FALKON: An Optimal Large Scale Kernel Method

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

Luigi Carratino
University of Genoa
Genova, Italy

Lorenzo Rosasco
University of Genoa,
LCSL, IIT & MIT

Abstract

Kernel methods provide a principled way to perform non linear, nonparametric learning. They rely on solid functional analytic foundations and enjoy optimal statistical properties. However, at least in their basic form, they have limited applicability in large scale scenarios because of stringent computational requirements in terms of time and especially memory. In this paper, we take a substantial step in scaling up kernel methods, proposing FALKON, a novel algorithm that allows to efficiently process millions of points. FALKON is derived combining several algorithmic principles, namely stochastic subsampling, iterative solvers and preconditioning. Our theoretical analysis shows that optimal statistical accuracy is achieved requiring essentially $O(n)$ memory and $O(n\sqrt{n})$ time. An extensive experimental analysis on large scale datasets shows that, even with a single machine, FALKON outperforms previous state of the art solutions, which exploit parallel/distributed architectures.

1 Introduction

The goal in supervised learning is to learn from examples a function that predicts well new data. Nonparametric methods are often crucial since the functions to be learned can be non-linear and complex Kernel methods are probably the most popular among nonparametric learning methods, but despite excellent theoretical properties, they have limited applications in large scale learning because of time and memory requirements, typically at least quadratic in the number of data points.
Overcoming these limitations has motivated a variety of practical approaches including gradient methods, as well accelerated, stochastic and preconditioned extensions, to improve time complexity [1, 2, 3, 4, 5, 6]. Random projections provide an approach to reduce memory requirements, popular methods including Nyström [7, 8], random features [9], and their numerous extensions. From a theoretical perspective a key question has become to characterize statistical and computational tradeoffs, that is if, or under which conditions, computational gains come at the expense of statistical accuracy. In particular, recent results considering least squares, show that there are large class of problems for which, by combining Nyström or random features approaches [10, 11, 12, 13, 14, 15] with ridge regression, it is possible to substantially reduce computations, while preserving the same optimal statistical accuracy of exact kernel ridge regression (KRR). While statistical lower bounds exist for this setting, there are no corresponding computational lower bounds. The state of the art approximation of KRR, for which optimal statistical bounds are known, typically requires complexities that are roughly $O(n^2)$ in time and memory (or possibly $O(n)$ in memory, if kernel computations are made on the fly).

In this paper, we propose and study FALKON, a new algorithm that, to the best of our knowledge, has the best known theoretical guarantees. At the same time FALKON provides an efficient approach to apply kernel methods on millions of points, and tested on a variety of large scale problems

*E-mail: alessandro.rudi@inria.fr. This work was done when A.R. was working at Laboratory of Computational and Statistical Learning (Istituto Italiano di Tecnologia).

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

outperform previously proposed methods while utilizing only a fraction of computational resources. More precisely, we take a substantial step in provably reducing the computational requirements, showing that, up to logarithmic factors, a time/memory complexity of $O(n\sqrt{n})$ and $O(n)$ is sufficient for optimal statistical accuracy. Our new algorithm, exploits the idea of using Nyström methods to approximate the KRR problem, but also to efficiently compute a preconditioning to be used in conjugate gradient. To the best of our knowledge this is the first time all these ideas are combined and put to fruition. Our theoretical analysis derives optimal statistical rates both in a basic setting and under benign conditions for which fast rates are possible. The potential benefits of different sampling strategies are also analyzed. Most importantly, the empirical performances are thoroughly tested on available large scale data-sets. Our results show that, even on a single machine, FALKON can outperforms state of the art methods on most problems both in terms of time efficiency and prediction accuracy. In particular, our results suggest that FALKON could be a viable kernel alternative to deep fully connected neural networks for large scale problems.

The rest of the paper is organized as follows. In Sect. 2 we give some background on kernel methods. In Sect. 3 we introduce FALKON, while in Sect. 4 we present and discuss the main technical results. Finally in Sect. 5 we present experimental results.

2 Statistical and Computational Trade-offs in Kernel Methods

We consider the supervised learning problem of estimating a function from random noisy samples. In statistical learning theory, this can be formalized as the problem of solving

$$\inf_{f\in\mathcal{H}} \mathcal{E}(f), \qquad \mathcal{E}(f) = \int (f(x)-y)^2 d\rho(x,y), \tag{1}$$

given samples $(x_i, y_i)_{i=1}^n$ from ρ, which is fixed but unknown and where, $\mathcal{H}$ is a space of candidate solutions. Ideally, a good empirical solution $\widehat{f}$ should have small *excess risk*

$$\mathcal{R}(\widehat{f}\,) \;=\; \mathcal{E}(\widehat{f}\,) \;-\; \inf_{f\in\mathcal{H}} \mathcal{E}(f), \tag{2}$$

since this implies it will generalize/predict well new data. In this paper, we are interested in both computational and statistical aspects of the above problem. In particular, we investigate the computational resources needed to achieve optimal statistical accuracy, i.e. minimal excess risk. Our focus is on the most popular class of nonparametric methods, namely kernel methods.

Kernel methods and ridge regression. Kernel methods consider a space $\mathcal{H}$ of functions

$$f(x) = \sum_{i=1}^{n} \alpha_j K(x, x_i), \tag{3}$$

where K is a positive definite kernel [2]. The coefficients $\alpha_1, \dots, \alpha_n$ are typically derived from a convex optimization problem, that for the square loss is

$$\widehat{f}_{n,\lambda} = \operatorname*{argmin}_{f\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} (f(x_i)-y_i)^2 + \lambda \|f\|_{\mathcal{H}}^2, \tag{4}$$

and defines the so called kernel ridge regression (KRR) estimator [16]. An advantage of least squares approaches is that they reduce computations to a linear system

$$(K_{nn} + \lambda n I)\,\alpha = \widehat{y}, \tag{5}$$

where K_{nn} is an $n \times n$ matrix defined by $(K_{nn})_{ij} = K(x_i, x_j)$ and $\widehat{y} = (y_1, \dots y_n)$. We next comment on computational and statistical properties of KRR.

Computations. Solving Eq. (5) for large datasets is challenging. A direct approach requires $O(n^2)$ in space, to allocate K_{nn}, $O(n^2)$ kernel evaluations, and $O(n^2 c_K + n^3)$ in time, to compute and invert K_{nn} (c_K is the kernel evaluation cost assumed constant and omitted throughout).

Statistics. Under basic assumptions, KRR achieves an error $\mathcal{R}(\widehat{f}_{\lambda_n}) = O(n^{-1/2})$, for $\lambda_n = n^{-1/2}$, which is optimal in a minimax sense and can be improved *only* under more stringent assumptions [17, 18].

[2] K is positive definite, if the matrix with entries $K(x_i, x_j)$ is positive semidefinite $\forall x_1, \dots, x_N, N \in \mathbb{N}$ [16]

The question is then if it is possible to achieve the statistical properties of KRR, with less computations.

Gradient methods and early stopping. A natural idea is to consider iterative solvers and in particular gradient methods, because of their simplicity and low iteration cost. A basic example is computing the coefficients in (3) by

$$\alpha_t = \alpha_{t-1} + \tau \left[(K_{nn}\alpha_{t-1} - \widehat{y}) + \lambda n \alpha_{t-1} \right], \tag{6}$$

for a suitable step-size choice τ.

Computations. In this case, if t is the number of iterations, gradient methods require $O(n^2 t)$ in time, $O(n^2)$ in memory and $O(n^2)$ in kernel evaluations, if the kernel matrix is stored. Note that, the kernel matrix can also be computed on the fly with only $O(n)$ memory, but $O(n^2 t)$ kernel evaluations are required. We note that, beyond the above simple iteration, several variants have been considered including accelerated [1, 19] and stochastic extensions [20].

Statistics. The statistical properties of iterative approaches are well studied and also in the case where λ is set to zero, and regularization is performed by choosing a suitable stopping time [21]. In this latter case, the number of iterations can roughly be thought of $1/\lambda$ and $O(\sqrt{n})$ iterations are needed for basic gradient descent, $O(n^{1/4})$ for accelerated methods and possible $O(1)$ iterations/epochs for stochastic methods. Importantly, we note that unlike most optimization studies, here we are considering the number of iterations needed to solve (1), rather than (4).

While the time complexity of these methods dramatically improves over KRR, and computations can be done in blocks, memory requirements (or number of kernel evaluations) still makes the application to large scale setting cumbersome. Randomization provides an approach to tackle this challenge.

Random projections. The rough idea is to use random projections to compute K_{nn} only approximately. The most popular examples in this class of approaches are Nyström [7, 8] and random features [9] methods. In the following we focus in particular on a basic Nyström approach based on considering functions of the form

$$\widetilde{f}_{\lambda,M}(x) = \sum_{i=1}^{M} \widetilde{\alpha}_i K(x, \widetilde{x}_i), \quad \text{with} \quad \{\widetilde{x}_1, \dots, \widetilde{x}_M\} \subseteq \{x_1, \dots, x_n\}, \tag{7}$$

defined considering only a subset of M training points sampled uniformly. In this case, there are only M coefficients that, following the approach in (4), can be derived considering the linear system

$$H\widetilde{\alpha} = z, \qquad \text{where} \quad H = K_{nM}^\top K_{nM} + \lambda n K_{MM}, \quad z = K_{nM}^\top \hat{y}. \tag{8}$$

Here K_{nM} is the $n \times M$ matrix with $(K_{nM})_{ij} = K(x_i, \widetilde{x}_j)$ and K_{MM} is the $M \times M$ matrix with $(K_{MM})_{ij} = K(\widetilde{x}_i, \widetilde{x}_j)$. This method consists in subsampling the columns of K_{nn} and can be seen as a particular form of random projections.

Computations. Direct methods for solving (8) require $O(nM^2)$ in time to form $K_{nM}^\top K_{nM}$ and $O(M^3)$ for solving the linear system, and only $O(nM)$ kernel evaluations. The naive memory requirement is $O(nM)$ to store K_{nM}, however if $K_{nM}^\top K_{nM}$ is computed in blocks of dimension at most $M \times M$ only $O(M^2)$ memory is needed. Iterative approaches as in (6) can also be combined with random projections [22, 23, 24] to slightly reduce time requirements (see Table. 1, or Sect. F in the appendix, for more details).

Statistics. The key point though, is that random projections allow to dramatically reduce memory requirements as soon as $M \ll n$ and the question arises of whether this comes at expenses of statistical accuracy. Interestingly, recent results considering this question show that there are large classes of problems for which $M = \tilde{O}(\sqrt{n})$ suffices for the same optimal statistical accuracy of the exact KRR [11, 12, 13].

In summary, in this case the computations needed for optimal statistical accuracy are reduced from $O(n^2)$ to $O(n\sqrt{n})$ kernel evaluations, but the best time complexity is basically $O(n^2)$. In the rest of the paper we discuss how this requirement can indeed be dramatically reduced.

3 FALKON

Our approach is based on a novel combination of randomized projections with iterative solvers plus preconditioning. The main novelty is that we use random projections to approximate both the problem *and* the preconditioning.

Preliminaries: preconditioning and KRR. We begin recalling the basic idea behind preconditioning. The key quantity is the condition number, that for a linear system is the ratio between the largest and smallest singular values of the matrix defining the problem [25]. For example, for problem (5) the condition number is given by

$$\text{cond}(K_{nn} + \lambda n I) = (\sigma_{\max} + \lambda n)/(\sigma_{\min} + \lambda n),$$

with $\sigma_{\max}, \sigma_{\min}$ largest and smallest eigenvalues of K_{nn}, respectively. The importance of the condition number is that it captures the time complexity of iteratively solving the corresponding linear system. For example, if a simple gradient descent (6) is used, the number of iterations needed for an ϵ accurate solution of problem (5) is

$$t = O(\text{cond}(K_{nn} + \lambda n I)\log(1/\epsilon)).$$

It is shown in [23] that in this case $t = \sqrt{n}\log n$ are needed to achieve a solution with good statistical properties. Indeed, it can be shown that roughly $t \approx 1/\lambda \log(\frac{1}{\epsilon})$ are needed where $\lambda = 1/\sqrt{n}$ and $\epsilon = 1/n$. The idea behind preconditioning is to use a suitable matrix B to define an equivalent linear system with better condition number. For (5), an ideal choice is B such that

$$BB^\top = (K_{nn} + \lambda n I)^{-1} \tag{9}$$

and $B^\top(K_{nn} + \lambda n I)B\,\beta \;=\; B^\top \hat{y}$. Clearly, if β_* solves the latter problem, $\alpha_* = B\beta_*$ is a solution of problem (5). Using a preconditioner B as in (9) one iteration is sufficient, but computing the B is typically as hard as the original problem. The problem is to derive preconditioning such that (9) might hold only approximately, but that can be computed efficiently. Derivation of efficient preconditioners for the exact KRR problem (5) has been the subject of recent studies, [3, 4, 26, 5, 6]. In particular, [4, 26, 5, 6] consider random projections to approximately compute a preconditioner. Clearly, while preconditioning (5) leads to computational speed ups in terms of the number of iterations, requirements in terms of memory/kernel evaluation are the same as standard kernel ridge regression.

The key idea to tackle this problem is to consider an efficient preconditioning approach for problem (8) rather than (5).

Basic FALKON algorithm. We begin illustrating a basic version of our approach. The key ingredient is the following preconditioner for Eq. (8),

$$BB^\top = \Big(\frac{n}{M}K_{MM}^2 + \lambda n K_{MM}\Big)^{-1}, \tag{10}$$

which is itself based on a Nyström approximation[3]. The above preconditioning is a natural approximation of the ideal preconditioning of problem (8) that is $BB^\top = (K_{nM}^\top K_{nM} \;+\; \lambda n K_{MM})^{-1}$ and reduces to it if $M = n$. Our theoretical analysis, shows that $M \ll n$ suffices for deriving optimal statistical rates. In its basic form FALKON is derived combining the above preconditioning and gradient descent,

$$\widehat{f}_{\lambda,M,t}(x) = \sum_{i=1}^{M} \alpha_{t,i} K(x, \widetilde{x}_i), \quad \text{with} \quad \alpha_t = B\beta_t \quad \text{and} \tag{11}$$

$$\beta_k = \beta_{k-1} - \frac{\tau}{n} B^\top \left[K_{nM}^\top (K_{nM}(B\beta_{k-1}) - \widehat{y}) \;+\; \lambda n K_{MM}(B\beta_{k-1})\right], \tag{12}$$

for $t \in \mathbb{N}$, $\beta_0 = 0$ and $1 \le k \le t$ and a suitable chosen τ. In practice, a refined version of FALKON is preferable where a faster gradient iteration is used and additional care is taken in organizing computations.

FALKON. The actual version of FALKON we propose is Alg. 1 (see Sect. A, Alg. 2 for the complete algorithm). It consists in solving the system $B^\top H B\beta = B^\top z$ via conjugate gradient [25], since it is a fast gradient method and does not require to specify the step-size. Moreover, to compute B quickly, with reduced numerical errors, we consider the following strategy

$$B = \frac{1}{\sqrt{n}} T^{-1} A^{-1}, \qquad T = \text{chol}(K_{MM}), \quad A = \text{chol}\left(\frac{1}{M}\, T\, T^\top + \lambda I\right), \tag{13}$$

where chol() is the Cholesky decomposition (in Sect. A the strategy for non invertible K_{MM}).

[3] For the sake of simplicity, here we assume K_{MM} to be invertible and the Nyström centers selected with uniform sampling from the training set, see Sect. A and Alg. 2 in the appendix for the general algorithm.

Algorithm 1 MATLAB code for FALKON. It requires $O(nMt + M^3)$ in time and $O(M^2)$ in memory. See Sect. A and Alg. 2 in the appendixes for the complete algorithm.

Input: Dataset $X = (x_i)_{i=1}^n \in \mathbb{R}^{n\times D}$, $\hat{y} = (y_i)_{i=1}^n \in \mathbb{R}^n$, centers $C = (\tilde{x}_j)_{j=1}^M \in \mathbb{R}^{M\times D}$, KernelMatrix computing the kernel matrix given two sets of points, regularization parameter λ, number of iterations t.
Output: Nyström coefficients α.

```
function alpha = FALKON(X, C, Y, KernelMatrix, lambda, t)
     n = size(X,1); M = size(C,1); KMM = KernelMatrix(C,C);
     T = chol(KMM + eps*M*eye(M));
     A = chol(T*T'/M + lambda*eye(M));

     function w = KnM_times_vector(u, v)
        w = zeros(M,1); ms = ceil(linspace(0, n, ceil(n/M)+1));
        for i=1:ceil(n/M)
           Kr = KernelMatrix( X(ms(i)+1:ms(i+1),:), C );
           w = w + Kr'*(Kr*u + v(ms(i)+1:ms(i+1),:));
        end
     end

     BHB = @(u) A'\(T'\(KnM_times_vector(T\(A\u), zeros(n,1))/n) + lambda*(A\u));
     r = A'\(T'\KnM_times_vector(zeros(M,1), Y/n));
     alpha = T\(A\conjgrad(BHB, r, t));
 end
```

Computations. in Alg. 1, B is never built explicitly and A, T are two upper-triangular matrices, so $A^{-\top}u$, $A^{-1}u$ for a vector u costs M^2, and the same for T. The cost of computing the preconditioner is only $\frac{4}{3}M^3$ floating point operations (consisting in two Cholesky decompositions and one product of two triangular matrices). Then FALKON requires $O(nMt + M^3)$ in time and the same $O(M^2)$ memory requirement of the basic Nyström method, if matrix/vector multiplications at each iteration are performed in blocks. This implies $O(nMt)$ kernel evaluations are needed.

The question remains to characterize M and the number of iterations needed for good statistical accuracy. Indeed, in the next section we show that roughly $O(n\sqrt{n})$ computations and $O(n)$ memory are sufficient for optimal accuracy. This implies that FALKON is currently the most efficient kernel method with the same optimal statistical accuracy of KRR, see Table 1.

4 Theoretical Analysis

In this section, we characterize the generalization properties of FALKON showing it achieves the optimal generalization error of KRR, with dramatically reduced computations. This result is given in Thm. 3 and derived in two steps. First, we study the difference between the excess risk of FALKON and that of the basic Nyström (8), showing it depends on the condition number induced by the preconditioning, hence on M (see Thm.1). Deriving these results requires some care, since differently to standard optimization results, our goal is to solve (1) i.e. achieve small excess risk, not to minimize the empirical error. Second, we show that choosing $M = \widetilde{O}(1/\lambda)$ allows to make this difference as small as $e^{-t/2}$ (see Thm.2). Finally, recalling that the basic Nyström for $\lambda = 1/\sqrt{n}$ has essentially the same statistical properties of KRR [13], we answer the question posed at the end of the last section and show that roughly $\log n$ iterations are sufficient for optimal statistical accuracy. Following the discussion in the previous section this means that the computational requirements for optimal accuracy are $\widetilde{O}(n\sqrt{n})$ in time/kernel evaluations and $\widetilde{O}(n)$ in space. Later in this section faster rates under further regularity assumptions are also derived and the effect of different selection methods for the Nyström centers considered. The proofs for this section are provided in Sect. E of the appendixes.

4.1 Main Result

The first result is interesting in its own right since it corresponds to translating optimization guarantees into statistical results. In particular, we derive a relation the excess risk of the FALKON algorithm $\widehat{f}_{\lambda,M,t}$ from Alg. 1 and the Nyström estimator $\widetilde{f}_{\lambda,M}$ from Eq. (8) with uniform sampling.

Algorithm	train time	kernel evaluations	memory	test time
SVM / KRR + direct method	n^3	n^2	n^2	n
KRR + iterative [1, 2]	$n^2\sqrt[4]{n}$	n^2	n^2	n
Doubly stochastic [22]	$n^2\sqrt{n}$	$n^2\sqrt{n}$	n	n
Pegasos / KRR + sgd [27]	n^2	n^2	n	n
KRR + iter + precond [3, 28, 4, 5, 6]	n^2	n^2	n	n
Divide & Conquer [29]	n^2	$n\sqrt{n}$	n	n
Nyström, random features [7, 8, 9]	n^2	$n\sqrt{n}$	n	$\sqrt{n}$
Nyström + iterative [23, 24]	n^2	$n\sqrt{n}$	n	$\sqrt{n}$
Nyström + sgd [20]	n^2	$n\sqrt{n}$	n	$\sqrt{n}$
FALKON (see Thm. 3)	$\boldsymbol{n\sqrt{n}}$	$\boldsymbol{n\sqrt{n}}$	$\boldsymbol{n}$	$\boldsymbol{\sqrt{n}}$

Table 1: Computational complexity required by different algorithms, for optimal generalization. Logarithmic terms are not showed.

Theorem 1. *Let $n, M \geq 3$, $t \in \mathbb{N}$, $0 < \lambda \leq \lambda_1$ and $\delta \in (0, 1]$. Assume there exists $\kappa \geq 1$ such that $K(x, x) \leq \kappa^2$ for any $x \in X$. Then, the following inequality holds with probability $1 - \delta$*

$$\mathcal{R}(\widehat{f}_{\lambda,M,t})^{1/2} \;\leq\; \mathcal{R}(\widetilde{f}_{\lambda,M})^{1/2} \;+\; 4\widehat{v}\, e^{-\nu t} \sqrt{1 + \frac{9\kappa^2}{\lambda n} \log \frac{n}{\delta}},$$

where $\widehat{v}^2 = \frac{1}{n}\sum_{i=1}^{n} y_i^2$ and $\nu = \log(1 + 2/(\operatorname{cond}(B^\top H B)^{1/2} - 1))$, with $\operatorname{cond}(B^\top H B)$ the condition number of $B^\top H B$. Note that $\lambda_1 > 0$ is a constant not depending on λ, n, M, δ, t.

The additive term in the bound above decreases exponentially in the number of iterations. If the condition number of $B^\top H B$ is smaller than a small universal constant (e.g. 17), then $\nu > 1/2$ and the additive term decreases as $e^{-\frac{t}{2}}$. Next, theorems derive a condition on M that allows to control $\operatorname{cond}(B^\top H B)$, and derive such an exponential decay.

Theorem 2. *Under the same conditions of Thm. 1, if*

$$M \geq 5 \left[1 + \frac{14\kappa^2}{\lambda}\right] \log \frac{8\kappa^2}{\lambda\delta}.$$

then the exponent ν in Thm. 1 satisfies $\nu \geq 1/2$.

The above result gives the desired exponential bound showing that after $\log n$ iterations the excess risk of FALKON is controlled by that of the basic Nyström, more precisely

$$\mathcal{R}(\widehat{f}_{\lambda,M,t}) \leq 2\mathcal{R}(\widetilde{f}_{\lambda,M}) \quad \text{when} \quad t \geq \log \mathcal{R}(\widetilde{f}_{\lambda,M}) \;+\; \log\left(1 + \frac{9\kappa^2}{\lambda n} \log \frac{n}{\delta}\right) \;+\; \log\left(16\widehat{v}^2\right).$$

Finally, we derive an excess risk bound for FALKON. By the no-free-lunch theorem, this requires some conditions on the learning problem. We first consider a standard basic setting where we only assume it exists $f_{\mathcal{H}} \in \mathcal{H}$ such that $\mathcal{E}(f_{\mathcal{H}}) = \inf_{f \in \mathcal{H}} \mathcal{E}(f)$.

Theorem 3. *Let $\delta \in (0, 1]$. Assume there exists $\kappa \geq 1$ such that $K(x, x) \leq \kappa^2$ for any $x \in X$, and $y \in [-\frac{a}{2}, \frac{a}{2}]$, almost surely, $a > 0$. There exist $n_0 \in \mathbb{N}$ such that for any $n \geq n_0$, if*

$$\lambda = \frac{1}{\sqrt{n}}, \qquad M \geq 75\sqrt{n}\,\log\frac{48\kappa^2 n}{\delta}, \qquad t \;\geq\; \frac{1}{2}\log(n) \;+\; 5 + 2\log(a + 3\kappa),$$

then with probability $1 - \delta$,

$$\mathcal{R}(\widehat{f}_{\lambda,M,t}) \leq \frac{c_0 \log^2 \frac{24}{\delta}}{\sqrt{n}}.$$

In particular n_0, c_0 do not depend on λ, M, n, t and c_0 do not depend on δ.

The above result provides the desired bound, and all the constants are given in the appendix. The obtained learning rate is the same as the full KRR estimator and is known to be optimal in a minmax sense [17], hence not improvable. As mentioned before, the same bound is also achieved by the

basic Nyström method but with much worse time complexity. Indeed, as discussed before, using a simple iterative solver typically requires $O(\sqrt{n}\log n)$ iterations, while we need only $O(\log n)$. Considering the choice for M this leads to a computational time of $O(nMt) = O(n\sqrt{n})$ for optimal generalization (omitting logarithmic terms). To the best of our knowledge FALKON currently provides the best time/space complexity to achieve the statistical accuracy of KRR. Beyond the basic setting considered above, in the next section we show that FALKON can achieve much faster rates under refined regularity assumptions and also consider the potential benefits of leverage score sampling.

4.2 Fast learning rates and Nyström with approximate leverage scores

Considering fast rates and Nyström with more general sampling is considerably more technical and a heavier notation is needed. Our analysis apply to any approximation scheme (e.g. [30, 12, 31]) satisfying the definition of q-approximate leverage scores [13], satisfying $q^{-1}l_i(\lambda) \leq \widehat{l}_i(\lambda) \leq ql_i(\lambda), \quad \forall\, i \in \{1,\dots,n\}$. Here $\lambda > 0$, $l_i(\lambda) = (K_{nn}(K_{nn}+\lambda nI)^{-1})_{ii}$ are the leverage scores and $q \geq 1$ controls the quality of the approximation. In particular, given λ, the Nyström points are sampled independently from the dataset with probability $p_i \propto \widehat{l}_i(\lambda)$. We need a few more definitions. Let $K_x = K(x,\cdot)$ for any $x \in X$ and $\mathcal{H}$ the reproducing kernel Hilbert space [32] of functions with inner product defined by $\mathcal{H} = \overline{\operatorname{span}\{K_x \mid x \in X\}}$ and closed with respect to the inner product $\langle\cdot,\cdot\rangle_{\mathcal{H}}$ defined by $\langle K_x, K_{x'}\rangle_{\mathcal{H}} = K(x,x')$, for all $x, x' \in X$. Define $C : \mathcal{H} \to \mathcal{H}$ to be the linear operator $\langle f, Cg\rangle_{\mathcal{H}} = \int_X f(x)g(x)d\rho_X(x)$, for all $f, g \in \mathcal{H}$. Finally define the following quantities,

$$\mathcal{N}_\infty(\lambda) = \sup_{x\in X} \|(C+\lambda I)^{-1/2}K_x\|_{\mathcal{H}}, \quad \mathcal{N}(\lambda) = \operatorname{Tr}(C(C+\lambda I)^{-1}).$$

The latter quantity is known as *degrees of freedom* or *effective dimension*, can be seen as a measure of the *size* of $\mathcal{H}$. The quantity $\mathcal{N}_\infty(\lambda)$ can be seen to provide a uniform bound on the leverage scores. In particular note that $\mathcal{N}(\lambda) \leq \mathcal{N}_\infty(\lambda) \leq \frac{\kappa^2}{\lambda}$ [13]. We can now provide a refined version of Thm. 2.

Theorem 4. *Under the same conditions of Thm. 1, the exponent ν in Thm. 1 satisfies $\nu \geq 1/2$, when*

1. *either Nyström uniform sampling is used with $M \geq 70\,[1+\mathcal{N}_\infty(\lambda)]\log\frac{8\kappa^2}{\lambda\delta}$.*
2. *or Nyström q-approx. lev. scores [13] is used, with $\lambda \geq \frac{19\kappa^2}{n}\log\frac{n}{2\delta}$, $n \geq 405\kappa^2\log\frac{12\kappa^2}{\delta}$,*

$$M \geq 215\left[2+q^2\mathcal{N}(\lambda)\right]\log\frac{8\kappa^2}{\lambda\delta}.$$

We then recall the standard, albeit technical, assumptions leading to fast rates [17, 18]. The *capacity condition* requires the existence of $\gamma \in (0,1]$ and $Q \geq 0$, such that $\mathcal{N}(\lambda) \leq Q^2\lambda^{-\gamma}$. Note that this condition is always satisfied with $Q = \kappa$ and $\gamma = 1$. The *source condition* requires the existence of $r \in [1/2, 1]$ and $g \in \mathcal{H}$, such that $f_{\mathcal{H}} = C^{r-1/2}g$. Intuitively, the *capacity condition* measures the size of $\mathcal{H}$, if γ is small then $\mathcal{H}$ is small and rates are faster. The *source condition* measures the regularity of $f_{\mathcal{H}}$, if r is big $f_{\mathcal{H}}$ is regular and rates are faster. The case $r = 1/2$ and $\gamma = D/(2s)$ (for a kernel with smoothness s and input space $\mathbb{R}^D$) recovers the classic Sobolev condition. For further discussions on the interpretation of the conditions above see [17, 18, 11, 13]. We can then state our main result on fast rates

Theorem 5. *Let $\delta \in (0,1]$. Assume there exists $\kappa \geq 1$ such that $K(x,x) \leq \kappa^2$ for any $x \in X$, and $y \in [-\frac{a}{2}, \frac{a}{2}]$, almost surely, with $a > 0$. There exist an $n_0 \in \mathbb{N}$ such that for any $n \geq n_0$ the following holds. When*

$$\lambda = n^{-\frac{1}{2r+\gamma}}, \qquad t \;\geq\; \log(n) \;+\; 5 + 2\log(a+3\kappa^2),$$

1. *and either Nyström uniform sampling is used with $M \geq 70\,[1+\mathcal{N}_\infty(\lambda)]\log\frac{8\kappa^2}{\lambda\delta}$,*
2. *or Nyström q-approx. lev. scores [13] is used with $M \geq 220\left[2+q^2\mathcal{N}(\lambda)\right]\log\frac{8\kappa^2}{\lambda\delta}$,*

then with probability $1-\delta$,

$$\mathcal{R}(\widehat{f}_{\lambda,M,t}) \;\leq\; c_0\log^2\frac{24}{\delta}\;\; n^{-\frac{2r}{2r+\gamma}}.$$

where $\widehat{f}_{\lambda,M,t}$ is the FALKON estimator (Sect. 3, Alg. 1 and Sect. A, Alg. 2 in the appendix for the complete version). In particular n_0, c_0 do not depend on λ, M, n, t and c_0 do not depend on δ.

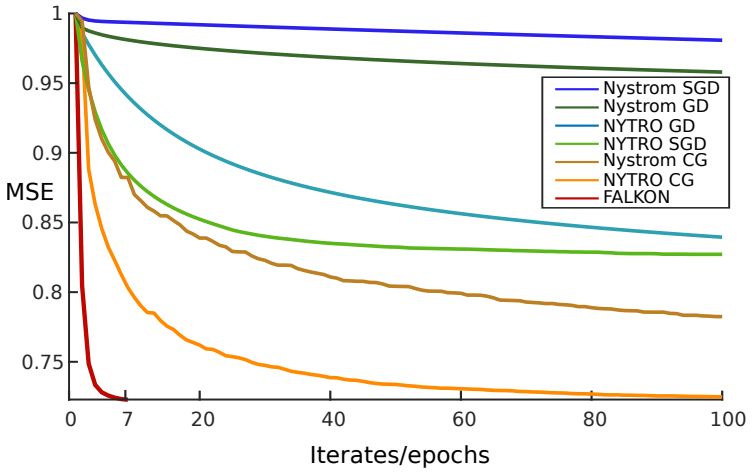

Figure 1: Falkon is compared to stochastic gradient, gradient descent and conjugate gradient applied to Problem (8), while NYTRO refer to the variants described in [23]. The graph shows the test error on the HIGGS dataset (1.1×10^7 examples) with respect to the number of iterations (epochs for stochastic algorithms).

The above result shows that FALKON achieves the same fast rates as KRR, under the same conditions [17]. For $r = 1/2, \gamma = 1$, the rate in Thm. 3 is recovered. If $\gamma < 1, r > 1/2$, FALKON achieves a rate close to $O(1/n)$. By selecting the Nyström points with uniform sampling, a bigger M could be needed for fast rates (albeit always less than n). However, when approximate leverage scores are used M, smaller than $n^{\gamma/2} \ll \sqrt{n}$ is always enough for optimal generalization. This shows that FALKON with approximate leverage scores is the first algorithm to achieve fast rates with a computational complexity that is $O(n\mathcal{N}(\lambda)) = O(n^{1+\frac{\gamma}{2r+\gamma}}) \leq O(n^{1+\frac{\gamma}{2}})$ in time.

5 Experiments

We present FALKON's performance on a range of large scale datasets. As shown in Table 2, 3, FALKON achieves state of the art accuracy and typically outperforms previous approaches in all the considered large scale datasets including IMAGENET. This is remarkable considering FALKON required only a fraction of the competitor's computational resources. Indeed we used a single machine equipped with two Intel Xeon E5-2630 v3, one NVIDIA Tesla K40c and 128 GB of RAM and a basic MATLAB FALKON implementation, while typically the results for competing algorithm have been performed on clusters of GPU workstations (accuracies, times and used architectures are cited from the corresponding papers).
A minimal MATLAB implementation of FALKON is presented in Appendix G. The code necessary to reproduce the following experiments, plus a FALKON version that is able to use the GPU, is available on GitHub at `https://github.com/LCSL/FALKON_paper` . The error is measured with MSE, RMSE or relative error for regression problems, and with classification error (c-err) or AUC for the classification problems, to be consistent with the literature. For datasets which do not have a fixed test set, we set apart 20% of the data for testing. For all datasets, but YELP and IMAGENET, we normalize the features by their z-score. From now on we denote with n the cardinality of the dataset, d the dimensionality. A comparison of FALKON with respect to other methods to compute the Nyström estimator, in terms of the MSE test error on the HIGGS dataset, is given in Figure 1.

MillionSongs [36] (Table 2, $n = 4.6 \times 10^5$, $d = 90$, regression). We used a Gaussian kernel with $\sigma = 6$, $\lambda = 10^{-6}$ and 10^4 Nyström centers. Moreover with 5×10^4 center, FALKON achieves a 79.20 MSE, and 4.49×10^{-3} rel. error in 630 *sec*.

TIMIT (Table 2, $n = 1.2 \times 10^6$, $d = 440$, multiclass classification). We used the same preprocessed dataset of [6] and Gaussian Kernel with $\sigma = 15$, $\lambda = 10^{-9}$ and 10^5 Nyström centers.

YELP (Table 2, $n = 1.5 \times 10^6$, $d = 6.52 \times 10^7$, regression). We used the same dataset of [24]. We extracted the 3-grams from the plain text with the same pipeline as [24], then we mapped

them in a sparse binary vector which records if the 3-gram is present or not in the example. We used a linear kernel with 5×10^4 Nyström centers. With 10^5 centers, we get a RMSE of 0.828 in 50 minutes.

Table 2: Architectures: ‡ cluster 128 EC2 r3.2xlarge machines, † cluster 8 EC2 r3.8xlarge machines, ≀ single machine with two Intel Xeon E5-2620, one Nvidia GTX Titan X GPU, 128GB RAM, ⋆ cluster with IBM POWER8 12-core processor, 512 GB RAM, ∗ unknown platform.

	MillionSongs			YELP		TIMIT	
	MSE	Relative error	Time(s)	RMSE	Time(m)	c-err	Time(h)
FALKON	**80.10**	$\mathbf{4.51 \times 10^{-3}}$	**55**	**0.833**	**20**	32.3%	**1.5**
Prec. KRR [4]	-	4.58×10^{-3}	$289^{\dagger}$	-	-	-	-
Hierarchical [33]	-	4.56×10^{-3}	$293^{\star}$	-	-	-	-
D&C [29]	80.35	-	737^{*}	-	-	-	-
Rand. Feat. [29]	80.93	-	772^{*}	-	-	-	-
Nyström [29]	80.38	-	876^{*}	-	-	-	-
ADMM R. F.[4]	-	5.01×10^{-3}	$958^{\dagger}$	-	-	-	-
BCD R. F. [24]	-	-	-	0.949	$42^{\ddagger}$	34.0%	$1.7^{\ddagger}$
BCD Nyström [24]	-	-	-	0.861	$60^{\ddagger}$	33.7%	$1.7^{\ddagger}$
EigenPro [6]	-	-	-	-	-	32.6%	$3.9^{\wr}$
KRR [33] [24]	-	4.55×10^{-3}	-	0.854	$500^{\ddagger}$	33.5%	$8.3^{\ddagger}$
Deep NN [34]	-	-	-	-	-	32.4%	-
Sparse Kernels [34]	-	-	-	-	-	**30.9%**	-
Ensemble [35]	-	-	-	-	-	33.5%	-

Table 3: Architectures: † cluster with IBM POWER8 12-core cpu, 512 GB RAM, ≀ single machine with two Intel Xeon E5-2620, one Nvidia GTX Titan X GPU, 128GB RAM, ‡ single machine [37]

	SUSY			HIGGS		IMAGENET	
	c-err	AUC	Time(m)	AUC	Time(h)	c-err	Time(h)
FALKON	**19.6%**	0.877	**4**	0.833	**3**	20.7%	**4**
EigenPro [6]	19.8%	-	$6^{\wr}$	-	-	-	-
Hierarchical [33]	20.1%	-	$40^{\dagger}$	-	-	-	-
Boosted Decision Tree [38]	-	0.863	-	0.810	-	-	-
Neural Network [38]	-	0.875	-	0.816	-	-	-
Deep Neural Network [38]	-	**0.879**	$4680^{\ddagger}$	**0.885**	$78^{\ddagger}$	-	-
Inception-V4 [39]	-	-	-	-	-	**20.0%**	-

SUSY (Table 3, $n = 5 \times 10^6$, $d = 18$, binary classification). We used a Gaussian kernel with $\sigma = 4$, $\lambda = 10^{-6}$ and 10^4 Nyström centers.

HIGGS (Table 3, $n = 1.1 \times 10^7$, $d = 28$, binary classification). Each feature has been normalized subtracting its mean and dividing for its variance. We used a Gaussian kernel with diagonal matrix width learned with cross validation on a small validation set, $\lambda = 10^{-8}$ and 10^5 Nyström centers. If we use a single $\sigma = 5$ we reach an AUC of 0.825.

IMAGENET (Table 3, $n = 1.3 \times 10^6$, $d = 1536$, multiclass classification). We report the top 1 c-err over the validation set of ILSVRC 2012 with a single crop. The features are obtained from the convolutional layers of pre-trained Inception-V4 [39]. We used Gaussian kernel with $\sigma = 19$, $\lambda = 10^{-9}$ and 5×10^4 Nyström centers. Note that with linear kernel we achieve c-err $= 22.2\%$.

Acknowledgments.
The authors would like to thank Mikhail Belkin, Benjamin Recht and Siyuan Ma, Eric Fosler-Lussier, Shivaram Venkataraman, Stephen L. Tu, for providing their features of the TIMIT and YELP datasets, and NVIDIA Corporation for the donation of the 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 FIRB project RBFR12M3AC (Italian Ministry of Education, University and Research).

References

[1] A. Caponnetto and Yuan Yao. Adaptive rates for regularization operators in learning theory. *Analysis and Applications*, 08, 2010.

[2] L. Lo Gerfo, Lorenzo Rosasco, Francesca Odone, Ernesto De Vito, and Alessandro Verri. Spectral Algorithms for Supervised Learning. *Neural Computation*, 20(7):1873–1897, 2008.

[3] Gregory E Fasshauer and Michael J McCourt. Stable evaluation of gaussian radial basis function interpolants. *SIAM Journal on Scientific Computing*, 34(2):A737–A762, 2012.

[4] Haim Avron, Kenneth L Clarkson, and David P Woodruff. Faster kernel ridge regression using sketching and preconditioning. *arXiv preprint arXiv:1611.03220*, 2016.

[5] Alon Gonen, Francesco Orabona, and Shai Shalev-Shwartz. Solving ridge regression using sketched preconditioned svrg. *arXiv preprint arXiv:1602.02350*, 2016.

[6] Siyuan Ma and Mikhail Belkin. Diving into the shallows: a computational perspective on large-scale shallow learning. *arXiv preprint arXiv:1703.10622*, 2017.

[7] Christopher Williams and Matthias Seeger. Using the Nyström Method to Speed Up Kernel Machines. In *NIPS*, pages 682–688. MIT Press, 2000.

[8] Alex J. Smola and Bernhard Schölkopf. Sparse Greedy Matrix Approximation for Machine Learning. In *ICML*, pages 911–918. Morgan Kaufmann, 2000.

[9] Ali Rahimi and Benjamin Recht. Random Features for Large-Scale Kernel Machines. In *NIPS*, pages 1177–1184. Curran Associates, Inc., 2007.

[10] 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.

[11] Francis Bach. Sharp analysis of low-rank kernel matrix approximations. In *COLT*, volume 30 of *JMLR Proceedings*, pages 185–209. JMLR.org, 2013.

[12] Ahmed Alaoui and Michael W Mahoney. Fast randomized kernel ridge regression with statistical guarantees. In *Advances in Neural Information Processing Systems 28*, pages 775–783. 2015.

[13] Alessandro Rudi, Raffaello Camoriano, and Lorenzo Rosasco. Less is more: Nyström computational regularization. In *Advances in Neural Information Processing Systems*, pages 1648–1656, 2015.

[14] Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. *arXiv preprint arXiv:1602.04474*, 2016.

[15] Francis Bach. On the equivalence between kernel quadrature rules and random feature expansions. *Journal of Machine Learning Research*, 18(21):1–38, 2017.

[16] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning)*. MIT Press, 2002.

[17] Andrea Caponnetto and Ernesto De Vito. Optimal rates for the regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368, 2007.

[18] Ingo Steinwart, Don R Hush, Clint Scovel, et al. Optimal rates for regularized least squares regression. In *COLT*, 2009.

[19] F. Bauer, S. Pereverzev, and L. Rosasco. On regularization algorithms in learning theory. *Journal of complexity*, 23(1):52–72, 2007.

[20] Aymeric Dieuleveut and Francis Bach. Non-parametric stochastic approximation with large step sizes. *arXiv preprint arXiv:1408.0361*, 2014.

[21] Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315, 2007.

[22] 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.

[23] Raffaello Camoriano, Tomás Angles, Alessandro Rudi, and Lorenzo Rosasco. Nytro: When subsampling meets early stopping. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pages 1403–1411, 2016.

[24] Stephen Tu, Rebecca Roelofs, Shivaram Venkataraman, and Benjamin Recht. Large scale kernel learning using block coordinate descent. *arXiv preprint arXiv:1602.05310*, 2016.

[25] Yousef Saad. *Iterative methods for sparse linear systems*. SIAM, 2003.

[26] Kurt Cutajar, Michael Osborne, John Cunningham, and Maurizio Filippone. Preconditioning kernel matrices. In *International Conference on Machine Learning*, pages 2529–2538, 2016.

[27] Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, 127(1):3–30, 2011.

[28] Yun Yang, Mert Pilanci, and Martin J Wainwright. Randomized sketches for kernels: Fast and optimal non-parametric regression. *arXiv preprint arXiv:1501.06195*, 2015.

[29] Yuchen Zhang, John C. Duchi, and Martin J. Wainwright. Divide and Conquer Kernel Ridge Regression. In *COLT*, volume 30 of *JMLR Proceedings*, pages 592–617. JMLR.org, 2013.

[30] Petros Drineas, Malik Magdon-Ismail, Michael W. Mahoney, and David P. Woodruff. Fast approximation of matrix coherence and statistical leverage. *JMLR*, 13:3475–3506, 2012.

[31] Michael B. Cohen, Yin Tat Lee, Cameron Musco, Christopher Musco, Richard Peng, and Aaron Sidford. Uniform Sampling for Matrix Approximation. In *ITCS*, pages 181–190. ACM, 2015.

[32] I. Steinwart and A. Christmann. *Support Vector Machines*. Information Science and Statistics. Springer New York, 2008.

[33] Jie Chen, Haim Avron, and Vikas Sindhwani. Hierarchically compositional kernels for scalable nonparametric learning. *CoRR*, abs/1608.00860, 2016.

[34] Avner May, Alireza Bagheri Garakani, Zhiyun Lu, Dong Guo, Kuan Liu, Aurelien Bellet, Linxi Fan, Michael Collins, Daniel J. Hsu, Brian Kingsbury, Michael Picheny, and Fei Sha. Kernel approximation methods for speech recognition. *CoRR*, abs/1701.03577, 2017.

[35] Po-Sen Huang, Haim Avron, Tara N. Sainath, Vikas Sindhwani, and Bhuvana Ramabhadran. Kernel methods match deep neural networks on timit. *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 205–209, 2014.

[36] Thierry Bertin-Mahieux, Daniel P. W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *ISMIR*, 2011.

[37] Alexandre Alves. Stacking machine learning classifiers to identify higgs bosons at the lhc. *CoRR*, abs/1612.07725, 2016.

[38] Pierre Baldi, Peter Sadowski, and Daniel Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nature communications*, 5, 2014.

[39] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. pages 4278–4284, 2017.

[40] Michael Reed and Barry Simon. *Methods of Modern Mathematical Physics: Vol.: 1.: Functional Analysis*. Academic press, 1980.

[41] Ernesto D Vito, Lorenzo Rosasco, Andrea Caponnetto, Umberto D Giovannini, and Francesca Odone. Learning from examples as an inverse problem. In *Journal of Machine Learning Research*, pages 883–904, 2005.

[42] Alessandro Rudi, Guillermo D Canas, and Lorenzo Rosasco. On the Sample Complexity of Subspace Learning. In *NIPS*, pages 2067–2075, 2013.

[43] Stéphane Boucheron, Gábor Lugosi, and Olivier Bousquet. Concentration inequalities. In *Advanced Lectures on Machine Learning*. 2004.