Learning Adaptive Random Features

Yanjun Li,[1] Kai Zhang,[*2] Jun Wang,[3] Sanjiv Kumar[4]
[1]University of Illinois at Urbana-Champaign, [2]Temple University,
[3]East China Normal University, [4]Google Research
[1]yli145@illinois.edu, [2]zhang.kai@temple.edu,
[3]jwang@sei.ecnu.edu.cn, [4]sanjivk@google.com

Abstract

Random Fourier features are a powerful framework to approximate shift invariant kernels with Monte Carlo integration, which has drawn considerable interest in scaling up kernel-based learning, dimensionality reduction, and information retrieval. In the literature, many sampling schemes have been proposed to improve the approximation performance. However, an interesting theoretic and algorithmic challenge still remains, i.e., *how to optimize the design of random Fourier features to achieve good kernel approximation on any input data using a low spectral sampling rate?* In this paper, we propose to compute more adaptive random Fourier features with optimized spectral samples ($\mathbf{w}_j$'s) and feature weights (p_j's). The learning scheme not only significantly reduces the spectral sampling rate needed for accurate kernel approximation, but also allows joint optimization with any supervised learning framework. We establish generalization bounds using Rademacher complexity, and demonstrate advantages over previous methods. Moreover, our experiments show that the empirical kernel approximation provides effective regularization for supervised learning.

Introduction

Despite the immense popularity of kernel-based learning algorithms (Schölkopf and Smola 2002), the expensive evaluation of nonlinear kernels has prohibited their application to large datasets. Low-rank kernel approximation is a powerful tool in alleviating the memory and computational cost of large kernel machines (Williams and Seeger 2001; Drineas and Mahoney 2005; Fowlkes et al. 2004; Halko, Martinsson, and Tropp 2011; Mahoney 2011; Kumar, Mohri, and Talwalkar 2012; Si, Hsieh, and Dhillon 2014). These methods adopt various sampling schemes on the rows/columns of the kernel matrix to obtain an efficient, low-rank decomposition, which in turn serves as a highly compact "empirical" kernel map that can reduce the cost of kernel machines from cubic to linear scale.

Random Fourier features, a highly innovative feature map pioneered by Rahimi and Recht (2008), has attracted significant interest, which will also be the focus of our work. Rather than decomposing the kernel matrix directly, the method resorts to the Fourier transform of positive semidefinite and shift-invariant kernels and obtains explicit feature maps using Monte Carlo approximation of the Fourier representation. The features can then be written as the cosine (or sine) of the inner product between the input data and random spectral samples drawn from a density specified by the characteristic function. For example, the characteristic function for Gaussian kernel is still Gaussian, hence the spectral samples should follow a Gaussian distribution.

In recent years, there has been continuing effort in designing optimal sampling schemes in computing Fourier features for accurate approximation. Le et al. (2013) proposed Fastfood, a feature map that is faster to compute thanks to a combination of diagonal Gaussian matrices and the Hadamard transform. Yang et al. (2014) showed that integral approximation using low-discrepancy quasi-Monte Carlo (QMC) sequence has a faster convergence than random Monte Carlo samples, and achieves lower error especially for high-dimensional data. Shen et al. (2017) proposed to apply moment matching on the spectral samples so that their empirical distribution is closer to the intended Gaussian. Besides shift-invariant kernels, approximate feature maps have also been considered for other nonlinear kernels, such as additive kernels (Vedaldi and Zisserman 2012), and polynomial kernels using spherical sampling (Pennington, Felix, and Kumar 2015). Despite these recent advances, open challenges still exist. First, most of the existing works ignore the impact of the input data distribution on the design of the feature map. Instead, the main dependency considered is the relation between the kernel and its characteristic function. Second, better kernel approximation with random Fourier features may not always translate to better generalization performance. Despite reported improvements in regression/classification, it has been found that such improvements do not correlate well with the improvements in the quality of kernel features (Avron et al. 2016; Chang et al. 2017).

In this paper, we argue that a good Fourier feature map should adapt to input data distribution. For example, the optimal sampling scheme in the spectral domain will ideally be different when approximating the kernel on data from different distributions, in order to achieve a desired accuracy with low sampling rate. To illustrate this, we use importance sampling (Barber 2012) as a motivating example in approximat-

[*]Contact author.

ing the kernel Fourier transform, and demonstrate that the optimal proposal density should be adaptive to the input distribution. Based on these observations, we propose a novel method to learn adaptive Fourier features, in which the spectral samples and their corresponding feature weights are optimized jointly through minimization of the empirical kernel approximation loss (EKAL). We also propose two approximators of the EKAL, so that an efficient iterative algorithm can be designed, reducing the overall time/space complexity to scale linearly with input sample size and dimension.

We want to emphasize some fundamental differences between our method and existing randomized algorithms. In those methods, a sampling probability is computed to select useful matrix columns often involving costly operations in the input space (e.g. SVD for leverage scores (Mahoney 2011) or computing norms of all matrix columns (Kumar, Mohri, and Talwalkar 2012). In comparison, we do not resort to any sampling strategy but instead explicitly optimize the spectral basis and weights in the *Fourier* space; besides, our approach is significantly cheaper and the cost of obtaining the basis can be as low as sub-linear. Recently (Bach 2017) proposed to select spectral samples by computing a discrete probability distribution over a large number of reference points. Note that their goal is functional approximation in the RKHS, while ours is kernel matrix approximation; besides, sampling in a high-dimensional space can be quite challenging, while optimizing the spectral basis based on explicit objective function can be computationally more tractable and more convenient.

We establish rigorous generalization bounds tailored to the minimizers of the EKAL approximators, using Rademacher average and McDiarmid's inequality. Unlike the loss function in a typical statistical learning problem, EKAL contains a small set of pairwise kernel evaluations that are not all independent. We overcome this challenge by creating independent *games* in statistically dependent *rounds* based on *round-robin tournament* scheduling. Our bounds can be easily translated to guarantees for supervised learning, using results that connect low-rank kernel approximation and learning accuracy (Cortes, Mohri, and Talwalkar 2010). These theoretical findings are complemented by numerical experiments, which show the clear advantage of our learned Fourier features over previous ones in kernel approximation. Indeed, our method is the first to fully exploit the input data to significantly improve the quality of Fourier features, bridging the gap between data-driven methods and fixed-basis methods. Besides effectively reducing the dimension of the kernel map (i.e., spectral sampling rate) to achieve desired accuracy, our method can also be incorporated in any supervised learning framework. In particular, we employ EKAL minimization as a regularization to building linear prediction models with learned Fourier features, leading to improved generalization performance.

Our main contributions include: (1) Theoretical justification of learning data-driven Fourier features using importance sampling as an example. (2) Joint optimization of spectral samples and feature weights in minimizing two types of EKAL approximators; generalization error bound for both. (3) Hybrid loss with both unsupervised kernel regularization and supervised prediction loss to improve the performance in classification/regression. (4) Extensive experimental results both in kernel approximation and in supervised learning tasks.

Random Fourier Features

Monte Carlo Method with Uniform Weights

Given a shift-invariant kernel function k, we wish to construct a feature map $\mathbf{Z}(\mathbf{x})$ whose pairwise inner-product approximates the kernel by $k(\mathbf{x}_1 - \mathbf{x}_2) \approx \langle \mathbf{Z}(\mathbf{x}_1), \mathbf{Z}(\mathbf{x}_2)\rangle$. By doing this, the kernel matrix $\mathbf{K}$ defined on $\{\mathbf{x}_\ell\}_{\ell=1}^N$ can then be approximated by a low-rank decomposition $\mathbf{Z}\mathbf{Z}^\top$, where $\mathbf{K}_{st} = k(\mathbf{x}_s - \mathbf{x}_t)$ and the s-th row of $\mathbf{Z}$ is $\mathbf{Z}(\mathbf{x}_s)$.

Rahimi and Recht (2008) pioneered the use of Fourier transform in solving this problem, by noting that any PSD shift invariant kernel k can be reconstructed using the Fourier basis sampled under the probability density defined by the characteristic function of k,

$$k(\mathbf{x}_1 - \mathbf{x}_2) = \int_{\mathbb{R}^d} e^{i\mathbf{w}^\top(\mathbf{x}_1 - \mathbf{x}_2)} p(\mathbf{w}) d\mathbf{w}. \qquad (1)$$

The density $p(\mathbf{w})$ is the Gaussian PDF $N(\mathbf{0}, \sigma^{-\mathbf{2}}\mathbf{I})$ for Gaussian kernel $k(\mathbf{x}_1 - \mathbf{x}_2) = \exp(-\|\mathbf{x}_1 - \mathbf{x}_2\|^2/(2\sigma^2))$. Suppose the feature map is defined as

$$\mathbf{Z}(\mathbf{x}) := \frac{1}{\sqrt{r}} \left[e^{i\mathbf{w}_1^\top \mathbf{x}}, e^{i\mathbf{w}_2^\top \mathbf{x}}, ..., e^{i\mathbf{w}_r^\top \mathbf{x}} \right], \qquad (2)$$

where $\{\mathbf{w}_j\}_{j=1}^r$ are d-dimensional spectral samples drawn independently from $p(\mathbf{w})$. It can then be observed that $\langle \mathbf{Z}(\mathbf{x}_1), \mathbf{Z}(\mathbf{x}_2)\rangle = \frac{1}{r}\sum_{j=1}^r e^{i\mathbf{w}_j^\top(\mathbf{x}_1 - \mathbf{x}_2)}$, which is an unbiased estimator of the Fourier representation of the kernel[1], and such a Monte Carlo approximation will asymptotically converge to the true integral (1) (Rahimi and Recht 2008).

Importance Sampling with Non-Uniform Weights

The sampling probability $p(\mathbf{w})$ in Monte Carlo method is the characteristic function of the kernel. For a given kernel, $p(\mathbf{w})$ is fixed regardless of the distribution of the input samples. However, given the input samples $\mathbf{x}_i$'s, we believe that its distribution $P(\mathbf{x})$ should also have an impact on the sampling probability $p(\mathbf{w})$. In particular, the optimal $p(\mathbf{w})$ should be adaptive to $P(\mathbf{x})$ in terms of accurately approximating kernel matrix defined on $P(\mathbf{x})$, using as few spectral samples ($\mathbf{w}_j$'s) as possible.

In order to see this, we consider the use of importance sampling, which is widely used in Bayesian inference to reduce the variance of approximations. Since k is real, one can rewrite the real part of Fourier representation as

$$k(\mathbf{x}_1 - \mathbf{x}_2) = \int_{\mathbb{R}^d} \cos(\mathbf{w}^\top(\mathbf{x}_1 - \mathbf{x}_2)) \frac{p(\mathbf{w})}{q(\mathbf{w})} q(\mathbf{w}) d\mathbf{w}.$$

[1] Since the kernel is real, one can remove the imaginary parts of (1) and (2). The real Fourier representation and features are $k(\mathbf{x}_1 - \mathbf{x}_2) = \int_{\mathbb{R}^d} \cos\left(\mathbf{w}^\top(\mathbf{x}_1 - \mathbf{x}_2)\right) p(\mathbf{w}) d\mathbf{w}$ and $\mathbf{Z}(\mathbf{x}) := \frac{1}{\sqrt{r}}\left[\cos(\mathbf{w}_1^\top \mathbf{x}), \dots, \cos(\mathbf{w}_r^\top \mathbf{x}), \sin(\mathbf{w}_1^\top \mathbf{x}), \dots, \sin(\mathbf{w}_r^\top \mathbf{x})\right]$.

Here, $q(\mathbf{w})$ is an importance-weighted proposal density. If $\mathbf{w}_j$'s are drawn from the distribution $q(\mathbf{w})$, then we can approximate the above integral with $\sum_{j=1}^{r} p_j \cos(\mathbf{w}_j^\top(\mathbf{x}_1 - \mathbf{x}_2))$, where $p_j = p(\mathbf{w}_j)/(r \cdot q(\mathbf{w}_j))$ satisfies $\mathbb{E}[p_j] = 1/r$. The above finite sample estimate is the inner product $\langle \widetilde{\mathbf{Z}}(\mathbf{x}_1), \widetilde{\mathbf{Z}}(\mathbf{x}_2)\rangle$ between weighted Fourier features:[2]

$$\widetilde{\mathbf{Z}}(\mathbf{x}) := \big[\sqrt{p_1}\cos(\mathbf{w}_1^\top \mathbf{x}), ..., \sqrt{p_r}\cos(\mathbf{w}_r^\top \mathbf{x}), \sqrt{p_1}\sin(\mathbf{w}_1^\top \mathbf{x}), ..., \sqrt{p_r}\sin(\mathbf{w}_r^\top \mathbf{x})\big]. \quad (3)$$

The optimal importance distribution minimizes the following expected error, which depends on input distribution:

$$\min_{q(\mathbf{w})} \mathbb{E}_{\mathbf{x}_1,\mathbf{x}_2}\mathbb{E}_{\mathbf{w}_j}\big|\langle \widetilde{\mathbf{Z}}(\mathbf{x}_1), \widetilde{\mathbf{Z}}(\mathbf{x}_2)\rangle - k(\mathbf{x}_1 - \mathbf{x}_2)\big|^2. \quad (4)$$

For example, in the case $\mathbf{x}_1 - \mathbf{x}_2$ is drawn from a Gaussian distribution, we can prove the following.

Claim 1. *If $k(\mathbf{x}_1 - \mathbf{x}_2) = \exp(-\|\mathbf{x}_1 - \mathbf{x}_2\|^2/(2\sigma^2))$, and $\mathbf{x}_1 - \mathbf{x}_2$ follows a Gaussian distribution $N(\mathbf{0}, \sigma_0^2\mathbf{I})$, then the optimal $q(\mathbf{w})$ satisfies*

$$q(\mathbf{w}) \propto \left(e^{-\sigma^2\|\mathbf{w}\|^2} + e^{-(\sigma^2+2\sigma_0^2)\|\mathbf{w}\|^2}\right)^{1/2}.$$

Claim 1 shows that the optimal $q(\mathbf{w})$ depends on the input distribution $P(\mathbf{x})$ (particularly on σ_0). It follows that, in the finite sample estimate, the choice of samples $\mathbf{w}_j$'s should also be data-dependent. Meanwhile, the corresponding feature weighting $p_j = p(\mathbf{w}_j)/(r \cdot q(\mathbf{w}_j))$ appears non-uniform and should depend on the input data distribution as well.

In practice, the underlying input distribution $P(\mathbf{x})$ is unknown, hence solving (4) for the optimal importance distribution $q(\mathbf{w})$ is intractable. Therefore, instead of designing the optimal $q(\mathbf{w})$, we propose to learn $\mathbf{w}_j$'s (spectral samples) and p_j's (corresponding feature weights) directly by minimizing the finite-sample kernel approximation error.

Adaptive Fourier Features

Empirical Kernel Approximation Loss

In this section, we explain how $\mathbf{w}_j$'s and p_j's in weighted Fourier features (3) can be learned efficiently from data. Suppose the input data $\mathbf{x}_\ell$ belong to a compact set $\mathcal{X}$. To simplify the notation, we define function $f : \mathcal{X} - \mathcal{X} \to \mathbb{R}$, parametrized by $\{\mathbf{w}_j, p_j\}_{j=1}^r$:

$$f(\mathbf{x}_1 - \mathbf{x}_2) := \sum_{j=1}^{r} p_j \cos\left(\mathbf{w}_j^\top(\mathbf{x}_1 - \mathbf{x}_2)\right).$$

The goal for kernel approximation is to learn the optimal $\mathbf{w}_j \in \mathcal{W}$ $(1 \le j \le r)$ and $\mathbf{p} \in \Delta^{r-1}$ that minimizes the mean squared error $\mathbb{E}_{\mathbf{x}_1,\mathbf{x}_2}[(k(\mathbf{x}_1 - \mathbf{x}_2) - f(\mathbf{x}_1 - \mathbf{x}_2))^2]$. If we restrict our attention to approximating the kernel matrix on a dataset $\{\mathbf{x}_\ell\}_{\ell=1}^N$, the loss function is the mean squared error over the empirical distribution over the dataset:

$$L(f) := \frac{1}{N^2}\sum_{s=1}^{N}\sum_{t=1}^{N}\left(f(\mathbf{x}_s - \mathbf{x}_t) - k(\mathbf{x}_s - \mathbf{x}_t)\right)^2.$$

In this objective, we intend to optimize both $\mathbf{w}_j$'s and their weights p_j's. Such an optimization will adapt the resulting Fourier features to the input data, hence making them more likely to obtain desired approximation accuracy with minimal sampling rate in the spectral domain.

Evaluating $k(\mathbf{x}_s - \mathbf{x}_t)$ over the whole sample is expensive and unnecessary. An important contribution of our work is to approximately evaluate the loss function $L(f)$ using a small number n of *landmark* data points ($n \ll N$), similar to the idea of sketching for regression (Avron, Sindhwani, and Woodruff 2013; Woodruff and others 2014). We use the following two strategies to choose landmark points and compute empirical kernel approximation loss (EKAL):

Random sampling: We randomly sample n points from the dataset with equal probability without replacement, i.e., choose $\{\mathbf{x}_{\ell_1}, \ldots, \mathbf{x}_{\ell_n}\} \subset \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, and compute

$$L^s(f) := \frac{1}{n^2}\sum_{s=1}^{n}\sum_{t=1}^{n}\big(f(\mathbf{x}_{\ell_s} - \mathbf{x}_{\ell_t}) - k(\mathbf{x}_{\ell_s} - \mathbf{x}_{\ell_t})\big)^2.$$

K-means clustering: We choose the landmark points as the cluster centers of the k-means algorithm. Suppose the s-th cluster has N_s members, whose centroid is $\mathbf{x}_s^c$ $(1 \le s \le n, \sum_{s=1}^n N_s = N)$. Thus EKAL with clustering is:

$$L^c(f) := \sum_{s=1}^{n}\sum_{t=1}^{n}\frac{N_s N_t}{N^2}\big(f(\mathbf{x}_s^c - \mathbf{x}_t^c) - k(\mathbf{x}_s^c - \mathbf{x}_t^c)\big)^2.$$

Iterative Algorithm

We write a unified objective function subsuming both the sampling-based loss function $L^s(f)$ and the clustering-based loss function $L^c(f)$, as follows

$$\min_{\mathbf{w}_j,\ \mathbf{p}\succeq 0}\sum_{s=1}^{n}\sum_{t=1}^{n} q_s^2 q_t^2\Big(\sum_j p_j\cos\big(\mathbf{w}_j^\top(\mathbf{x}_s - \mathbf{x}_t)\big) - k(\mathbf{x}_s - \mathbf{x}_t)\Big)^2 + \lambda\|\mathbf{p}\|^2,$$

where the squared landmark weights are $q_s^2 = 1/n$ for $L^s(f)$, and $q_s^2 = N_s/N$ for $L^c(f)$. Our goal is to optimize $\mathbf{w}_j$'s and p_j's by minimizing the EKAL with weight decay on $\mathbf{p} = [p_1, p_2, \ldots, p_r]^\top \succeq 0$.

Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]^\top \in \mathbb{R}^{n\times d}$ denote the landmark points, $\mathbf{W} = [\mathbf{w}_1\mathbf{w}_2, ..., \mathbf{w}_r] \in \mathbb{R}^{d\times r}$ the spectral samples, $\mathbf{K} \in \mathbb{R}^{n\times n}$ the kernel matrix on the landmark points, and $\mathbf{q} = [q_1, q_2, \ldots, q_n]^\top \in \mathbb{R}^n$ the landmark weights. We use $\mathbf{A} \odot \mathbf{B}$ and $\mathbf{A}^{\odot 2}$ to denote the entrywise product and the entrywise square, respectively. Then the above objective function can be equivalently written as follows:

$$\min_{\mathbf{W},\mathbf{p}\succeq 0}\Big\|\Big(\cos(\mathbf{X}\mathbf{W})\operatorname{diag}(\mathbf{p})\cos(\mathbf{X}\mathbf{W})^\top + \sin(\mathbf{X}\mathbf{W})\operatorname{diag}(\mathbf{p})\sin(\mathbf{X}\mathbf{W})^\top - \mathbf{K}\Big)\odot\mathbf{q}\mathbf{q}^\top\Big\|_{\mathrm{F}}^2 + \lambda\|\mathbf{p}\|^2. \quad (5)$$

The iterative algorithm is summarized in Algorithm 1.

[2] Uniformly weighted Fourier features $\mathbf{Z}(\mathbf{x})$ is a special case of $\widetilde{\mathbf{Z}}(\mathbf{x})$: $q(\mathbf{w}) = p(\mathbf{w})$ and hence $p_j = 1/r$.

Solving for feature weights $\mathbf{p}$: Define $\mathbf{C} \coloneqq \cos(\mathbf{XW})$ and $\mathbf{S} \coloneqq \sin(\mathbf{XW})$. We solve a quadratic programming (QP): $\min_{\mathbf{p}\succeq 0} \mathbf{p}^\top \mathbf{A}\mathbf{p} - 2\mathbf{b}^\top \mathbf{p}$, where

$$\begin{aligned}\mathbf{A} &= \left(\mathbf{C}^\top \operatorname{diag}^2(\mathbf{q})\,\mathbf{C}\right)^{\odot 2} + \left(\mathbf{C}^\top \operatorname{diag}^2(\mathbf{q})\,\mathbf{S}\right)^{\odot 2} \\ &\quad + \left(\mathbf{S}^\top \operatorname{diag}^2(\mathbf{q})\,\mathbf{C}\right)^{\odot 2} + \left(\mathbf{S}^\top \operatorname{diag}^2(\mathbf{q})\,\mathbf{S}\right)^{\odot 2} + \lambda \mathbf{I},\end{aligned}$$

$$\begin{aligned}\mathbf{b} &= \operatorname{diag}\left(\mathbf{C}^\top \operatorname{diag}^2(\mathbf{q})\,\mathbf{K}\,\operatorname{diag}^2(\mathbf{q})\,\mathbf{C}\right) \\ &\quad + \operatorname{diag}\left(\mathbf{S}^\top \operatorname{diag}^2(\mathbf{q})\,\mathbf{K}\,\operatorname{diag}^2(\mathbf{q})\,\mathbf{S}\right).\end{aligned}$$

This is a simple non-negative QP with an $2r \times 2r$ Hessian.

Solving for spectral samples $\mathbf{W}$: When $\mathbf{p}$ is fixed, one can solve for $\mathbf{W}$ via gradient descent. Define

$$\mathbf{R} \coloneqq \left(\mathbf{C}\operatorname{diag}(\mathbf{p})\mathbf{C}^\top + \mathbf{S}\operatorname{diag}(\mathbf{p})\mathbf{S}^\top - \mathbf{K}\right) \odot \mathbf{q}\mathbf{q}^\top.$$

Then the gradient of loss L with respect to $\mathbf{W}$ can be computed as follows:

$$\begin{aligned}\nabla_{\mathbf{C}} L &= 2(\mathbf{R}\odot \mathbf{q}\mathbf{q}^\top)\mathbf{C}\operatorname{diag}(\mathbf{p}), \\ \nabla_{\mathbf{S}} L &= 2(\mathbf{R}\odot \mathbf{q}\mathbf{q}^\top)\mathbf{S}\operatorname{diag}(\mathbf{p}), \qquad (6) \\ \nabla_{\mathbf{W}} L &= \mathbf{X}^\top\left(-\mathbf{S}\odot\nabla_{\mathbf{C}}L + \mathbf{C}\odot\nabla_{\mathbf{S}}L\right).\end{aligned}$$

Algorithm 1 Learning Fourier Features

```
**Input:** $\mathbf{X} \in \mathbb{R}^{n\times d}$
**Output:** $\mathbf{W}^{(T)} \in \mathbb{R}^{d\times r}$, $\mathbf{p}^{(T)} \in \mathbb{R}^r$
**Parameters:** learning rate $\mu$, number of iterations $T$, $S$
Initialize $\mathbf{W}^{(0)}$ and $\mathbf{p}^{(0)}$ using random Fourier features
**for** $t = 1, 2, \dots, T$ **do**
    $\mathbf{p}^{(t)} \leftarrow \arg\min_{\mathbf{p}\succeq 0} L(\mathbf{W}^{(t-1)}, \mathbf{p}) + \lambda\|\mathbf{p}\|^2$
                                        // Solve a QP for $\mathbf{p}$
    $\mathbf{W}^{(t)} \leftarrow \mathbf{W}^{(t-1)}$
    **for** $s = 1, 2, \dots, S$ **do**
        $\mathbf{W}^{(t)} \leftarrow \mathbf{W}^{(t)} - \mu \cdot \nabla_{\mathbf{W}} L(\mathbf{W}^{(t)}, \mathbf{p}^{(t)})$
    **end for**
        // Update $\mathbf{W}$ via gradient descent
**end for**
```

The time complexity for the QP is $O(nr^2+n^2r+r^3)$, and that for computing the gradient $\nabla_{\mathbf{W}}$ is $O(n^2r+dnr)$, which reduce to $O(r^3)$ and $O(r^3+dr^2)$ if $n = O(r)$. Overall, the time complexity is $O(TS(r^3+dr^3))$, with T, S being the number of iterations, and $r, n \ll N$.

Very recently, Chang et al. (2017) also proposed a data-driven approach to learn weights for random features. Their motivation is to sacrifice the unbiasedness of the estimator with the hope to lower the variance. In comparison, we build both theoretic and algorithmic linkage between Fourier features and data distribution, and our framework allows joint optimization of samples and weights.

Generalization Error Analysis

In this section, we establish generalization error bounds for minimizing two types of EKAL approximators, i.e., the sampling-based $L^s(f)$ and the clustering-based $L^c(f)$. These results guarantee that learned Fourier features can approximate the kernel not only over selected landmark points but also the entire data. Suppose spectral samples $\mathbf{w}_j$ belong to a compact sets $\mathcal{W}$. In addition, $\mathbf{p} = [p_1, p_2, \dots, p_r]^\top$ resides in the standard simplex $\Delta^{r-1} = \{\mathbf{p} : p_j \geq 0, \ \sum_{j=1}^r p_j = 1\}$ (since $\mathbb{E}[\sum_{j=1}^r p_j] = \mathbb{E}_{\mathbf{w}\sim q}[p(\mathbf{w})/q(\mathbf{w})] = 1$). Note that such simplex constraint can be easily relaxed to any compact constraint set. Define the set of all functions $\mathcal{F} = \{f : \mathbf{w}_j \in \mathcal{W}, \mathbf{p} \in \Delta^{r-1}\}$.

Our theoretical analysis of EKAL with sampling departs slightly from the last section. To create independence between landmark samples that simplifies our statistical argument, we sample n landmark points $\{\mathbf{x}_{\ell_s}\}_{s=1}^n$ from $\{\mathbf{x}_\ell\}_{\ell=1}^N$ *with replacement*, and minimize the empirical loss

$$L_n(f) \coloneqq \frac{2}{n(n-1)} \sum_{1\leq s<t\leq n} \left(f(\mathbf{x}_{\ell_s} - \mathbf{x}_{\ell_t}) - k(\mathbf{x}_{\ell_s} - \mathbf{x}_{\ell_t})\right)^2.$$

The additional loss incurred by learning on a small set of sampled landmarks is bounded. From Proposition 1, it appears that $n = O(dr)$ landmark points are required to achieve small generalization error, while empirical experiments show that $n = O(r)$ landmarks suffice (Figure 2).

Proposition 1. *With probability at least* $1-\delta$*,*

$$\begin{aligned}\sup_{f\in\mathcal{F}} |L_n(f) - L(f)| &\leq 4\sqrt{\frac{2\log(1/\delta)}{n}} \\ &\quad + 16\sqrt{\frac{(dr+r-1)(\log n + 2\log(3r_{\mathcal{W}}d_{\mathcal{X}}+9))}{n}},\end{aligned}$$

where $r_{\mathcal{W}} = \sup_{\mathbf{w}\in\mathcal{W}} \|\mathbf{w}\|$ *is the radius of* $\mathcal{W}$*, and* $d_{\mathcal{X}} = \sup_{\mathbf{x}_1,\mathbf{x}_2\in\mathcal{X}} \|\mathbf{x}_1 - \mathbf{x}_2\|$ *is the diameter of* $\mathcal{X}$*.*

Proof Sketch. To be more legible, the double subscripts in $\{\mathbf{x}_{\ell_s}\}_{s=1}^n$ are dropped in favor of $\{\mathbf{x}_\ell\}_{\ell=1}^n$. We define the collection of sampled landmarks $\mathbf{X}_n \coloneqq \{\mathbf{x}_\ell\}_{\ell=1}^n$, and $\Omega(\mathbf{X}_n) \coloneqq \sup_{f\in\mathcal{F}} |L_n(f) - L(f)|$. We first bound the expectation of $\Omega(\mathbf{X}_n)$ using a variation of the symmetrization trick (Gine and Zinn 1984). Unlike the empirical loss of a classic learning problem, the terms in $L_n(f)$ are not all independent. We think of the $n(n-1)/2$ terms as the "games" in a round-robin tournament (Lucas 1883). Without loss of generality, we assume that n is even. We break the right-hand side into $n-1$ (not independent but identically distributed) rounds with $n/2$ independent games in each round. Then by symmetry, $\mathbb{E}\,\Omega(\mathbf{X}_n) \leq \frac{4}{n} \cdot \mathbb{E}\sup_{f\in\mathcal{F}} \left|\sum_{j=1}^{n/2} z_j \left(f(\mathbf{x}_{2j-1} - \mathbf{x}_{2j}) - k(\mathbf{x}_{2j-1} - \mathbf{x}_{2j})\right)^2\right|$, where $\{z_j\}_{j=1}^{n/2}$ are i.i.d. Rademacher random variables.

Next, we bound the *Rademacher average* by constructing an ϵ-net of $\mathcal{F}$, such that for every $f \in \mathcal{F}$ there exists f' in the net that satisfies $\sup_{\mathbf{x}_1,\mathbf{x}_2\in\mathcal{X}} |f(\mathbf{x}_1 - \mathbf{x}_2) - f'(\mathbf{x}_1 - \mathbf{x}_2)| \leq 2\sqrt{1/n}$. By Massart's Lemma (Massart 2000),

$$\mathbb{E}\,\Omega(\mathbf{X}_n) \leq 8\sqrt{\frac{2\log(3r_{\mathcal{W}}d_{\mathcal{X}}\sqrt{n})^{dr}(9\sqrt{n})^{r-1}}{n}} + \frac{16}{\sqrt{n}}. \qquad (7)$$

By McDiarmid's inequality (McDiarmid 1989),

$$\Pr\left[\Omega(\mathbf{X}_n) \geq \mathbb{E}\,\Omega(\mathbf{X}_n) + t\right] \leq \exp\left(-\frac{nt^2}{32}\right). \qquad (8)$$

Proposition 1 follows from

$$\sup_{f\in\mathcal{F}} |L_n(f) - L(f)| \leq \mathbb{E}\,\Omega(\mathbf{X}_n) + \big(\Omega(\mathbf{X}_n) - \mathbb{E}\,\Omega(\mathbf{X}_n)\big),$$

where $\mathbb{E}\Omega(\mathbf{X}_n)$ and $\Omega(\mathbf{X}_n) - \mathbb{E}\Omega(\mathbf{X}_n)$ are bounded in (7) and (8), respectively. □

Next, we show that the additional loss incurred by the features learned from the cluster centers approaches zero when the quantization error of k-means diminishes. The kernels we consider are Lipschitz continuous. For example, the Lipschitz constant for $k(\mathbf{x}) = e^{-\|\mathbf{x}\|^2/(2\sigma^2)}$ is $L_k = e^{-1/2}/\sigma$.

Proposition 2. $\sup_{f\in\mathcal{F}} |L^c(f) - L(f)| \leq 8\rho \cdot (r_{\mathcal{W}} + L_k)$, *where* $\rho = \max_\ell \|\mathbf{x}_\ell - \mathbf{x}^c_{j(\ell)}\|$ *is the quantization error of* k*-means,* $r_{\mathcal{W}} = \sup_{\mathbf{w}\in\mathcal{W}} \|\mathbf{w}\|$ *is the radius of* $\mathcal{W}$*, and* $L_k = \sup_{\mathbf{x}_1,\mathbf{x}_2\in\mathcal{X}-\mathcal{X}} |k(\mathbf{x}_1) - k(\mathbf{x}_2)|/\|\mathbf{x}_1 - \mathbf{x}_2\|$ *is the Lipschitz constant of the kernel* k*.*

Proof Sketch. Since $|k(\cdot)| \leq 1$ and $|f(\cdot)| \leq 1$, we have

$$\begin{aligned}
&\sup_{f\in\mathcal{F}} |L^c(f) - L(f)| \\
&\leq 4\max_{s,t}\sup_{f\in\mathcal{F}}\Big|f(\mathbf{x}_s - \mathbf{x}_t) - f(\mathbf{x}^c_{j(s)} - \mathbf{x}^c_{j(t)})\Big| \\
&+ 4\max_{s,t}\Big|k(\mathbf{x}_s - \mathbf{x}_t) - k(\mathbf{x}^c_{j(s)} - \mathbf{x}^c_{j(t)})\Big|,
\end{aligned}$$

where $\mathbf{x}^c_{j(s)}$ is the centroid of the cluster containing $\mathbf{x}_s$. We then bound the two terms using the Lipschitz continuity of the feature map and the kernel, respectively. □

Using the relation between low-rank kernel approximation and learning accuracy (Cortes, Mohri, and Talwalkar 2010), one can easily translate the above results to stability bounds of supervised learning algorithms, in terms of $L_n(f)$ (or $L^c(f)$) and the choice of landmarks.

Target-Aware Fourier Features

In the literature, the Fourier features are mainly designed for numerically approximating the kernel matrix. However better kernel approximation may not always lead to better generalization (Avron et al. 2016). Therefore, it's desirable to improve the Fourier features with supervised information. To achieve this, we propose a hybrid loss, which is the combination of unsupervised loss (kernel approximation) and supervised loss (classification or regression error), as

$$\begin{aligned}
\min_{\substack{\mathbf{W},\mathbf{p}\succeq 0\\ \boldsymbol{\alpha},\beta}} \frac{1}{N}\sum_{j=1}^{N} c\big(g(\mathbf{x}_j;\mathbf{W},\boldsymbol{\alpha},\beta),\, y_j\big) + \gamma\|\boldsymbol{\alpha}\|^2 \\
+\eta\cdot\Big(L(\mathbf{W},\mathbf{p}) + \lambda\|\mathbf{p}\|^2\Big).
\end{aligned} \qquad (9)$$

Here, the first line is the prediction error on labeled samples $\{\mathbf{x}_j, y_j\}_{j=1}^N$, with weight decay on $\boldsymbol{\alpha} \in \mathbb{R}^{2r}$. The predictor $g(\mathbf{x}_j;\mathbf{W},\boldsymbol{\alpha},\beta)$ is chosen as a linear function over learned Fourier features (and thus corresponds to a nonlinear function in the input space):

$$g(\mathbf{x}_j;\mathbf{W},\boldsymbol{\alpha},\beta) = \boldsymbol{\alpha}^\top[\cos(\mathbf{W}^\top\mathbf{x}_j);\, \sin(\mathbf{W}^\top\mathbf{x}_j)] + \beta, \qquad (10)$$

$c(g(\mathbf{x}_i), y_i)$ can be $(g(\mathbf{x}_i)-y_i)^2$ for regression, or hinge loss $\max(0, 1 - y_i \cdot g(\mathbf{x}_i))$ for classification.

The second line $L(\mathbf{W},\mathbf{p}) + \lambda\|\mathbf{p}\|^2$ is the unsupervised regularization term, which is the finite-sample kernel approximation error as defined in (5). Here, the spectral samples $\mathbf{W}$ not only appear in the predictor g in (10), but also faithfully reconstruct the kernel matrix as specified in (5). Therefore, the "unsupervised" kernel approximation loss imposes highly informative regularization on computing the model g, which can notably improve the generalization performance as we shall discuss in more detail in the experiments section.

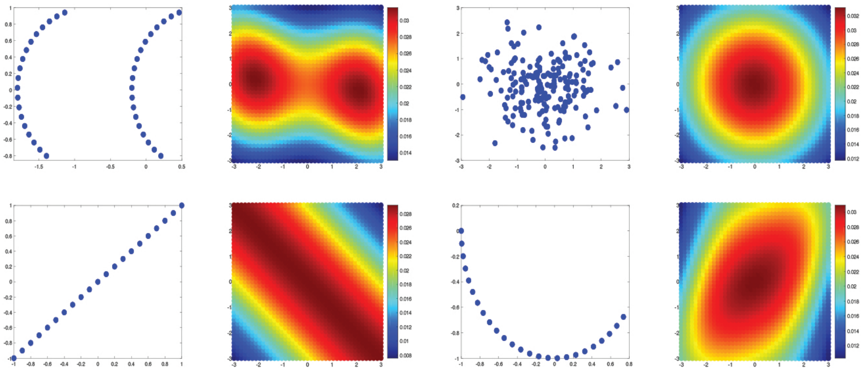
Algorithm 2 Learning Fourier Features with Supervision

```
Input: $\mathbf{X} \in \mathbb{R}^{n\times d}$
Output: $\mathbf{W}^{(T)} \in \mathbb{R}^{d\times r}$, $\mathbf{p}^{(T)} \in \mathbb{R}^r$, $\boldsymbol{\alpha}^{(T)}$, $\beta^{(T)}$
Parameters: learning rate $\mu$, number of iterations $T$, $S$
Initialize $\mathbf{W}^{(0)}$ and $\mathbf{p}^{(0)}$ using random Fourier features
for $t = 1, 2, \ldots, T$ do
    $\boldsymbol{\alpha}^{(t)}, \beta^{(t)} \leftarrow$ training linear machine
        // Update $\boldsymbol{\alpha}, \beta$ via gradient descent
    $\mathbf{p}^{(t)} \leftarrow \arg\min_{\mathbf{p}\succeq 0} L(\mathbf{W}^{(t-1)}, \mathbf{p}) + \lambda\|\mathbf{p}\|^2$
                                  // Solve a QP for $\mathbf{p}$
    $\mathbf{W}^{(t)} \leftarrow \mathbf{W}^{(t-1)}$
    for $s = 1, 2, \ldots, S$ do
        $\mathbf{W}^{(t)} \leftarrow \mathbf{W}^{(t)} - \mu \cdot \left(\frac{1}{N}\sum_{j=1}^{N} \nabla_{\mathbf{W}} c_j + \eta \cdot \nabla_{\mathbf{W}} L\right)$
    end for
        // Update $\mathbf{W}$ via gradient descent
end for
```

Table 1: Benchmark datasets

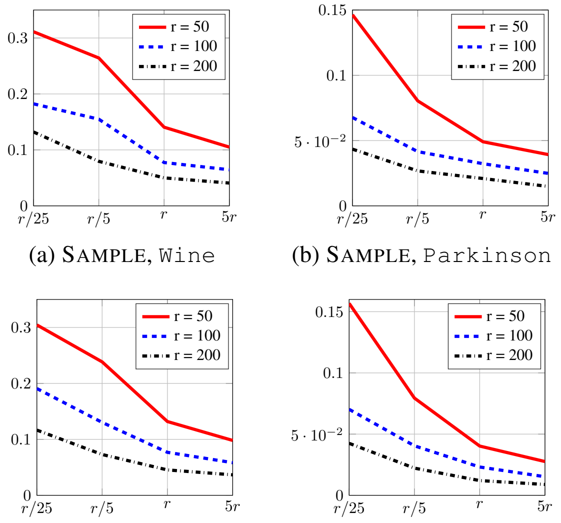
Dataset	Task	Input Dimension	Sample Size
Wine	Regression	11	4898
Parkinson	Regression	16	5875
CPU	Regression	21	8192
Adult	Classification	123	48842
Covtype	Classification	54	58101
MNIST	Classification	784	14780

Optimization procedures are in Algorithm 2. Each iteration of our optimization procedure has three steps. First, given $\mathbf{W}$, linear model g can be obtained using any off-the-shelf linear machines. Its parameters (weights $\boldsymbol{\alpha}$, bias β) do not need to be fully optimized in each iteration; empirically, a few steps of gradient descent suffice. Second, feature weights $\mathbf{p}$ are optimized by QP in (5). Third, spectral samples $\mathbf{W}$ are updated via gradient descent. The gradient of EKAL $\nabla_{\mathbf{W}} L$ is derived in (6); and that for the loss



Input data $\mathbf{x}_i$'s Weights p_j's Input data $\mathbf{x}_i$'s Weights p_j's

Figure 1: Some toy 2d samples $\mathbf{x}_i$'s (input space) and corresponding color-coded weights p_j's (spectral domain). The spectral samples $\mathbf{w}_j$'s are chosen as a grid for better visualization. Our approach generates data-dependent weighting.



(a) SAMPLE, `Wine` (b) SAMPLE, `Parkinson`

(c) CLUSTER, `Wine` (d) CLUSTER, `Parkinson`

Figure 2: Relative errors (y-axis) v.s. #landmarks n (x-axis) for varying dimension r; n expressed as multiples of r.

$c_j := c(g(\mathbf{x}_j), y_j)$ is

$$\nabla_{\mathbf{W}} c_j = \nabla_{g(\mathbf{x}_j)} c(g(\mathbf{x}_j), y_j) \cdot \mathbf{x}_j \cdot \big(-\sin(\mathbf{W}^\top \mathbf{x}_j) \odot \boldsymbol{\alpha}_1 + \cos(\mathbf{W}^\top \mathbf{x}_j) \odot \boldsymbol{\alpha}_2 \big)^\top,$$

where $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2 \in \mathbb{R}^r$ denote the first r entries (cosine part) and the second r entries (sine part) of $\boldsymbol{\alpha}$, respectively. Hence the gradient descent update for $\mathbf{W}$ is $\mathbf{W} \leftarrow \mathbf{W} - \mu \cdot \big(\frac{1}{N}\sum_{j=1}^{N} \nabla_{\mathbf{W}} c_j + \eta \cdot \nabla_{\mathbf{W}} L\big)$. Overall, the time complexity of each gradient descent step for $\boldsymbol{\alpha}$, β, and $\mathbf{W}$ is $O(Ndr)$, which is linear in the sample size.

Recently, Sinha and Duchi (2016) proposed to learn feature weights p_j's by kernel alignment; Yu et al. (2015) considered optimizing spectral samples $\mathbf{w}_j$'s using classification loss. There are two key differences between our work and theirs. First, previous works learn either p_j's or $\mathbf{w}_j$'s, while we adjust both. Second, we employ a hybrid loss incorporating EKAL minimization as a novel regularization to the learned model, leading to improved generalization.

Experiments

This section reports empirical evaluations in numerical kernel matrix approximation and supervised learning tasks. We denote our Fourier features learned by minimizing EKAL (Algorithm 1) with sampling and clustering as SAMPLE and CLUSTER, respectively. In supervised learning tasks, we minimize hybrid loss with sampling-based EKAL and name it SUPERVISE. We have compared with the following methods for constructing Fourier features:

- MC: Standard Monte Carlo sampling for random Fourier features (Rahimi and Recht 2008)
- FASTFOOD: Fast sampling using Hadamard matrices (Le, Sarlós, and Smola 2013).
- HALTON, SOBOL, LATTICE, DIGIT: 4 low-discrepancy sequences used in QMC sampling (Yang et al. 2014).
- MM: The moment matching approach (Shen, Yang, and Wang 2017).
- WEIGHT: Learning feature weights for kernel approximation via linear ridge regression (Chang et al. 2017).
- ALIGN: Kernel learning via alignment maximization (Sinha and Duchi 2016).

The benchmark datasets used are listed in Table 1. For simplicity, we convert `Covtype` and `MNIST` to binary classification (type 1 vs. not type 1, and digit 0 vs. digit 1). All data samples are split into training/test sets (2 : 1), unless provided in the original data. We tune the parameters via cross validation on training set. Input data is normalized to have zero mean and unit variance in each dimension, and the Gaussian kernel width $2\sigma^2$ is chosen as the dimension d of the input data, equal to $\mathbb{E}[\|\mathbf{x}_1 - \mathbf{x}_2\|^2/2]$ after normalization.

Two-Dimensional Toy Examples

In Figure 1, we plot some toy examples to demonstrate the intrinsic connection between Fourier features (in particular

Table 2: Relative kernel approximation errors. The errors of Lattice for MNIST are missing because the code provided by (Yang et al. 2014) cannot generate a lattice sequence of dimension larger than 250.

Dataset	r	MC	Fastfood	Halton	Sobol	Lattice	Digit	MM	Weight	Sample	Cluster
Wine	50	0.31	0.34	0.24	0.27	0.28	0.23	0.25	0.24	0.14	**0.13**
	100	0.19	0.25	0.18	0.20	0.19	0.15	0.17	0.15	**0.08**	**0.08**
	200	0.13	0.16	0.11	0.14	0.13	0.11	0.13	0.10	**0.05**	**0.05**
Parkinson	50	0.16	0.13	0.10	0.18	0.12	0.08	0.14	0.10	0.05	**0.04**
	100	0.08	0.14	0.09	0.12	0.07	0.05	0.09	0.05	0.03	**0.02**
	200	0.06	0.08	0.05	0.09	0.06	0.04	0.05	0.03	0.02	**0.01**
CPU	50	0.17	0.18	0.14	0.18	0.14	0.13	0.13	0.14	0.11	**0.09**
	100	0.13	0.13	0.11	0.12	0.11	0.09	0.09	0.10	0.07	**0.05**
	200	0.08	0.10	0.07	0.09	0.07	0.06	0.06	0.06	0.04	**0.03**
Adult	50	0.27	0.29	0.27	0.28	0.28	0.26	0.89	0.26	**0.23**	0.25
	100	0.19	0.22	0.18	0.20	0.20	0.18	0.24	0.19	**0.14**	0.16
	200	0.14	0.16	0.13	0.14	0.14	0.11	**0.10**	0.13	**0.10**	0.11
Covtype	50	0.23	0.25	0.22	0.22	0.23	0.21	0.19	0.22	0.16	**0.14**
	100	0.16	0.18	0.15	0.16	0.16	0.13	0.12	0.14	0.09	**0.07**
	200	0.11	0.12	0.10	0.12	0.11	0.09	0.09	0.09	0.05	**0.04**
MNIST	50	0.17	0.21	0.17	0.18	–	0.21	1.25	0.17	**0.14**	0.26
	100	0.13	0.15	0.13	0.13	–	0.13	1.10	0.12	**0.09**	0.24
	200	0.09	0.10	0.09	0.08	–	0.09	0.84	0.08	**0.06**	0.10

Table 3: Generalization error for regression (RMSE) and classification (%). Shaded methods use labels in computing features.

Dataset	MC	Fastfood	Halton	Sobol	Lattice	Digit	MM	Weight	Align	Sample	Cluster	Supervise
Wine	0.712	**0.697**	**0.697**	0.707	0.709	0.703	0.716	0.710	0.703	0.706	0.703	**0.697**
Parkinson	6.931	6.929	6.963	6.898	6.959	6.997	6.946	6.950	6.895	6.957	6.931	**6.432**
CPU	7.238	6.632	7.196	7.471	6.455	7.415	7.664	7.533	6.889	7.495	7.060	**3.687**
Adult	16.02	15.91	16.01	15.90	15.94	15.87	15.75	16.10	15.94	15.98	15.96	**15.64**
Covtype	19.37	19.94	19.39	19.63	19.39	19.95	19.78	19.67	19.46	19.52	**19.32**	19.78
MNIST	0.757	0.757	0.567	0.615	–	0.709	**0.378**	0.851	0.662	0.757	0.735	**0.378**

their weights p_j's) and input data distribution $P(\mathbf{x})$. For visualization purpose, $\mathbf{w}_j$'s are chosen from a uniform grid and only p_j's are optimized (in real-data experiments, $\mathbf{w}_j$'s and p_j's are optimized together). Note that the color-coded weight map can be deemed intuitively as an "distribution sensitive" importance distribution in the spectral domain.

Kernel Approximation Experiments

We use the relative error $\|\widetilde{\mathbf{K}} - \mathbf{K}\|_{\mathrm{F}}/\|\mathbf{K}\|_{\mathrm{F}}$ to quantify the performance of kernel approximation, where $\mathbf{K}$ is the exact kernel, and $\tilde{\mathbf{K}}$ the approximate one by Algorithm 1.

Number of Landmarks in EKAL-Approximators. We first explore the number of landmarks in the EKAL-approximators that guarantees accurate result on the entire data. For number of features r = 50, 100, and 200, we run Sample and Clusterp with number of landmarks $n = r/25$, $r/5$, r, and $5r$. Errors are shown in Figure 2. Clearly, one achieves relatively small generalization error for n as small as r. So we fix $n = r$ for the rest experiments.

Relative Approximation Errors. We reportp relative kernel approximation errors of all competing algorithms on six benchmark datasets for r = 50, 100, 200 (Table 2). In each dataset, at least one of our two methods, Sample and Cluster, achieves the lowest error. This demonstrates the advantage learned Fourier features in kernel approximation.

Usually, Cluster performs better than Sample, meaning that landmarks chosen as cluster centers yield better kernel approximation. However, when the number of landmarks n is lower than the input dimension ($n \leq 200 < 784 = d$ for MNIST), Cluster becomes worse than Sample. This is because when the number of clusters is too small, the cluster centers can be undesirably far from the actual input samples. In such a case, replacing the cluster centers with their closest samples proves an effective cure. Due to the space limit, we defer the empirical study of the performance of the modified EKAL approximator (that uses the closest samples to the cluster centers as landmarks) to an extended version of this paper. Indeed, such modification improves the performance of Cluster in high-dimensional cases, and achieves competitive performance against both Cluster and Sample for all datasets.

Supervised Learning Experiments

We perform supervised learning tasks of regression and classification; in both cases, a linear predictor is applied directly on learned Fourier features, leading to a nonlinear counterpart in the input space. For regression, we use ridge regression and report root mean square error (RMSE); for classification, we use ℓ_2-regularized SVM and report classification error. All the experiments use $n = r = 200$.

The regression or classification results on six benchmark datasets are reported in Table 3. When comparing methods that do not use label information in constructing the Fourier features (methods not shaded in Table 3), the results are inconclusive, i.e., any method performs better on certain tasks and worse on others. Overall, the unsupervised features learned by our methods (SAMPLE and CLUSTER) are quite comparable to others. When labels are used in constructing the Fourier features, our method (SUPERVISE) attains lowest generalization error on most datasets, demonstrating its superiority over both unsupervised feature construction methods and recently proposed, supervised method (ALIGN).

Conclusion

In this paper, we propose a novel framework for learning Fourier features that adapt to input data. Both spectral samples and weights are optimized jointly, which can be further engaged in any supervised learning framework. Extensive theoretical/empirical results demonstrate advantages of our method. In the future, we will study theoretic connections between the proposed Fourier features and explicit kernel low-rank decomposition.

Acknowledgement

Jun Wang is supported in part by the National Science Foundation of China NO.61672236.

References

Avron, H.; Sindhwani, V.; Yang, J.; and Mahoney, M. W. 2016. Quasi-monte carlo feature maps for shift-invariant kernels. *Journal of Machine Learning Research* 17(120):1–38.

Avron, H.; Sindhwani, V.; and Woodruff, D. 2013. Sketching structured matrices for faster nonlinear regression. In *Advances in Neural Information Processing Systems*, 2994–3002.

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

Barber, D. 2012. *Bayesian reasoning and machine learning*. Cambridge University Press.

Chang, W.-C.; Li, C.-L.; Yang, Y.; and Póczos, B. 2017. Data-driven random fourier features using stein effect. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, 1497–1503.

Cortes, C.; Mohri, M.; and Talwalkar, A. 2010. On the impact of kernel approximation on learning accuracy. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 113–120.

Drineas, P., and Mahoney, M. W. 2005. On the nyström method for approximating a gram matrix for improved kernel-based learning. *Journal of Machine Learning Research* 6(Dec):2153–2175.

Fowlkes, C.; Belongie, S.; Chung, F.; and Malik, J. 2004. Spectral grouping using the nystrom method. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26(2):214–225.

Gine, E., and Zinn, J. 1984. Some limit theorems for empirical processes. *The Annals of Probability* 12(4):929–989.

Halko, N.; Martinsson, P. G.; and Tropp, J. A. 2011. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review* 53(2):217–288.

Kumar, S.; Mohri, M.; and Talwalkar, A. 2012. Sampling methods for the nyström method. *Journal of Machine Learning Research* 13(Apr):981–1006.

Le, Q.; Sarlós, T.; and Smola, A. 2013. Fastfood-approximating kernel expansions in loglinear time. In *Proceedings of the International Conference on Machine Learning*, volume 85.

Lucas, É. 1883. Les jeux de demoiselles. In *Récréations Mathématiques*. Gauthier-Villars (Paris). 161–197.

Mahoney, M. W. 2011. Randomized algorithms for matrices and data. *Foundations and Trends® in Machine Learning* 3(2):123–224.

Massart, P. 2000. Some applications of concentration inequalities to statistics. *Annales de la Faculté des sciences de Toulouse : Mathématiques* 9(2):245–303.

McDiarmid, C. 1989. On the method of bounded differences. *Surveys in combinatorics* 141(1):148–188.

Pennington, J.; Felix, X. Y.; and Kumar, S. 2015. Spherical random features for polynomial kernels. In *Advances in Neural Information Processing Systems*, 1846–1854.

Rahimi, A., and Recht, B. 2008. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, 1177–1184.

Schölkopf, B., and Smola, A. J. 2002. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

Shen, W.; Yang, Z.; and Wang, J. 2017. Random features for shift-invariant kernels with moment matching. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*.

Si, S.; Hsieh, C.-J.; and Dhillon, I. 2014. Memory efficient kernel approximation. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, 701–709. PMLR.

Sinha, A., and Duchi, J. C. 2016. Learning kernels with random features. In *Advances in Neural Information Processing Systems*, 1298–1306.

Vedaldi, A., and Zisserman, A. 2012. Efficient additive kernels via explicit feature maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34(3):480–492.

Williams, C. K., and Seeger, M. 2001. Using the nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems*, 682–688.

Woodruff, D. P., et al. 2014. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science* 10(1–2):1–157.

Yang, J.; Sindhwani, V.; Avron, H.; and Mahoney, M. 2014. Quasi-monte carlo feature maps for shift-invariant kernels. In *Proceedings of the 31st International Conference on Machine Learning*, 485–493.

Yu, F. X.; Kumar, S.; Rowley, H.; and Chang, S.-F. 2015. Compact nonlinear maps and circulant extensions. *arXiv preprint arXiv:1503.03893*.