Low-Precision Random Fourier Features for Memory-Constrained Kernel Approximation

Jian Zhang* **Avner May*** **Tri Dao** **Christopher Ré**
Stanford University

Abstract

We investigate how to train kernel approximation methods that generalize well under a memory budget. Building on recent theoretical work, we define a measure of kernel approximation error which we find to be more predictive of the empirical generalization performance of kernel approximation methods than conventional metrics. An important consequence of this definition is that a kernel approximation matrix must be high rank to attain close approximation. Because storing a high-rank approximation is memory intensive, we propose using a *low-precision* quantization of random Fourier features (LP-RFFs) to build a high-rank approximation under a memory budget. Theoretically, we show quantization has a negligible effect on generalization performance in important settings. Empirically, we demonstrate across four benchmark datasets that LP-RFFs can match the performance of full-precision RFFs and the Nyström method, with 3x-10x and 50x-460x less memory, respectively.

1 INTRODUCTION

Kernel methods are a powerful family of machine learning methods. A key technique for scaling kernel methods is to construct feature representations whose inner products approximate the kernel function, and then learn a linear model with these features; important examples of this technique include the Nyström method (Williams and Seeger, 2000) and random Fourier features (RFFs) (Rahimi and Recht, 2007). Unfortunately, a large number of features are typically needed for attaining strong generalization performance with these methods on big datasets (Rahimi and Recht, 2008; Tu et al., 2016; May et al., 2017). Thus, the memory required to store these features can become the training bottleneck for kernel approximation models. In this paper we work to alleviate this memory bottleneck by optimizing the generalization performance for these methods under a fixed memory budget.

To gain insight into how to design more memory-efficient kernel approximation methods, we first investigate the generalization performance vs. memory utilization of Nyström and RFFs. While prior work (Yang et al., 2012) has shown that the Nyström method generalizes better than RFFs under the the same number of features, we demonstrate that the opposite is true under a memory budget. Strikingly, we observe that 50,000 standard RFFs can achieve the same held-out accuracy as 20,000 Nyström features with 10x less memory on the TIMIT classification task. Furthermore, this cannot be easily explained by the Frobenius or spectral norms of the kernel approximation error matrices of these methods, even though these norms are the most common metrics for evaluating kernel approximation methods (Gittens and Mahoney, 2016; Yang et al., 2014; Sutherland and Schneider, 2015; Yu et al., 2016; Dao et al., 2017); the above Nyström features attain 1.7x smaller Frobenius error and 17x smaller spectral error compared to the RFFs. This observation suggests the need for a more refined measure of kernel approximation error—one which better aligns with generalization performance, and can thus better guide the design of new approximation methods.

Building on recent theoretical work (Avron et al., 2017), we define a measure of approximation error which we find to be much more predictive of empirical generalization performance than the conventional metrics. In particular, we extend Avron et al.'s definition of Δ-*spectral approximation* to our definition of (Δ_1, Δ_2)-*spectral approximation* by decoupling the two roles played by Δ in the original definition.[1] This decoupling reveals that

*Equal contribution.

Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89.

[1]The original definition uses the same scalar Δ to upper and lower bound the approximate kernel matrix in terms of the exact kernel matrix in the semidefinite order.

Table 1: Memory utilization for kernel approximation methods. We consider data $x \in \mathbb{R}^d$, kernel features $z(x) \in \mathbb{R}^m$, mini-batch size s, # of classes c (for regression/binary classification $c = 1$). We assume full-precision numbers are 32 bits. We measure a method's memory utilization as the sum of the three components in this table.

Approximation Method	Feature generation	Feature mini-batch	Model parameters
Nyström	$32(md + m^2)$	$32ms$	$32mc$
RFFs	$32md$	$32ms$	$32mc$
Circulant RFFs	$32m$	$32ms$	$32mc$
Low-precision RFFs, b bits (ours)	$32m$	bms	$32mc$

Δ_1 and Δ_2 impact generalization differently, and can together much better explain the relative generalization performance of Nyström and RFFs than the original Δ, or the Frobenius or spectral errors. This (Δ_1, Δ_2) definition has an important consequence—in order for an approximate kernel matrix to be close to the exact kernel matrix, it is necessary for it to be *high rank*.

Motivated by the above connection between rank and generalization performance, we propose using *low-precision random Fourier features* (LP-RFFs) to attain a high-rank approximation under a memory budget. Specifically, we store each random Fourier feature in a low-precision fixed-point representation, thus achieving a higher-rank approximation with more features in the same amount of space. Theoretically, we show that when the quantization noise is much smaller than the regularization parameter, using low precision has negligible effect on the number of features required for the approximate kernel matrix to be a (Δ_1, Δ_2)-spectral approximation of the exact kernel matrix. Empirically, we demonstrate across four benchmark datasets (TIMIT, YearPred, CovType, Census) that in the mini-batch training setting, LP-RFFs can match the performance of full-precision RFFs (FP-RFFs) as well as the Nyström method, with 3x-10x and 50x-460x less memory, respectively. These results suggest that LP-RFFs could be an important tool going forward for scaling kernel methods to larger and more challenging tasks.

The rest of this paper is organized as follows: In Section 2 we compare the performance of the Nyström method and RFFs in terms of their training memory footprint. In Section 3 we present a more refined measure of kernel approximation error to explain the relative performance of Nyström and RFFs. We introduce the LP-RFF method and corresponding analysis in Section 4, and present LP-RFF experiments in Section 5. We review related work in Section 6, and conclude in Section 7.

2 NYSTRÖM VS. RFFS: AN EMPIRICAL COMPARISON

To inform our design of memory-efficient kernel approximation methods, we first perform an empirical study of the generalization performance vs. memory utilization of Nyström and RFFs. We begin by reviewing the memory utilization for these kernel approximation methods in the mini-batch training setting; this is a standard setting for training large-scale kernel approximation models (Huang et al., 2014; Yang et al., 2015; May et al., 2017), and it is the setting we will be using to evaluate the different approximation methods (Sections 2.2, 5.1). We then show that RFFs outperform Nyström given the same training memory budget, even though the opposite is true given a budget for the number of features (Yang et al., 2012). Lastly, we demonstrate that the Frobenius and spectral norms of the kernel approximation error matrix align poorly with generalization performance, suggesting the need for a more refined measure of approximation error for evaluating the quality of a kernel approximation method; we investigate this in Section 3.

For background on RFFs and the Nyström method, and for a summary of our notation, see Appendix A.

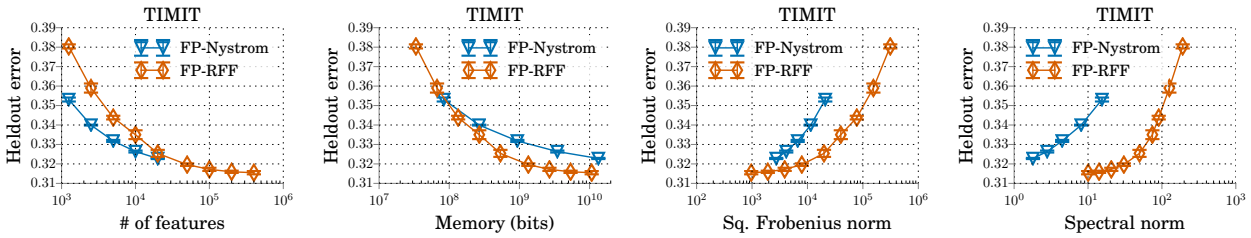
2.1 Memory Utilization

The optimization setting we consider is mini-batch training over kernel approximation features. To understand the training memory footprint, we present in Table 1 the memory utilization of the different parts of the training pipeline. The three components are:

1. *Feature generation*: Computing m RFFs over data in $\mathbb{R}^d$ requires a random projection matrix $W \in \mathbb{R}^{m \times d}$. The Nyström method stores m "landmark points" $\hat{x}_i \in \mathbb{R}^d$, and a projection matrix in $\mathbb{R}^{m \times m}$.
2. *Feature mini-batch*: Kernel approximation features $z(x_i) \in \mathbb{R}^m$ for all x_i in a mini-batch are stored.[2]
3. *Model parameters*: For binary classification and regression, the linear model learned on the $z(x)$ features is a vector $\theta \in \mathbb{R}^m$; for c-class classification, it is a matrix $\theta \in \mathbb{R}^{m \times c}$.

In this work we focus on reducing the memory occupied by the mini-batches of features, which can occupy a

[2]For simplicity, we ignore the memory occupied by the mini-batches of d-dim. inputs and c-dim. outputs, as generally the number of kernel approx. features $m \gg d, c$.



(a) (b) (c) (d)

Figure 1: Generalization performance of full-precision RFFs and Nyström with respect to the number of features and training memory footprint on TIMIT (a,b). Nyström performs better for a fixed number of features, while RFFs perform better under a memory budget. We also see that the generalization performance of these methods does not align well with the Frobenius or spectral norms of their respective kernel approximation error matrices (c,d). For results on YearPred, CovType, and Census, see Appendix D.2.

significant fraction of the training memory. Our work is thus orthogonal to existing work which has shown how to reduce the memory utilization of the feature generation (Le et al., 2013; Yu et al., 2015) and the model parameters (Sainath et al., 2013a; Sindhwani et al., 2015; De Sa et al., 2018) (e.g., using structured matrices or low precision). Throughout this paper, we measure the memory utilization of a kernel approximation method as the sum of the above three components.

2.2 Empirical Comparison

We now compare the generalization performance of RFFs and the Nyström method in terms of their training memory footprint. We demonstrate that RFFs can outperform the Nyström method given a memory budget, and show that the difference in performance between these methods cannot be explained by the Frobenius or spectral norms of their kernel approximation error matrices.

In experiments across four datasets (TIMIT, YearPred, CovType, Census (Garofolo et al., 1993; Dheeru and Karra Taniskidou, 2017)), we use up to 20k Nyström features and 400k RFFs to approximate the Gaussian kernel;[3] we train the models using mini-batch stochastic gradient descent with early stopping, with a mini-batch size of 250. We present results averaged from three random seeds, with error bars indicating standard deviations (for further experiment details, see Appendix D.2). In Figure 1(a) we observe that as a function of the number of kernel approximation features the Nyström method generally outperforms RFFs, though the gap narrows as m approaches 20k. However, we see in Figure 1(b) that RFFs attain better generalization performance as a function of memory. Interestingly, the relative performance of these methods cannot simply be explained by the Frobenius or spectral norms of the kernel approximation error matrices;[4] in Figure 1(c,d) we see that there are many cases in which the RFFs attain better generalization performance, in spite of having larger Frobenius or spectral approximation error. This is a phenomenon we observe on other datasets as well (Appendix D.2). This suggests the need for a more refined measure of the approximation error of a kernel approximation method, which we discuss in the following section.

3 A REFINED MEASURE OF KERNEL APPROX. ERROR

To explain the important differences in performance between Nyström and RFFs, we define a more refined measure of kernel approximation error—(Δ_1, Δ_2)-*spectral approximation*. Our definition is an extension of Avron et al.'s definition of Δ-spectral approximation, in which we decouple the two roles played by Δ in the original definition. This decoupling allows for a more fine-grained understanding of the factors influencing the generalization performance of kernel approximation methods, both theoretically and empirically. Theoretically, we present a generalization bound for kernel approximation methods in terms of (Δ_1, Δ_2) (Sec. 3.1), and show that Δ_1 and Δ_2 influence the bound in different ways (Prop. 1). Empirically, we show that Δ_1 and Δ_2 are more predictive of the Nyström vs. RFF performance than the Δ from the original definition, and the Frobenius and spectral norms of the kernel approximation error matrix (Sec. 3.2, Figure 2). An important consequence of the (Δ_1, Δ_2) definition is that attaining a small Δ_1 requires a large number of features; we leverage this insight to motivate our proposed method, low-precision random Fourier features, in Section 4.

[3] We consider different ranges for the number of Nyström vs. RFF features because the memory footprint for training with 400k RFFs is similar to 20k Nyström features.

[4] We consider the Frobenius and spectral norms of $K - \tilde{K}$, where K and $\tilde{K}$ are the exact and approximate kernel matrices for 20k randomly sampled heldout points.

3.1 (Δ_1, Δ_2)-spectral Approximation

We begin by reviewing what it means for a matrix A to be a Δ-spectral approximation of a matrix B (Avron et al., 2017). We then extend this definition to (Δ_1, Δ_2)-spectral approximation, and bound the generalization performance of kernel approximation methods in terms of Δ_1 and Δ_2 in the context of fixed design kernel ridge regression.

Definition 1. *For $\Delta \geq 0$, a symmetric matrix A is a* Δ-spectral approximation *of another symmetric matrix B if $(1-\Delta)B \preceq A \preceq (1+\Delta)B$.*

We extend this definition by allowing for different values of Δ in the left and right inequalities above:

Definition 2. *For $\Delta_1, \Delta_2 \geq 0$, a symmetric matrix A is a* (Δ_1, Δ_2)-spectral approximation *of another symmetric matrix B if $(1-\Delta_1)B \preceq A \preceq (1+\Delta_2)B$.*

Throughout the text, we will use Δ to denote the variable in Def. 1, and (Δ_1, Δ_2) to denote the variables in Def. 2. In our discussions and experiments, we always consider the smallest Δ, Δ_1, Δ_2 satisfying the above definitions; thus, $\Delta = \max(\Delta_1, \Delta_2)$.

In the paragraphs that follow we present generalization bounds for kernel approximation models in terms of Δ_1 and Δ_2 in the context of fixed design kernel ridge regression, and demonstrate that Δ_1 and Δ_2 influence generalization in different ways (Prop. 1). We consider the fixed design setting because its expected generalization error has a closed-form expression, which allows us to analyze generalization performance in a fine-grained fashion. For an overview of fixed design kernel ridge regression, see Appendix A.3.

In the fixed design setting, given a kernel matrix $K \in \mathbb{R}^{n \times n}$, a regularization parameter $\lambda \geq 0$, and a set of labeled points $\{(x_i, y_i)\}_{i=1}^n$ where the observed labels $y_i = \bar{y}_i + \epsilon_i$ are randomly perturbed versions of the true labels $\bar{y}_i \in \mathbb{R}$ (ϵ_i independent, $\mathbb{E}[\epsilon_i] = 0$, $\mathbb{E}\left[\epsilon_i^2\right] = \sigma^2 < \infty$), it is easy to show (Alaoui and Mahoney, 2015) that the optimal kernel regressor[5] f_K has expected error

$$\mathcal{R}(f_K) = \frac{\lambda^2}{n} \bar{y}^T (K + \lambda I)^{-2} \bar{y} + \frac{\sigma^2}{n} \operatorname{tr}\left(K^2 (K + \lambda I)^{-2}\right),$$

where $\bar{y} = (\bar{y}_1, \ldots, \bar{y}_n)$ is the vector of true labels.

This closed-form expression for generalization error allows us to bound the expected loss $\mathcal{R}(f_{\tilde{K}})$ of a kernel ridge regression model $f_{\tilde{K}}$ learned using an approximate kernel matrix $\tilde{K}$ in place of the exact kernel matrix K. In particular, if we define

$$\widehat{\mathcal{R}}(f_K) := \frac{\lambda}{n} \bar{y}^T (K + \lambda I)^{-1} \bar{y} + \frac{\sigma^2}{n} \operatorname{tr}\left(K (K + \lambda I)^{-1}\right),$$

which is an upper bound on $\mathcal{R}(f_K)$, we can bound the expected loss of $f_{\tilde{K}}$ as follows:

Proposition 1. *(Extended from (Avron et al., 2017)) Suppose $\tilde{K} + \lambda I$ is (Δ_1, Δ_2)-spectral approximation of $K + \lambda I$, for $\Delta_1 \in [0, 1)$ and $\Delta_2 \geq 0$. Let m denote the rank of $\tilde{K}$, and let f_K and $f_{\tilde{K}}$ be the kernel ridge regression estimators learned using these matrices, with regularizing constant $\lambda \geq 0$ and label noise variance $\sigma^2 < \infty$. Then*

$$\mathcal{R}(f_{\tilde{K}}) \leq \frac{1}{1-\Delta_1} \widehat{\mathcal{R}}(f_K) + \frac{\Delta_2}{1+\Delta_2} \frac{m}{n} \sigma^2. \quad (1)$$

We include a proof in Appendix B.1. This result shows that smaller values for Δ_1 and Δ_2 imply tighter bounds on the generalization performance of the model trained with $\tilde{K}$. We can see that as Δ_1 approaches 1 the bound diverges, and as Δ_2 approaches ∞ the bound plateaus. We leverage this generalization bound to understand the difference in performance between Nyström and RFFs (Sec. 3.2), and to motivate and analyze our proposed low-precision random Fourier features (Sec. 4).

Remark The generalization bound in Prop. 1 assumes the regressor f_K is computed via the closed-form solution for kernel ridge regression. However, in Sections 4-5 we focus on stochastic gradient descent (SGD) training for kernel approximation models. Because SGD can *also* find the model which minimizes the regularized empirical loss (Nemirovski et al., 2009), the generalization results carry over to our setting.

3.2 Revisiting Nyström vs. RFF Comparison

In this section we show that the values of Δ_1 and Δ_2 such that the approximate kernel matrix is a (Δ_1, Δ_2)-spectral approximation of the exact kernel matrix correlate better with generalization performance than the original Δ, and the Frobenius and spectral norms of the kernel approximation error; we measure correlation using Spearman's rank correlation coefficient ρ.

To study the correlation of these metrics with generalization performance, we train Nyström and RFF models for many feature dimensions on the Census regression task, and on a subsampled version of 20k train and heldout points from the CovType classification task. We choose these small datasets to be able to compute the various measures of kernel approximation error over the entire heldout set. We measure the spectral and Frobenius norms of $K - \tilde{K}$, and the Δ and (Δ_1, Δ_2) values between $K + \lambda I$ and $\tilde{K} + \lambda I$ (λ chosen via cross-validation), where K and $\tilde{K}$ are the exact and approximate kernel matrices for the heldout set. For more details about these experiments and how we compute Δ and (Δ_1, Δ_2), see Appendix D.3.

In Figure 2, we plot the generalization performance on these tasks as a function of these metrics; while the

[5] $f_K(x) = \sum_i \alpha_i k(x, x_i)$ for $\alpha = (K + \lambda I)^{-1} y$.

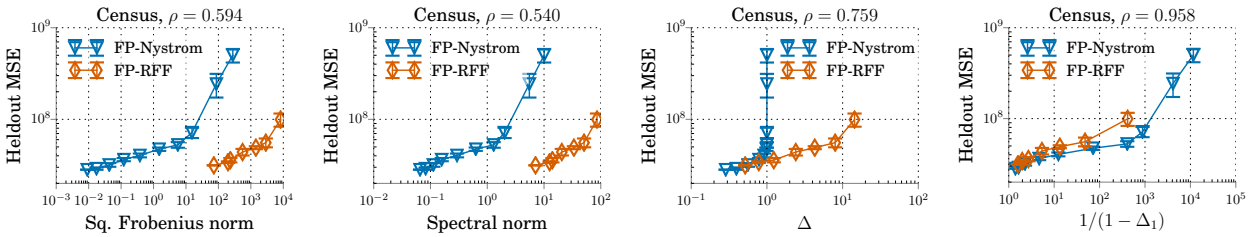


Figure 2: The correlation between generalization performance and different measures of kernel approximation error for the full-precision RFF and Nyström methods. We see that generalization performance aligns well with $1/(1-\Delta_1)$ (Spearman rank correlation coefficient $\rho = 0.958$), while aligning poorly with Δ and the spectral and squared Frobenius norms of the kernel approximation error matrix. See Appendix D.3 for results on CovType.

original Δ and the Frobenius and spectral norms generally do not align well with generalization performance, we see that $\frac{1}{1-\Delta_1}$ does. Specifically, $\frac{1}{1-\Delta_1}$ attains a Spearman rank correlation coefficient of $\rho = 0.958$, while squared Frobenius norm, spectral norm, and the original Δ attain values of 0.594, 0.540, and 0.759.[6] In Appendix D.3 we show these trends are robust to different kernel approximation methods and datasets. For example, we show that while other approximation methods (e.g., orthogonal RFFs (Yu et al., 2016)), like Nyström, can attain much lower Frobenius and spectral error than standard RFFs, this does not translate to improved Δ_1 or heldout performance. These results mirror the generalization bound in Proposition 1, which grows linearly with $\frac{1}{1-\Delta_1}$. For simplicity, we ignore the role of Δ_2 here, as Δ_1 appears to be sufficient for explaining the main differences in performance between these full-precision methods.[7] In Sections 4.2 and 5.2, however, we show that Δ_2 has a large influence on generalization performance for low-precision features.

Now that we have seen that Δ_1 has significant theoretical and empirical impact on generalization performance, it is natural to ask how to construct kernel approximation matrices that attain small Δ_1. An important consequence of the definition of Δ_1 is that for $\tilde{K}+\lambda I$ to have small Δ_1 relative to $K+\lambda I$, $\tilde{K}$ must be *high-rank*; in particular, a necessary condition is $\Delta_1 \geq \frac{\lambda_{m+1}(K)}{\lambda_{m+1}(K)+\lambda}$, where m is the rank of $\tilde{K}$ and $\lambda_i(K)$ is the i^{th} largest eigenvalue of K.[8] This sets a lower bound on the rank necessary for $\tilde{K}$ to attain small Δ_1 which holds regardless of the approximation method used, motivating us to design high-rank kernel approximation methods.

[6]One reason Δ_1 correlates better than Δ is because when $\Delta_2 > \Delta_1$, $\Delta = \max(\Delta_1, \Delta_2)$ hides the value of Δ_1. This shows why decoupling the two roles of Δ is important.

[7]While $1/(1-\Delta_1)$ aligns well with performance, it is not perfect—for a fixed Δ_1, Nyström generally performs slightly better than RFFs. In App. D.3.1 we suggest this is because Nyström has $\Delta_2 = 0$ while RFFs has larger Δ_2.

[8]By definition, $(K+\lambda I)(1-\Delta_1) \preceq \tilde{K}+\lambda I$. By Weyl's inequality this implies $\forall i\ (\lambda_i(K)+\lambda)(1-\Delta_1) \leq \lambda_i(\tilde{K})+\lambda$. If $\tilde{K}$ is rank m, then $\lambda_{m+1}(\tilde{K}) = 0$, and the result follows.

4 LOW-PRECISION RANDOM FOURIER FEATURES (LP-RFFS)

Taking inspiration from the above-mentioned connection between the rank of the kernel approximation matrix and generalization performance, we propose *low-precision random Fourier features* (LP-RFFs) to create a high-rank approximation matrix under a memory budget. In particular, we quantize each random Fourier feature to a low-precision fixed-point representation, thus allowing us to store more features in the same amount of space. Theoretically, we show that when the quantization noise is small relative to the regularization parameter, using low precision has minimal impact on the number of features required for the approximate kernel matrix to be a (Δ_1, Δ_2)-spectral approximation of the exact kernel matrix; by Proposition 1, this implies a bound on the generalization performance of the model trained on the low-precision features. At the end of this section (Section 4.3), we discuss a memory-efficient implementation for training a full-precision model on top of LP-RFFs.

4.1 Method Details

The core idea behind LP-RFFs is to use b bits to store each RFF, instead of 32 or 64 bits. We implement this with a simple stochastic rounding scheme. We use the parametrization $z_i(x) = \sqrt{2/m}\cos(w_i^T x + a_i) \in [-\sqrt{2/m}, \sqrt{2/m}]$ for the RFF vector $z(x) \in \mathbb{R}^m$ (Rahimi and Recht, 2007), and divide this interval into $2^b - 1$ sub-intervals of equal size $r = \frac{2\sqrt{2/m}}{2^b-1}$. We then randomly round each feature $z_i(x)$ to either the top or bottom of the sub-interval $[\underline{z}, \overline{z}]$ containing it, in such a way that the expected value is equal to $z_i(x)$; specifically, we round $z_i(x)$ to $\underline{z}$ with probability $\frac{\overline{z}-z}{\overline{z}-\underline{z}}$ and to $\overline{z}$ with probability $\frac{z-\underline{z}}{\overline{z}-\underline{z}}$. The variance of this stochastic rounding scheme is at most δ_b^2/m, where $\delta_b^2 := 2/(2^b-1)^2$ (Prop. 7 in App. C.2). For each low-precision feature $\tilde{z}_i(x)$ we only need to store the integer $j \in [0, 2^b-1]$ such that $\tilde{z}_i(x) = -\sqrt{2/m} + jr$, which takes b bits. Letting $\tilde{Z} \in \mathbb{R}^{n\times m}$ denote the matrix

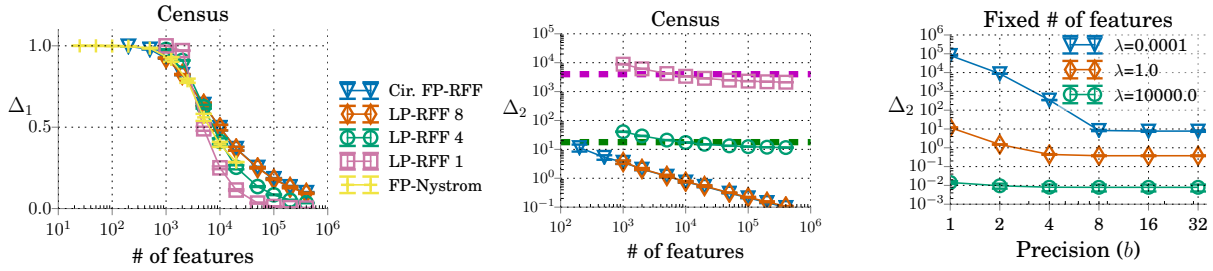


Figure 3: Empirical validation of Theorem 2. In the left and middle plots (shared legend), we see that as the # of features grows, LP-RFFs approach $\Delta_1 = 0$, but plateau at larger Δ_2 values (at most δ_b^2/λ, marked by dashed lines) for very low precisions. In the right plot we see that the larger λ is, the lower the precision at which using low precision does not impact Δ_2. For Δ_1 and Δ_2 vs. # features plots on CovType, see Appendix D.4.

of quantized features, we call $\tilde{K} = \tilde{Z}\tilde{Z}^T$ an *m-feature b-bit LP-RFF approximation* of a kernel matrix K.

As a way to further reduce the memory footprint during training, we leverage existing work on using circulant random matrices (Yu et al., 2015) for the RFF random projection matrix to only occupy $32m$ bits.[9] All our LP-RFF experiments use circulant projections.

4.2 Theoretical Results

In this section we show quantization has minimal impact on the number of features required to guarantee strong generalization performance in certain settings. We do this in the following theorem by lower bounding the probability that $\tilde{K} + \lambda I$ is a (Δ_1, Δ_2)-spectral approximation of $K + \lambda I$, for the LP-RFF approximation $\tilde{K}$ using m features and b bits per feature.[10]

Theorem 2. *Let $\tilde{K}$ be an m-feature b-bit LP-RFF approximation of a kernel matrix K, assume $\|K\| \geq \lambda \geq \delta_b^2 := 2/(2^b - 1)^2$, and define $a := 8 \operatorname{tr}\left((K + \lambda I_n)^{-1}(K + \delta_b^2 I_n)\right)$. Then for any $\Delta_1 \geq 0$, $\Delta_2 \geq \delta_b^2/\lambda$,*

$$\mathbb{P}\Big[(1-\Delta_1)(K+\lambda I) \preceq \tilde{K} + \lambda I \preceq (1+\Delta_2)(K+\lambda I)\Big] \geq$$

$$1 - a\left(\exp\left(\frac{-m\Delta_1^2}{\frac{4n}{\lambda}(1+\frac{2}{3}\Delta_1)}\right) + \exp\left(\frac{-m(\Delta_2 - \frac{\delta_b^2}{\lambda})^2}{\frac{4n}{\lambda}(1+\frac{2}{3}(\Delta_2 - \frac{\delta_b^2}{\lambda}))}\right)\right).$$

The proof of Theorem 2 is in Appendix C. To provide more intuition we present the following corollary:

Corollary 2.1. *Assuming $\Delta_1 \leq 3/2$, it follows that $(1-\Delta_1)(K+\lambda I_n) \preceq \tilde{K}+\lambda I_n$ with probability at least $1-\rho$ if $m \geq \frac{8n/\lambda}{\Delta_1^2} \log\left(\frac{a}{\rho}\right)$. Similarly, assuming $\Delta_2 \in \left[\frac{\delta_b^2}{\lambda}, \frac{3}{2}\right]$, it follows that $\tilde{K} + \lambda I_n \preceq (1+\Delta_2)(K+\lambda I_n)$ with probability at least $1-\rho$ if $m \geq \frac{8n/\lambda}{(\Delta_2 - \delta_b^2/\lambda)^2} \log\left(\frac{a}{\rho}\right)$.*

The above corollary suggests that using low precision has negligible effect on the number of features necessary to attain a certain value of Δ_1, and also has negligible effect for Δ_2 as long as $\delta_b^2/\lambda \ll \Delta_2$.

Validation of Theory We now empirically validate the following two predictions made by the above theory: (1) Using low precision has no effect on the asymptotic behavior of Δ_1 as the number of features m approaches infinity, while having a significant effect on Δ_2 when δ_b^2/λ is large. Specifically, as $m \to \infty$, Δ_1 converges to 0 for any precision b, while Δ_2 converges to a value upper bounded by δ_b^2/λ.[11] (2) If $\delta_b^2/\lambda \ll \Delta_2$, using b-bit precision will have negligible effect on the number of features required to attain this Δ_2. Thus, the larger λ is, the smaller the impact of using low precision should be on Δ_2.

To validate the first prediction, in Figure 3 (left, middle) we plot Δ_1 and Δ_2 as a function of the number of features m, for FP-RFFs and LP-RFFs; we use the same λ as in the Section 2 Census experiments. We show that for large m, all methods approach $\Delta_1 = 0$; in contrast, for precisions $b \leq 4$ the LP-RFFs converge to a Δ_2 value much larger than 0, and slightly less than δ_b^2/λ (marked by dashed lines).

To validate the second prediction, in Figure 3 (right) we plot Δ_2 vs. precision for various values of λ, using $m = 2000$ features for all precisions; we do this on a random subsample of 8000 Census training points. We see that for large enough precision b, the Δ_2 is very similar to the value from using 32-bit precision. Furthermore, the larger the value of λ, the smaller the precision b can be without significantly affecting Δ_2.

[9] Technically, m additional bits are needed to store a vector of Rademacher random variables in $\{-1, 1\}^m$.

[10] This theorem extends directly to the quantization of any kernel approximation feature matrix $Z \in \mathbb{R}^{n \times m}$ with i.i.d. columns and with entries in $[-\sqrt{2/m}, \sqrt{2/m}]$.

[11] By Lemma 3 in Appendix C, we know that $\mathbb{E}\left[\tilde{Z}\tilde{Z}^T\right] = K + D$ for a diagonal matrix D satisfying $0 \preceq D \preceq \delta_b^2 I_n$, where D is independent of m. As $m \to \infty$, Δ_2 converges to $\|(K + \lambda I)^{-1/2} D (K + \lambda I)^{-1/2}\| \leq \delta_b^2/\lambda$.

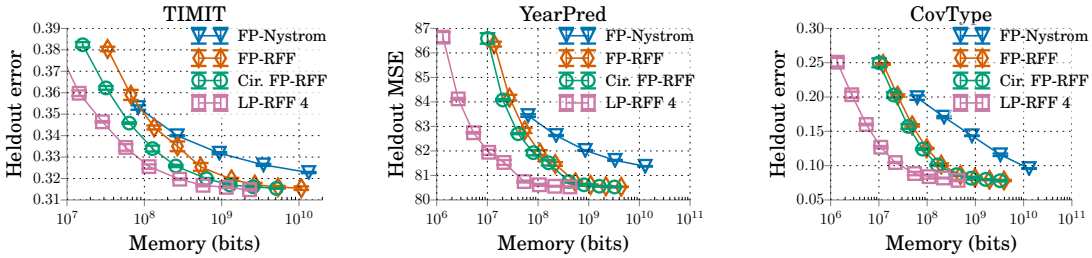


Figure 4: Generalization performance of FP-Nyström, FP-RFFs, circulant FP-RFFs, and LP-RFFs with respect to memory (sum of components in Table 1) on TIMIT, YearPred and CovType. LP-RFFs attain the best performance across a wide range of memory budgets. The same trend holds for Census in Appendix D.5.

Table 2: The compression ratios achieved by LP-RFFs relative to the best performing full-precision baselines.

	FP-RFFs	Cir. FP-RFFs	Nyström
Census	2.9x	15.6x	63.2x
YearPred	10.3x	7.6x	461.6x
Covtype	4.7x	3.9x	237.2x
TIMIT	5.1x	2.4x	50.9x

4.3 Implementation Considerations

In this paper, we focus on training full-precision models using mini-batch training over low-precision features. Here we describe how this mixed-precision optimization can be implemented in a memory-efficient manner.

Naively, to multiply the low-precision features with the full-precision model, one could first cast the features to full-precision, requiring significant intermediate memory. We can avoid this by *casting in the processor registers*. Specifically, to perform multiplication with the full-precision model, the features can be streamed to the processor registers in low precision, and then cast to full precision in the registers. In this way, only the features in the registers exist in full precision. A similar technique can be applied to avoid intermediate memory in the low-precision feature computation—after a full-precision feature is computed in the registers, it can be directly quantized in-place before it is written back to main memory. We leave a more thorough investigation of these systems issues for future work.

5 EXPERIMENTS

In this section, we empirically demonstrate the performance of LP-RFFs under a memory budget, and show that (Δ_1, Δ_2) are predictive of generalization performance. We show in Section 5.1 that LP-RFFs can attain the same performance as FP-RFFs and Nyström, while using 3x-10x and 50x-460x less memory. In Section 5.2, we show the strong alignment between (Δ_1, Δ_2) and generalization performance, once again validating the importance of this measure.

5.1 Empirical Evaluation of LP-RFFs

To empirically demonstrate the generalization performance of LP-RFFs, we compare their performance to FP-RFFs, circulant FP-RFFs, and Nyström features for various memory budgets. We use the same datasets and protocol as the large-scale Nyström vs. RFF comparisons in Section 2.2; the only significant additions here are that we also evaluate the performance of circulant FP-RFFs, and LP-RFFs for precisions $b \in \{1, 2, 4, 8, 16\}$. Across our experiments, we compute the total memory utilization as the sum of all the components in Table 1. We note that all our low-precision experiments are done in *simulation*, which means we store the quantized values as full-precision floating-point numbers. We report average results from three random seeds, with error bars showing standard deviations. For more details about our experiments, see Appendix D.5. We use the above protocol to validate the following claims on the performance of LP-RFFs.[12]

LP-RFFs can outperform full-precision features under memory budgets. In Figure 4, we plot the generalization performance for these experiments as a function of the total training memory for TIMIT, YearPred, and CovType. We observe that LP-RFFs attain better generalization performance than the full-precision baselines under various memory budgets. To see results for all precisions, as well as results on additional benchmark datasets (Census, Adult, CodRNA, CPU, Forest) from the UCI repository (Dheeru and Karra Taniskidou, 2017), see Appendix D.5.

LP-RFFs can match the performance of full-precision features with significantly less memory. In Table 2 we present the compression ratios we achieve with LP-RFFs relative to the best performing baseline methods. For each baseline (FP-RFFs, circulant FP-RFFs, Nyström), we find the smallest LP-RFF model, as well as the smallest baseline model, which attain within 10^{-4} relative performance of the best-performing baseline model; we then compute the ratio

[12] Our code: `github.com/HazyResearch/lp_rffs`.

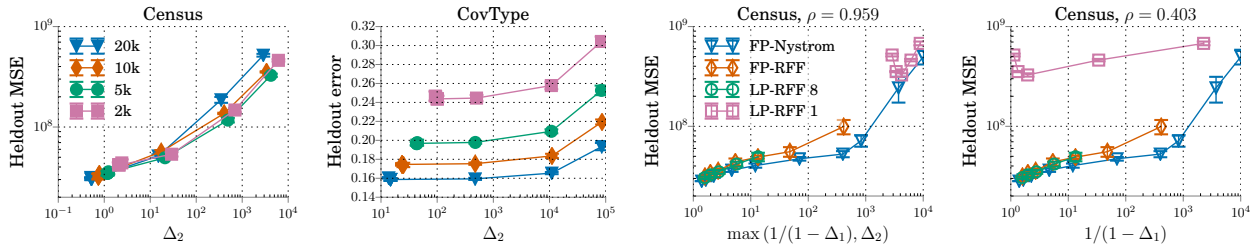


Figure 5: Generalization perf. vs. Δ_2 (left plots, shared legend), and vs. $1/(1-\Delta_1)$ and $\max(1/(1-\Delta_1), \Delta_2)$ (right plots, shared legend). Left: heldout performance deteriorates as Δ_2 gets larger due to lower precision. Right: $\max(1/(1-\Delta_1), \Delta_2)$ aligns well with performance across LP-RFF precisions (Spearman rank correlation coefficient $\rho = 0.959$), while $1/(1-\Delta_1)$ aligns poorly ($\rho = 0.403$). See Appendix D.6 for CovType results.

of the memory used by these two models (baseline/LP-RFF) for three random seeds, and report the average. We can see that LP-RFFs demonstrate significant memory saving over FP-RFFs, circulant FP-RFFs, and Nyström, attaining compression ratios of 2.9x-10.3x, 2.4x-15.6x, and 50.9x-461.6x, respectively.

5.2 Generalization Performance vs. (Δ_1, Δ_2)

In this section we show that Δ_1 and Δ_2 are together quite predictive of generalization performance across all the kernel approximation methods we have discussed. We first show that performance deteriorates for larger Δ_2 values as we vary the precision of the LP-RFFs, when keeping the number of features constant (thereby limiting the influence of Δ_1 on performance). We then combine this insight with our previous observation (Section 3.2) that performance scales with $\frac{1}{1-\Delta_1}$ in the full-precision setting by showing that across precisions the performance aligns well with $\max\left(\frac{1}{1-\Delta_1}, \Delta_2\right)$. For these experiments, we use the same protocol as for the (Δ_1, Δ_2) experiments in Section 3.2, but additionally consider LP-RFFs for precisions $b \in \{1, 2, 4, 8, 16\}$.

We show in Figure 5 (left plots) that for a fixed number of random Fourier features, performance deteriorates as Δ_2 grows. As we have shown in Figure 3 (left), Δ_1 is primarily governed by the rank of the approximation matrix, and thus holding the number of features constant serves as a proxy for holding Δ_1 roughly constant. This allows us to isolate the impact of Δ_2 on performance as we vary the precision.

To integrate the influence of Δ_1 and Δ_2 on generalization performance into a single scalar, we consider $\max\left(\frac{1}{1-\Delta_1}, \Delta_2\right)$. In Figure 5 (right plots) we show that when considering both low-precision and full-precision features, $\max\left(\frac{1}{1-\Delta_1}, \Delta_2\right)$ aligns well with performance ($\rho = 0.959$, incorporating *all* precisions), while $\frac{1}{1-\Delta_1}$ aligns poorly ($\rho = 0.403$).

In Appendix B we argue that performance scales roughly as Δ_2 instead of as $\Delta_2/(1+\Delta_2)$ (as suggested by Prop. 1) due to looseness in the Prop. 1 bound.

6 RELATED WORK

Low-Memory Kernel Approximation For RFFs, there has been work on using structured random projections (Le et al., 2013; Yu et al., 2015, 2016), and feature selection (Yen et al., 2014; May et al., 2016) to reduce memory utilization. Our work is orthogonal, as LP-RFFs can be used with both. For Nyström, there has been extensive work on improving the choice of landmark points, and reducing the memory footprint in other ways (Kumar et al., 2009; Hsieh et al., 2014; Si et al., 2014; Musco and Musco, 2017). In our work, we focus on the effect of *quantization* on generalization performance per bit, and note that RFFs are much more amenable to quantization. For our initial experiments quantizing Nyström features, see Appendix D.7.

Low Precision for Machine Learning There has been much recent interest in using low precision for accelerating training and inference of machine learning models, as well as for model compression (Gupta et al., 2015; De Sa et al., 2015; Hubara et al., 2016; De Sa et al., 2018, 2017; Han et al., 2016). There have been many advances in hardware support for low precision as well (Jouppi et al., 2017; Caulfield et al., 2017).

This work is inspired by the Nyström vs. RFF experiments in the PhD dissertation of May (2018), and provides a principled understanding of the prior results. For more related work discussion, see Appendix E.

7 CONCLUSION

We defined a new measure of kernel approximation error and demonstrated its close connection to the empirical and theoretical generalization performance of kernel approximation methods. Inspired by this measure, we proposed LP-RFFs and showed they can attain improved generalization performance under a memory budget in theory and in experiments. We believe these contributions provide fundamental insights into the generalization performance of kernel approximation methods, and hope to use these insights to scale kernel methods to larger and more challenging tasks.

Acknowledgements

We thank Michael Collins for his helpful guidance on the Nyström vs. RFF experiments in Avner May's PhD dissertation (May, 2018), which inspired this work. We also thank Jared Dunnmon, Albert Gu, Beliz Gunel, Charles Kuang, Megan Leszczynski, Alex Ratner, Nimit Sohoni, Paroma Varma, and Sen Wu for their helpful discussions and feedback on this project.

We gratefully acknowledge the support of DARPA under Nos. FA87501720095 (D3M) and FA86501827865 (SDH), NIH under No. N000141712266 (Mobilize), NSF under Nos. CCF1763315 (Beyond Sparsity) and CCF1563078 (Volume to Velocity), ONR under No. N000141712266 (Unifying Weak Supervision), the Moore Foundation, NXP, Xilinx, LETI-CEA, Intel, Google, NEC, Toshiba, TSMC, ARM, Hitachi, BASF, Accenture, Ericsson, Qualcomm, Analog Devices, the Okawa Foundation, and American Family Insurance, and members of the Stanford DAWN project: Intel, Microsoft, Teradata, Facebook, Google, Ant Financial, NEC, SAP, and VMWare. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views, policies, or endorsements, either expressed or implied, of DARPA, NIH, ONR, or the U.S. Government.

References

Ahmed El Alaoui and Michael W. Mahoney. Fast randomized kernel ridge regression with statistical guarantees. In *NIPS*, pages 775–783, 2015.

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 *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 253–262. PMLR, 2017.

Adrian M. Caulfield, Eric S. Chung, Andrew Putnam, Hari Angepat, Daniel Firestone, Jeremy Fowers, Michael Haselman, Stephen Heil, Matt Humphrey, Puneet Kaur, Joo-Young Kim, Daniel Lo, Todd Massengill, Kalin Ovtcharov, Michael Papamichael, Lisa Woods, Sitaram Lanka, Derek Chiou, and Doug Burger. Configurable clouds. *IEEE Micro*, 37(3): 52–61, 2017.

Jie Chen, Lingfei Wu, Kartik Audhkhasi, Brian Kingsbury, and Bhuvana Ramabhadrari. Efficient one-vs-one kernel ridge regression for speech recognition. In *ICASSP*, pages 2454–2458. IEEE, 2016.

Corinna Cortes, Mehryar Mohri, and Ameet Talwalkar. On the impact of kernel approximation on learning accuracy. In *AISTATS*, volume 9 of *JMLR Proceedings*, pages 113–120. JMLR.org, 2010.

Tri Dao, Christopher De Sa, and Christopher Ré. Gaussian quadrature for kernel features. In *NIPS*, pages 6109–6119, 2017.

Christopher De Sa, Ce Zhang, Kunle Olukotun, and Christopher Ré. Taming the wild: A unified analysis of Hogwild-style algorithms. In *NIPS*, pages 2674–2682, 2015.

Christopher De Sa, Matthew Feldman, Christopher Ré, and Kunle Olukotun. Understanding and optimizing asynchronous low-precision stochastic gradient descent. In *ISCA*, pages 561–574. ACM, 2017.

Christopher De Sa, Megan Leszczynski, Jian Zhang, Alana Marzoev, Christopher R. Aberger, Kunle Olukotun, and Christopher Ré. High-accuracy low-precision training. *arXiv preprint arXiv:1803.03383*, 2018.

Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017.

M. J. F. Gales. Maximum likelihood linear transformations for HMM-based speech recognition. *Computer Speech & Language*, 12(2):75–98, 1998.

J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren. DARPA TIMIT acoustic phonetic continuous speech corpus CDROM, 1993. URL `http://www.ldc.upenn.edu/Catalog/LDC93S1.html`.

Alex Gittens and Michael W. Mahoney. Revisiting the Nyström method for improved large-scale machine learning. *Journal of Machine Learning Research*, 17: 117:1–117:65, 2016.

Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. Deep learning with limited numerical precision. In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1737–1746, 2015.

Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and Huffman coding. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

Cho-Jui Hsieh, Si Si, and Inderjit S. Dhillon. Fast prediction for large-scale kernel machines. In *NIPS*, pages 3689–3697, 2014.

Po-Sen Huang, Haim Avron, Tara N. Sainath, Vikas Sindhwani, and Bhuvana Ramabhadran. Kernel methods match deep neural networks on TIMIT. In *ICASSP*, pages 205–209. IEEE, 2014.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *NIPS*, pages 4107–4115, 2016.

Norman P. Jouppi, Cliff Young, Nishant Patil, David A. Patterson, et al. In-datacenter performance analysis of a tensor processing unit. In *ISCA*, pages 1–12. ACM, 2017.

Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Ensemble Nyström method. In *NIPS*, pages 1060–1068, 2009.

Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Sampling methods for the Nyström method. *Journal of Machine Learning Research*, 13:981–1006, 2012.

Quoc V. Le, Tamás Sarlós, and Alexander J. Smola. Fastfood - computing Hilbert space expansions in log-linear time. In *ICML*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 244–252, 2013.

Zhu Li, Jean-Francois Ton, Dino Oglic, and Dino Sejdinovic. A unified analysis of random Fourier features. *arXiv preprint arXiv:1806.09178*, 2018.

Avner May. *Kernel Approximation Methods for Speech Recognition*. PhD thesis, Columbia University, 2018.

Avner May, Michael Collins, Daniel J. Hsu, and Brian Kingsbury. Compact kernel models for acoustic modeling via random feature selection. In *ICASSP*, pages 2424–2428. IEEE, 2016.

Avner May, Alireza Bagheri Garakani, Zhiyun Lu, Dong Guo, Kuan Liu, Aurélien Bellet, Linxi Fan, Michael Collins, Daniel J. Hsu, Brian Kingsbury, Michael Picheny, and Fei Sha. Kernel approximation methods for speech recognition. *arXiv preprint arXiv:1701.03577*, 2017.

N. Morgan and H. Bourlard. Generalization and parameter estimation in feedforward nets: Some experiments. In *NIPS*, 1990.

Cameron Musco and Christopher Musco. Recursive sampling for the Nyström method. In *NIPS*, pages 3836–3848, 2017.

Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19(4):1574–1609, 2009.

Tiberiu Popoviciu. Sur les équations algébriques ayant toutes leurs racines réelles. *Mathematica*, 9:129–145, 1935.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *NIPS*, pages 1177–1184, 2007.

Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *NIPS*, pages 1313–1320, 2008.

Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. In *NIPS*, pages 3218–3228, 2017.

Tara N. Sainath, Brian Kingsbury, Vikas Sindhwani, Ebru Arisoy, and Bhuvana Ramabhadran. Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In *ICASSP*, pages 6655–6659. IEEE, 2013a.

Tara N. Sainath, Brian Kingsbury, Hagen Soltau, and Bhuvana Ramabhadran. Optimization techniques to improve training speed of deep neural networks for large speech tasks. *IEEE Trans. Audio, Speech & Language Processing*, 21(11):2267–2276, 2013b.

Si Si, Cho-Jui Hsieh, and Inderjit S. Dhillon. Memory efficient kernel approximation. In *ICML*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 701–709, 2014.

Vikas Sindhwani, Tara N. Sainath, and Sanjiv Kumar. Structured transforms for small-footprint deep learning. In *NIPS*, pages 3088–3096, 2015.

Dougal J. Sutherland and Jeff G. Schneider. On the error of random Fourier features. In *UAI*, pages 862–871. AUAI Press, 2015.

Joel A. Tropp. An introduction to matrix concentration inequalities. *Foundations and Trends in Machine Learning*, 8(1-2):1–230, 2015.

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

Yuting Wei, Fanny Yang, and Martin J. Wainwright. Early stopping for kernel boosting algorithms: A general analysis with localized complexities. In *NIPS*, pages 6067–6077, 2017.

Christopher K. I. Williams and Matthias W. Seeger. Using the Nyström method to speed up kernel machines. In *NIPS*, pages 682–688. MIT Press, 2000.

Jiyan Yang, Vikas Sindhwani, Haim Avron, and Michael W. Mahoney. Quasi-Monte Carlo feature maps for shift-invariant kernels. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 485–493, 2014.

Tianbao Yang, Yu-Feng Li, Mehrdad Mahdavi, Rong Jin, and Zhi-Hua Zhou. Nyström method vs random Fourier features: A theoretical and empirical comparison. In *NIPS*, pages 485–493, 2012.

Zichao Yang, Marcin Moczulski, Misha Denil, Nando de Freitas, Alexander J. Smola, Le Song, and Ziyu Wang. Deep fried convnets. In *ICCV*, pages 1476–1483. IEEE Computer Society, 2015.

Ian En-Hsu Yen, Ting-Wei Lin, Shou-De Lin, Pradeep Ravikumar, and Inderjit S. Dhillon. Sparse random feature algorithm as coordinate descent in Hilbert space. In *NIPS*, pages 2456–2464, 2014.

Felix X. Yu, Sanjiv Kumar, Henry A. Rowley, and Shih-Fu Chang. Compact nonlinear maps and circulant extensions. *arXiv preprint arXiv:1503.03893*, 2015.

Felix X. Yu, Ananda Theertha Suresh, Krzysztof Marcin Choromanski, Daniel N. Holtmann-Rice, and Sanjiv Kumar. Orthogonal random features. In *NIPS*, pages 1975–1983, 2016.

Hantian Zhang, Jerry Li, Kaan Kara, Dan Alistarh, Ji Liu, and Ce Zhang. Zipml: Training linear models with end-to-end low precision, and a little bit of deep learning. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pages 4035–4043. PMLR, 2017.

Tong Zhang, Bin Yu, et al. Boosting with early stopping: Convergence and consistency. *The Annals of Statistics*, 33(4):1538–1579, 2005.