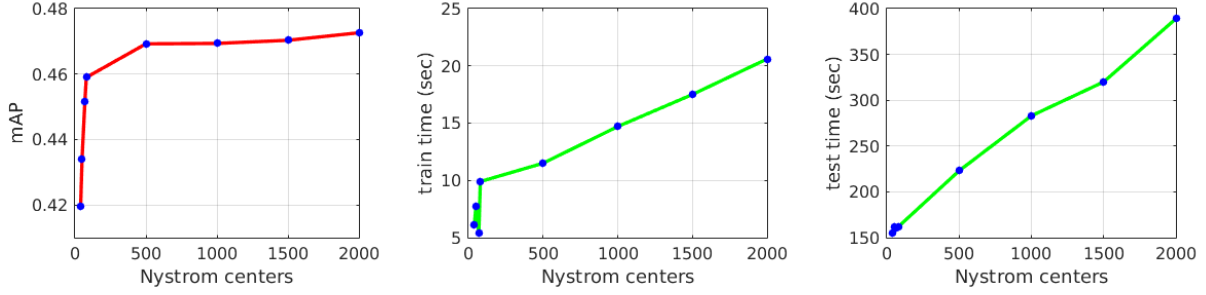


Fig. 2: The three plots refer to the experiment discussed in Sec. IV-A and report the mAP (Left), training time (Center) and testing time (Right) for increasing values of the number of Nystrom’s centers (M) when using FALKON.

As for PASCAL VOC, we considered Faster R-CNN as a baseline. To this end, we used the model obtained by training end-to-end on PREV-TASK, and froze all layers (RPN and CNN’s layers up to $fc7$), while learning from scratch only the last fully-connected layers for classification and bounding box regression (namely, cls and $bbox-reg$). For this step we set the number of iterations to 20K.

Results. In Table II we compare the performance of Faster R-CNN baseline with the methods FALKON + RANDOM BKG and FALKON + MINI BOOTSTRAP (Sec. III-D).

Based on the empirical observations from Sec. IV-A, in these experiments we use a relatively large value of Nystrom centers (around half the size of the training set), because this does not have a negative impact on the training time as demonstrated by the experiments in Table I.

As in PASCAL VOC, for FALKON + RANDOM BKG we set the number of randomly sampled background regions to 6000, because we observed that further increasing this parameter did not improve performance.

Regarding FALKON + MINI BOOTSTRAP, we observed that a few bootstrapping iterations already give performance that are comparable to the Faster R-CNN. As an example, in Table II we report two different configurations of FALKON + MINI BOOTSTRAP: (i) $n_B = 4$ and $B = 2500$ (FALKON + MINI BOOTSTRAP (4×2500)) and (ii) $n_B = 10$ and $B = 1500$ (FALKON + MINI BOOTSTRAP (10×1500)).

It can be noticed that FALKON + RANDOM BKG has the fastest training time but with lowest mAP. On the contrary, the proposed bootstrap method, FALKON + MINI BOOTSTRAP (4×2500) provides mAP performance which are comparable with those obtained with Faster R-CNN, with just 40 seconds of training (in contrast fine-tuning Faster R-CNN takes 40 minutes). Moreover, FALKON + MINI BOOTSTRAP (10×1500) outperforms fine-tuning with a train time of ~ 50 seconds.

V. CONCLUSIONS

In this paper we propose a system for object detection that combines Faster R-CNN [22], with FALKON [23], a kernel-based method specifically designed for large-scale datasets. In addition, we include a novel approximated technique for

pruning the training set by selecting hard negative examples. Our approach can be trained much faster than Faster R-CNN ($\approx 60\times$) while preserving comparable detection performance.

Fast learning methods are fundamental for robots to quickly adapt to their environment. Deep-learning methods for object detection achieve remarkable performance, but are slow to train. This hinders their adoption in those scenarios that require robots to learn on-line. In this sense the work described in this paper is an important step toward the implementation of more adaptive robotic systems.

ACKNOWLEDGMENTS

The authors would like to thank NVIDIA Corporation for the donation of a Tesla K40c GPU used for this research. This work is funded by the Air Force project FA9550-17-1-0390 (European Office of Aerospace Research and Development) and by the MSCA-RISE, 734564.

REFERENCES

- [1] jifeng dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 379–387. Curran Associates, Inc., 2016.
- [2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.
- [3] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In Tony Jebara and Eric P. Xing, editors, *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 647–655. JMLR Workshop and Conference Proceedings, 2014.
- [4] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, January 2015.
- [5] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.
- [6] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, Sept 2010.
- [7] Ross Girshick. Fast R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015.
- [8] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.



Fig. 3: Example images representing the 10 object instances involved in the first object identification task (PREV-TASK), i.e., the task on which we learn the RPN and the CNN feature extractor (a) and in the second object identification task (TARGET-TASK), i.e., the actual task that is learned on-line (b).

Method	Train Time	mAP [%]	soda bottle	mug	pencil case	ring binder	wallet	flower	book	body lotion	hair clip	sprayer
Faster R-CNN Fine-tuning	~40 min	49,7	63,2	68,4	23,3	29,6	49,9	66,1	35,3	56,2	60,2	45,8
FALKON + Random BKG (0×6000)	~25 sec	40,5	57,7	67,9	17,5	23,1	23,8	59,5	26,8	39,6	48,5	40,5
FALKON + Mini Bootstrap (4×2500)	~40 sec	48,1	63,1	67,2	18,4	25,7	47,4	70,3	36,1	52,3	58,8	41,5
FALKON + Mini Bootstrap (10×1500)	~50 sec	51,3	64,7	71,2	27,2	31,7	56,9	69,4	39,6	54,0	60,7	37,1

TABLE II: Performance comparison on a 10-object identification task on ICWT (TARGET-TASK in Sec. IV-B). See the text for details about the task and Sec. III-D for details about the methods.

- [9] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask r-cnn. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2980–2988, 2017.
- [10] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and Kevin Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. *CoRR*, abs/1611.10012, 2016.
- [11] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the ACM International Conference on Multimedia - MM '14*, pages 675–678. ACM Press, 11 2014.
- [12] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, Dec 1989.
- [13] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. pages 2999–3007, 2017.
- [14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollr, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, Zrich, 2014. Oral.
- [15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, and Scott E. Reed. Ssd: Single shot multibox detector. *CoRR*, abs/1512.02325, 2015.
- [16] E. Maietini, G. Pasquale, L. Rosasco, and L. Natale. Interactive data collection for deep learning object detectors on humanoid robots. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 862–868, Nov 2017.
- [17] Giorgio Metta, Lorenzo Natale, Francesco Nori, Giulio Sandini, David Vernon, Luciano Fadiga, Claes von Hofsten, Kerstin Rosander, Manuel Lopes, José Santos-Victor, Alexandre Bernardino, and Luis Montesano. The icub humanoid robot: an open-systems platform for research in cognitive development. *Neural networks : the official journal of the International Neural Network Society*, 23(8-9):1125–34, 1 2010.
- [18] G. Pasquale, C. Ciliberto, L. Rosasco, and L. Natale. Object identification from few examples by improving the invariance of a deep convolutional neural network. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4904–4911, Oct 2016.
- [19] Giulia Pasquale, Tanis Mar, Carlo Ciliberto, Lorenzo Rosasco, and Lorenzo Natale. Enabling depth-driven visual attention on the icub humanoid robot: Instructions for use and new perspectives. *Frontiers in Robotics and AI*, 3:35, 2016.
- [20] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [21] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*, 2016.
- [22] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*, 2015.
- [23] Alessandro Rudi, Luigi Carratino, and Lorenzo Rosasco. Falkon: An optimal large scale kernel method. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3888–3898. Curran Associates, Inc., 2017.
- [24] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.
- [25] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd edition, 2003.
- [26] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Robert Fergus, and Yann Lecun. *Overfeat: Integrated recognition, localization and detection using convolutional networks*. 2014.
- [27] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: An astounding baseline for recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2014.
- [28] Abhinav Shrivastava, Abhinav Gupta, and Ross B. Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, pages 761–769. IEEE Computer Society, 2016.
- [29] Alex J. Smola and Bernhard Schölkopf. Sparse greedy matrix approximation for machine learning. pages 911–918. Morgan Kaufmann, 2000.
- [30] Kah Kay Sung. Learning and example selection for object and pattern detection. 1996. AAI0800657.
- [31] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.
- [32] Christopher K. I. Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 682–688. MIT Press, 2001.
- [33] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013.