Training-Efficient Feature Map for Shift-Invariant Kernels

Xixian Chen[1,2], **Haiqin Yang**[1,2], **Irwin King**[1,2], **Michael R. Lyu**[1,2]
[1]Shenzhen Key Laboratory of Rich Media Big Data Analytics and Applications,
Shenzhen Research Institute, The Chinese University of Hong Kong, Shenzhen, China
[2]Department of Computer Science and Engineering,
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
{xxchen, hqyang, king, lyu}@cse.cuhk.edu.hk

Abstract

Random feature map is popularly used to scale up kernel methods. However, employing a large number of mapped features to ensure an accurate approximation will still make the training time consuming. In this paper, we aim to improve the training efficiency of shift-invariant kernels by using fewer informative features without sacrificing precision. We propose a novel feature map method by extending Random Kitchen Sinks through fast data-dependent subspace embedding to generate the desired features. More specifically, we describe two algorithms with different tradeoffs on the running speed and accuracy, and prove that $O(l)$ features induced by them are able to perform as accurately as $O(l^2)$ features by other feature map methods. In addition, several experiments are conducted on the real-world datasets demonstrating the superiority of our proposed algorithms.

1 Introduction

Kernel methods are powerful since they can achieve excellent performance when learning the non-linear relationship embedded in the training data. Usually, the non-linear relationship is encoded by a kernel function, $k : k(\mathbf{x},\mathbf{y}) = \langle\Phi(\mathbf{x}),\Phi(\mathbf{y})\rangle$, where $\Phi(\cdot) : \mathbb{R}^m \to \mathcal{H}$ maps the training data in the original m-dimensional feature space to a high-dimensional or even infinite-dimensional feature space, $\mathcal{H}$. Kernel methods can then train on the kernel matrix that represents the similarity of pairs of training data points without explicitly defining the mapping function $\Phi(\cdot)$. This obviously can overcome the curse of dimensionality.

Although kernel methods have attracted extensive investigation in the past two decades due to their significant performance advantage, yet one main issue is that they scale poorly with the size of the training dataset. Taking the kernel ridge regression (KRR) [Saunders *et al.*, 1998] as an example, given a data matrix $\mathbf{X} \in \mathbb{R}^{n\times m}$ consisting of n data points in m dimension, KRR incurs $O(n^3 + n^2m)$ time for training, much more than $O(nm^2)$ in the linear counterpart when $n \gg m$. Therefore, the huge time complexity makes kernel methods impractical to use in large datasets.

To resolve the scalability issue, random feature map is proposed to map the data explicitly to a low-dimensional Euclidean inner product space using the Fourier transform. Through the feature map $\mathbf{Z} : \mathbb{R}^m \to \mathbb{R}^l$, one can obtain $k(\mathbf{x},\mathbf{y}) = \langle\Phi(\mathbf{x}),\Phi(\mathbf{y})\rangle \approx \langle\mathbf{Z}(\mathbf{x}),\mathbf{Z}(\mathbf{y})\rangle$. Then instead of KRR, we can directly employ linear ridge regression with the mapped features, taking $O(nl^2+nml)$ time that depends linearly on n.

The number of mapped features l, however, exerts great influence on the effectiveness and efficiency of random feature map. Only using a small l can speed up the training, while this loses the accuracy in the kernel matrix approximation and the learning. In contrast, a large l can ensure the precision, but it takes more training time. Unfortunately, demonstrated in many literatures [Rahimi and Recht, 2009; Dai *et al.*, 2014], l should be chosen in the same order of n. In this regard, even for the linear ridge regression, it will take $O(n^3 + n^2m)$ training time, leading to a computation complexity almost as the same as that in the original kernel methods.

In this paper, we consider accelerating the training using an important kernel functions which are shift-invariant, namely $k(\mathbf{x},\mathbf{y}) = k(\mathbf{x} - \mathbf{y})$. This kind of kernels (e.g., Gaussian, Laplacian and Cauchy) is popularly and widely used. Our contributions in this work are summarized as follows:

- First, we propose a training-efficient feature map (TEFM) method by extending the Random Kitchen Sinks (RKS) [Rahimi and Recht, 2007; 2009] via fast data-dependent subspace embedding (FDSE). It can enjoy both the advantages of the RKS and FDSE, where RKS is a much more representative and efficient feature map for shift-invariant kernels. On the other hand, FDSE stands for a powerful dimensionality reduction tool on both the accuracy and speed.
- Second, we specify the proposed method TEFM by two algorithms: TEFM-G and TEFM-S. The former may be a bit more accurate, while the latter runs much faster.
- Third, we give provable results indicating that $O(l)$ features from TEFM-G or TEFM-S are able to perform almost as accurately as $O(l^2)$ features from other methods, which is also confirmed by extensive experiments.

The rest of paper is organized as follows. In Section 2, we summarize some related work. In Section 3, we present our method with the theoretical analysis. In Section 4, we provide empirical results. Finally, in Section 5, we conclude the whole work.

Notation. We use bold letters to denote either a *vector* or a *matrix*. The transpose of $\mathbf{X}$ is denoted by $\mathbf{X}^T$. $\mathbf{X}^\dagger$ denotes its Moore-Penrose inverse, Tr$(\mathbf{X})$ its trace and span$(\mathbf{X})$ its column spaces. $\|\mathbf{X}\|$ and $\|\mathbf{X}\|_F$ represent the spectral norm and Frobenius norm of $\mathbf{X}$, respectively. Finally, we use $\mathbf{X} \preccurlyeq \mathbf{Y}$ to represent that $\mathbf{Y} - \mathbf{X}$ is positive semidefinite.

2 Related Work

In this section, we review and discuss several existing feature map methods for shift-invariant kernels. Then, we outline the subspace embedding and analyze its properties.

2.1 Random Feature Maps for Shift-Invariant Kernels

Existing random feature maps for shift-invariant kernels include the RKS and its variants. We first spend some efforts on the RKS, a basic and effective method. The RKS is due to Bochner's theorem [Rudin, 1990], by which each entry of kernel matrix $\mathbf{K} \in \mathbb{R}^{n\times n}$ can be approximated as:

$$\begin{aligned}
\mathrm{K}_{ij} &= k(\mathbf{x}_i - \mathbf{x}_j) = \int p(\mathbf{z}) e^{i\mathbf{z}^T(\mathbf{x}_i - \mathbf{x}_j)} d\mathbf{z} \qquad (1)\\
&\approx \frac{2}{l}\sum_{s=1}^{l/2} \langle e^{i\mathbf{z}_s^T\mathbf{x}_i}, e^{i\mathbf{z}_s^T\mathbf{x}_j}\rangle\\
&= \sum_{s=1}^{l/2} \langle \frac{1}{\sqrt{l/2}}\cos(\mathbf{z}_s^T\mathbf{x}_i), \frac{1}{\sqrt{l/2}}\cos(\mathbf{z}_s^T\mathbf{x}_j)\rangle \ldots\\
&\quad \ldots + \langle \frac{1}{\sqrt{l/2}}\sin(\mathbf{z}_s^T\mathbf{x}_i), \frac{1}{\sqrt{l/2}}\sin(\mathbf{z}_s^T\mathbf{x}_j)\rangle\\
&= \langle \mathbf{Z}(\mathbf{x}_i) \in \mathbb{R}^l, \mathbf{Z}(\mathbf{x}_j) \in \mathbb{R}^l\rangle, \qquad (2)
\end{aligned}$$

or

$$\begin{aligned}
\text{Eq. (1)} &\approx \frac{1}{l}\sum_{s=1}^{l} \langle e^{i\mathbf{z}_s^T\mathbf{x}_i}, e^{i\mathbf{z}_s^T\mathbf{x}_j}\rangle\\
&\approx \sum_{s=1}^{l} \langle \sqrt{\frac{2}{l}}\cos(\mathbf{z}_s^T\mathbf{x}_i + b_s), \sqrt{\frac{2}{l}}\cos(\mathbf{z}_s^T\mathbf{x}_j + b_s)\rangle\\
&= \langle \mathbf{Z}(\mathbf{x}_i) \in \mathbb{R}^l, \mathbf{Z}(\mathbf{x}_j) \in \mathbb{R}^l\rangle, \qquad (3)
\end{aligned}$$

where $\mathbf{z}_s$ is sampled based on a proper probability density function $p(\mathbf{z})$ set to be the inverse Fourier transform of shift-invariant kernel function $k(\cdot)$ and b_s is uniformly sampled over $[0, 2\pi]$ [Rahimi and Recht, 2007].

Training kernel methods can then gain acceleration by using linear algorithms incorporated with $\{\mathbf{Z}(\mathbf{x}_i)\}_{i=1}^n$ and achieve higher accuracy than the linear algorithms directly operated on $\{\mathbf{x}_i\}_{i=1}^n$, simultaneously. Compared with kernel methods, linear algorithms using $\{\mathbf{x}_i\}_{i=1}^n$ run much more quickly especially in the low-dimensional data space, but gain less accuracy. However, this disadvantage can be alleviated by applying $\{\mathbf{Z}(\mathbf{x}_i)\}_{i=1}^n$ which contains the information of kernels.

The linear algorithms may be still impractical, since a large d has to be chosen to ensure a comparable accuracy as the kernel methods. The performance of kernel approximation using RKS is given by the proved assertion: $|\mathrm{K}_{ij} - \langle \mathbf{Z}(\mathbf{x}_i), \mathbf{Z}(\mathbf{x}_j)\rangle| \leqslant O(1/\sqrt{l})$ holds with a constant probability $\forall i, j \in [n]$ if drawing d features [Rahimi and Recht, 2007]. Therefore, RKS converges at the rate of $O(1/\sqrt{l})$ in the kernel approximation error. Accordingly, this kernel approximation error degenerates the learning accuracy by increasing the generation error from $O(1/\sqrt{n})$ in the kernel methods to $O(1/\sqrt{n} + 1/\sqrt{l})$ in the linear algorithms trained by RKS features. This suggests that l should be set on the order of n to guarantee the generalization performance [Rahimi and Recht, 2009; Dai *et al.*, 2014].

There are some follow-up works to improve RKS, but they still suffer from certain training inefficiency problems. [Le *et al.*, 2013] ideally reduces the feature generation time of RKS from $O(nml)$ to $O(nl\log m)$ by using fast Hadamard matrix, while the kernel approximation error is larger than RKS with the same number of features. Hence, to guarantee the same accuracy, more features have to be employed requiring more training time. In [Yang *et al.*, 2014], an advanced sampling technique Quasi-Monte Carlo is introduced, but it improves the convergence rate of kernel approximation error only when the feature number d is exponential in the data dimension m. This is not well satisfied in many real-world datasets and thus more features are needed in the training.

In a conclusion, RKS is still a better choice to reduce training time. Next, we review subspace embedding to be applied in our proposed method.

2.2 Subspace Embedding

Subspace embedding, informally, is able to reduce dimensionality by performing a map $f : \mathbb{R}^m \to \mathbb{R}^l$ on the dataset $\mathbf{X} = \{\mathbf{x}_i \in \mathbb{R}^m\}_{i=1}^n$, where $l < m$. Briefly, it can be cast into two categories: data-independent or data-dependent one.

In the data-independent subspace embedding such as random projection [Bingham and Mannila, 2001], f is predefined without knowing how the dataset is distributed in the original spaces, and it retains the data information (e.g., $\|x_i\|$, $\|\mathbf{X}\|$, $\|\mathbf{X}\|_F$, etc.) by ensuring an approximate isometry over the embedded subspaces with high probability. These whole embedding procedures can be quickly completed, but the dimensionality of the embedded subspaces cannot be sufficiently low, otherwise it will be impossible to approximate an isometry accurately [Nelson and Nguyn, 2014]. In contrast, the data-dependent subspace embedding maps the dataset into some important spaces determined by the data distribution. The insight is that, typically the dataset is distributed differently in each space, and most of the distribution lies in only a few spaces also regarded as important. Therefore, dimensionality can be substantially reduced without sacrificing much information. The issue is that getting the important spaces usually takes much time.

To effectively and efficiently reduce the dimensionality, [Halko *et al.*, 2011] proposes a fast and accurate method to find certain important space on the dataset, which is referred to as FDSE in our paper. This essentially belongs to the data-dependent category. The difference lies in that the important spaces are approximated by performing the data-independent subspace embedding on the data, by which the whole computation time can be reduced.

3 Training-Efficient Feature Map

In this section, first we propose our method together with two algorithms. Later, we provide error analysis of the proposed algorithms in the kernel approximation and related learning tasks. Finally, we compare and discuss the time complexity of each method on the regression task.

3.1 Methods and Algorithms

Here, we are at the stage to specify our method TEFM. [Yang *et al.*, 2012] has pointed out that features in RKS are randomly sampled from a distribution independent of the training data, leading to a data-independent vector representation whose most information will be restricted within some specific spaces. Thus, we propose to improve the RKS further via FDSE, i.e., the FDSE maps the RKS feature matrix $\mathbf{F} \in \mathbb{R}^{n\times d}$ into its l-dimensional important orthogonal row space fast obtained by combining random projection (e.g., Gaussian projection matrix) with QR decomposition, and thus yields the desired feature matrix $\mathbf{G} \in \mathbb{R}^{n\times l}$. This is expected to run faster than the standard QR decomposition with the time complexity reduced from $O(nd^2)$ to $O(ndl + dl^2)$, where we assume $n > d > l$.

Algorithm 1: TEFM-G

```
Input: data $\mathbf{X}^T = \{\mathbf{x}_i \in \mathbb{R}^m\}_{i=1}^n$, shift-invariant kernel function $k(\cdot)$, scalars [$q$, $d$, $l$].
Output: $\mathbf{G} \in \mathbb{R}^{n\times l}$.
 1: Form $\mathbf{F} \in \mathbb{R}^{n\times d}$ with each row vector constructed by method as shown in Eq. (2) or Eq. (3).
 2: Draw $\mathbf{\Theta} \in \mathbb{R}^{n\times l}$ from $\mathcal{N}(0, 1)$.
 3: Construct $\mathbf{Y} = (\mathbf{F}^T\mathbf{F})^q\mathbf{F}^T\mathbf{\Theta}$.
 4: Run QR decomposition on $\mathbf{Y}$ such that $\mathbf{QR} = \mathbf{Y}$ where $\mathbf{Q} \in \mathbb{R}^{d\times l}$ is column-orthogonal.
 5: $\mathbf{G} = \mathbf{FQ}$.
```

We summarize this method in Algorithm 1. As can be seen, there are two parts in this algorithm. Step 1 represents a feature matrix $\mathbf{F}$, where the inverse Fourier transform $p(\mathbf{z})$ of different $k(\cdot)$ can be found in [Rahimi and Recht, 2007]. Steps 2 to 5 show the FDSE. q is typically set by 0, 1 or 2, which is used to speed up the decay of the singular value of $\mathbf{F}$ and thus improves the accuracy of the whole algorithm [Halko *et al.*, 2011].

Moreover, step 4 in Algorithm 1 can still be accelerated by replacing $\mathbf{\Theta}$ with SRHT (Subsampled Randomized Hadamard Transform), then running QR decomposition only requires $O(nd\log l + dl^2)$ time. For the details of SRHT, please refer to [Tropp, 2011]. We summarize this method in Algorithm 2 that runs faster than Algorithm 1.

3.2 Kernel Approximation Analysis

In this part, we provide theoretical analysis to show that $O(l)$ features generated by our algorithms can approximate kernel matrix as accurately as $O(l^2)$ features induced by other existing methods like RKS and its extensions. Before proceeding, we first state two extracted and reformulated propositions.

Algorithm 2: TEFM-S

```
Input: data $\mathbf{X}^T = \{\mathbf{x}_i \in \mathbb{R}^m\}_{i=1}^n$, shift-invariant kernel function $k(\cdot)$, scalars [$d$, $l$].
Output: $\mathbf{G} \in \mathbb{R}^{n\times l}$.
 1: Run step 1 of Algorithm 1.
 2: Draw a SRHT matrix $\mathbf{\Theta} \in \mathbb{R}^{n\times l}$.
 3: Construct $\mathbf{Y} = \mathbf{F}^T\mathbf{\Theta}$.
 4: Run steps 4, 5 of Algorithm 1.
```

Proposition 1. *[Tropp, 2015] Suppose $\{\mathbf{S}_i\}_{i=1}^n$ are random square matrices with $\mathbb{E}\mathbf{S}_i = \mathbf{0}$ and $\|\mathbf{S}_i\| \leqslant L$. Define $\mathbf{Z} = \sum_{i=1}^n \mathbf{S}_i$, $\mathbf{\Xi}_1$ and $\mathbf{\Xi}_2$ satisfying $\mathbb{E}(\mathbf{ZZ}^T) \preccurlyeq \mathbf{\Xi}_1$ and $\mathbb{E}(\mathbf{Z}^T\mathbf{Z}) \preccurlyeq \mathbf{\Xi}_2$. Let $\xi = \max\{\|\mathbf{\Xi}_1\|, \|\mathbf{\Xi}_2\|\}$ and $r = \mathrm{Tr}(\mathbf{\Xi}_1 + \mathbf{\Xi}_2)/\xi$. Then, $\mathbb{P}\{\|\mathbf{Z}\| \geqslant t\} \leqslant 4r\exp(\frac{-t^2/2}{\xi+Lt/3})$ for $t \geqslant \sqrt{\xi} + L/3$.*

Proposition 2. *[Halko* et al.*, 2011] Denote the SVD of $\mathbf{A} \in \mathbb{R}^{m\times n}$ by $\mathbf{A} = \mathbf{Udiag}(\mathbf{\Sigma}_1, \mathbf{\Sigma}_2)(\mathbf{V}_1, \mathbf{V}_2)^T$ where $\mathbf{\Sigma}_1 \in \mathbb{R}^{k\times k}$, $\mathbf{\Sigma}_2 \in \mathbb{R}^{(n-k)\times(n-k)}$, $\mathbf{V}_1 \in \mathbb{R}^{n\times k}$, $\mathbf{V}_2 \in \mathbb{R}^{n\times(n-k)}$ and $\mathbf{diag}(\mathbf{\Sigma}_1, \mathbf{\Sigma}_2) \in \mathbb{R}^{n\times n}$ with $\mathbf{\Sigma}_1$ and $\mathbf{\Sigma}_2$ on its main diagonal. Given $\mathbf{\Omega} \in \mathbb{R}^{n\times l}$ and define $\mathbf{Y} = (\mathbf{AA}^T)^q\mathbf{A\Omega}$. If $\mathbf{V}_1^T\mathbf{\Omega}$ has full row rank, then $\|(\mathbf{I} - \mathbf{YY}^\dagger)\mathbf{A}\|^2 \leqslant \|(\mathbf{I} - \mathbf{YY}^\dagger)(\mathbf{A}^T\mathbf{A})^q\mathbf{A}\|^{2/(2q+1)} \leqslant (\|\mathbf{\Sigma}_2\|^{4q+2} + \|\mathbf{\Sigma}_2^{4q+2}(\mathbf{V}_2^T\mathbf{\Omega})(\mathbf{V}_1^T\mathbf{\Omega})^\dagger\|^2)^{1/(2q+1)}$.*

They are our main tools to derive the theoretical analysis. Then, we present our main results.

Theorem 1. *Suppose we have a kernel matrix $\mathbf{K} \in \mathbb{R}^{n\times n}$ constructed from shift-invariant functions and run Algorithm 1 to get features $\mathbf{G} \in \mathbb{R}^{n\times l}$. Then, with limited failure probability,*

$$\|\mathbf{K} - \mathbf{GG}^T\| \leqslant O(n/l). \tag{4}$$

Proof. Without loss of generality, we analyze the case in detail where $\mathbf{F}$ in step 1 corresponds to Eq. (2) and Gaussian kernel function $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma^2)$. The theorem and analysis can be applied in other cases. Revisit the variables and parameters in Algorithm 1, we have

$$\begin{aligned}
\|\mathbf{K} - \mathbf{GG}^T\| &= \|\mathbf{K} - \mathbf{FQQ}^T\mathbf{F}^T\| \\
&\leqslant \|\mathbf{K} - \mathbf{FF}^T\| + \|\mathbf{FF}^T - \mathbf{FQQ}^T\mathbf{F}^T\| \\
&= \|\mathbf{K} - \mathbf{FF}^T\| + \|\mathbf{F}(\mathbf{I} - \mathbf{QQ}^T)^2\mathbf{F}^T\| \qquad (5) \\
&= \|\mathbf{K} - \mathbf{FF}^T\| + \|(\mathbf{I} - \mathbf{QQ}^T)\mathbf{F}^T\|^2 \qquad (6) \\
&\leqslant O(n/\sqrt{d}) + O(n/l), \qquad (7)
\end{aligned}$$

where Eq. (5) follows from the fact $(\mathbf{I} - \mathbf{QQ}^T)$ is an orthogonal projection matrix and Eq. (7) will be proved as follows.

The first term in Eq. (7) is proved by Proposition 1. Define $a = d/2$ and $\mathbf{H}_i \in \mathbb{R}^{n\times 2}$ formed by columns $2i-1$ and $2i$ of $\mathbf{F}$, i.e., $\{\cos(\mathbf{z}_i^T\mathbf{x}_j)/\sqrt{a}, \sin(\mathbf{z}_i^T\mathbf{x}_j)/\sqrt{a}\}_{j=1}^n \in \mathbb{R}^{n\times 2}$, $\forall i \in [a]$. Let $\mathbf{S}_i = (\mathbf{K} - a\mathbf{H}_i\mathbf{H}_i{}^T)/a$. Then, $\mathbf{K} - \mathbf{FF}^T = \sum_{i=1}^a \mathbf{S}_i$ and $\mathbb{E}\mathbf{S}_i = \mathbf{0}$. $\mathbb{E}_{i\neq j}(\mathbf{S}_i^T\mathbf{S}_j) = \mathbf{0}$ also holds by that $\{\mathbf{H}_i\}_{i=1}^a$ are independent. Moreover, $\|\mathbf{K}\| \leqslant n$ due to $\mathbf{K}_{jj} \leqslant 1$, and

$\|\mathbf{H}_i\|^2 = \|\mathbf{H}_i\mathbf{H}_i^T\| \leqslant (\sqrt{1/a})^2 n = n/a$. Thus, $\|\mathbf{S}_i\| \leqslant 2n/a = L$. We accordingly define $\mathbf{\Xi}_1 = \mathbf{\Xi}_2 = \mathbb{E}(\mathbf{Z}^T\mathbf{Z}) = \mathbb{E}(\sum_{i=1}^a \mathbf{S}_i)^T(\sum_{i=1}^a \mathbf{S}_i) = a\mathbb{E}(\mathbf{S}_i^T\mathbf{S}_i)$ since $\mathbf{Z} = \sum_{i=1}^a \mathbf{S}_i$ is symmetric and $\mathbb{E}_{i\neq j}(\mathbf{S}_i^T\mathbf{S}_j) = \mathbf{0}$. Then $\xi = \|\mathbf{\Xi}_1\| = \|a\mathbb{E}(\mathbf{K} - a\mathbf{H}_i\mathbf{H}_i^T)^2/a^2\| = \|\{a^2\mathbb{E}(\mathbf{H}_i\mathbf{H}_i^T)^2 - \mathbf{K}^2\}/a\| = \|\{a^2\mathbb{E}(\mathbf{H}_i\mathbf{H}_i^T\mathbf{H}_i\mathbf{H}_i^T) - \mathbf{K}^2\}/a\| \leqslant \|\{a^2\mathbb{E}(\|\mathbf{H}_i\|^2\mathbf{H}_i\mathbf{H}_i^T) - \mathbf{K}^2\}/a\| \leqslant \|(n\mathbf{K} - \mathbf{K}^2)/a\| = \|\{\mathbf{U}(n\mathbf{\Sigma} - \mathbf{\Sigma}^2)\mathbf{U}^T\}/a\| = \|(n\mathbf{\Sigma} - \mathbf{\Sigma}^2)/a\| \leqslant n^2/(4a)$, where $\mathbf{U\Sigma U}^T$ is the SVD of $\mathbf{K}$ with the eigenvalues $\Sigma_{ii} = \sigma_i$ listed in the descending order. In the above inequalities, the first one holds by $0 \preccurlyeq \mathbf{\Xi}_1 = \{a^2\mathbb{E}(\mathbf{H}_i\mathbf{H}_i^T\mathbf{H}_i\mathbf{H}_i^T) - \mathbf{K}^2\}/a \preccurlyeq \{a^2\mathbb{E}(\|\mathbf{H}_i\|^2\mathbf{H}_i\mathbf{H}_i^T) - \mathbf{K}^2\}/a$ due to that $\mathbf{x}^T\mathbf{H}_i\mathbf{H}_i^T\mathbf{H}_i\mathbf{H}_i^T\mathbf{x} = \|\mathbf{x}^T\mathbf{H}_i\mathbf{H}_i^T\|^2 \leqslant \|\mathbf{x}^T\mathbf{H}_i\|^2\|\mathbf{H}_i^T\|^2 = \mathbf{x}^T\mathbf{H}_i\mathbf{H}_i^T\mathbf{x}\|\mathbf{H}_i^T\|^2$ for any $\mathbf{x} \in \mathbb{R}^n$, and the last one follows from that $\max_{0\leqslant\sigma_i\leqslant n} |(n\sigma_i - \sigma_i^2)/a| = n^2/(4a)$. $r = \mathrm{Tr}(\mathbf{\Xi}_1 + \mathbf{\Xi}_2)/\xi$ usually can satisfy that $1 \leqslant r \ll O(n)$. Now we can get the concentration bound that $\mathbb{P}(\|\mathbf{K} - \mathbf{FF}^T\| \geqslant t) \leqslant 4r\exp\{\frac{-t^2/2}{n^2/(4a)+2nt/(3a)}\}$. Let $\delta = 4r\exp\{\frac{-t^2/2}{n^2/(4a)+2nt/(3a)}\}$ and $\theta = 4r/\delta$. Then, $\|\mathbf{K} - \mathbf{FF}^T\| \leqslant \ln\theta\{2n/3a + \sqrt{4n^2/9a^2 + n^2/(2a\ln\theta)}\} = O(n/\sqrt{a}) = O(n/\sqrt{d})$ holds with failure probability at most δ, where we can regard $\ln\theta$ as a small value and also easily check that $t \geqslant \sqrt{\xi} + L/3$ holds.

To prove the second term in Eq. (7), we use Proposition 2 and get

$$\begin{aligned}\|(\mathbf{I}-\mathbf{QQ}^T)\mathbf{F}^T\|^2 &\leqslant \|(\mathbf{I} - \mathbf{YY}^\dagger)\mathbf{F}^T\|^2 \qquad (8)\\ &\leqslant (\|\mathbf{\Sigma}_2\|^{4q+2} + \|\mathbf{\Sigma}_2^{4q+2}(\mathbf{V}_2^T\mathbf{\Theta})(\mathbf{V}_1^T\mathbf{\Theta})^\dagger\|^2)^{1/(2q+1)}\\ &\leqslant (1 + \|(\mathbf{V}_2^T\mathbf{\Theta})(\mathbf{V}_1^T\mathbf{\Theta})^\dagger\|^2)^{1/(2q+1)}\|\mathbf{\Sigma}_2\|^2, \qquad (9)\end{aligned}$$

where $\mathbf{F} = \mathbf{U\Sigma V}^T = \mathbf{U}\mathbf{diag}(\mathbf{\Sigma}_1, \mathbf{\Sigma}_2)(\mathbf{V}_1, \mathbf{V}_2)^T$ with $\mathbf{\Sigma}_1 \in \mathbb{R}^{k\times k}$ and $\mathbf{\Sigma}_2 \in \mathbb{R}^{(d-k)\times(d-k)}$, and Eq. (8) holds because $\mathrm{span}(\mathbf{Y}) \subset \mathrm{span}(\mathbf{Q})$.

Notice that $\mathbf{\Theta}$ is a centered Gaussian matrix, thus $\mathbf{V}_1^T\mathbf{\Theta}$ has full row rank with probability one and Proposition 2 can be applied. The derivation for the upper bound of $\|(\mathbf{V}_2\mathbf{\Theta})(\mathbf{V}_1^T\mathbf{\Theta})^\dagger\|$ follows from [Halko *et al.*, 2011]. Then, combining and reformulating them, we get that $\|(\mathbf{I} - \mathbf{QQ}^T)\mathbf{F}^T\|^2 \leqslant [k,q,d,p]^{1/(2q+1)}n/(k+1) = O(n/l)$ holds with failure probability at most $6\exp(-p)$, where $l = k + p$ with p typically set to be a small constant (e.g.,5), $[k,q,d,p]$ means an expression on variables k,q,d,p, which turns smaller as p increases and is briefly represented here due to limited space. $[k,q,d,p]^{1/(2q+1)}$ can be considered as a small value if with a slightly big q.

By union bound and setting $d = \Omega(l^2)$, we prove Eq. (4) holds with failure probability at most $\delta + 6\exp(-p)$. □

The bound in Eq. (7) actually is incurred by RKS and FDSE. The first term of Eq. (7) denotes a spectral norm bound for RKS by Eq. (2) or Eq. (3), which is slightly tighter and more general than the expectation bound in [Lopez-paz *et al.*, 2014], and reflects the same convergence rate on RKS feature number as the element-wise bound [Rahimi and Recht, 2007]. If $d = \Omega(l^2)$, then training on $O(l)$ feature generated by our two algorithms will be much faster than $O(l^2)$ features by RKS with the approximation error almost on the same scale. In the experiments, we will see the accuracy versus the computation time for the training and feature map.

Remark. [Hamid *et al.*, 2014] tries to approximate the polynomial kernel more concisely and accurately, by which our approach is motivated similarly but has important differences. First, we use FDSE instead of random projection to approximate more accurately while maintaining the computation efficiency. Second, the theoretical result therein shows that the extent of approximation improvement relies on the degree of the polynomial kernel, while ours depends on the number of final generated features, which directly demonstrates that we can use *much fewer informative* features to train. Third, the experiment shows that our method can achieve improvement in the shift-invariant kernels, while [Hamid *et al.*, 2014] even *degenerates* the performance, which we conjecture is due to the different sampling procedures for the feature maps.

Theorem 2. *Suppose we have a kernel matrix* $\mathbf{K} \in \mathbb{R}^{n\times n}$ *constructed from shift-invariant functions and run Algorithm 2 to get features* $\mathbf{G} \in \mathbb{R}^{n\times l}$*. Then, with limited failure probability,*

$$\|\mathbf{K} - \mathbf{GG}^T\| \leqslant O(n/l). \qquad (10)$$

Proof. The proof is similar to that in Theorem 1 by setting $q = 0$ in Eq. (9). Then, with failure probability at most c/k, $\|(\mathbf{V}_2\mathbf{\Theta})(\mathbf{V}_1^T\mathbf{\Theta})^\dagger\|$ has a constant upper bound if $l = Ck\log k$, where c and C are some constants [Tropp, 2011]. Treating $\log k$ as a small value and using $\|\mathbf{\Sigma}_2\|^2 \leqslant n/(k+1)$ can complete the proof. □

Here, q has to be zero and c/k can be larger than $6\exp(-p)$ in Theorem 1. This implies Algorithm 1 may perform more accurately than Algorithm 2 although it runs slower.

3.3 Impact on Learning Tasks

In this part, we show how our features impact the learning accuracy on regression and classification tasks. The related algorithms including KRR and SVM can be unified into the following optimization problem [Yang *et al.*, 2011a; 2011b],

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n} \ell\{\hbar(\mathbf{w}^T\mathbf{Z}(\mathbf{x}_i), y_i)\}, \qquad (11)$$

where $\ell(t)$ means convex loss functions such as $\ell(t) = t^2/2$, $l(t) = \log\{1 + \exp(-t)\}$ and $l(t) = \max(0, 1-t)$. $t = \hbar(\mathbf{w}^T\mathbf{Z}(\mathbf{x}_i), y_i)$ denotes $\mathbf{w}^T\mathbf{Z}(\mathbf{x}_i) - y_i$ or $\mathbf{w}^T\mathbf{Z}(\mathbf{x}_i)y_i$. We also assume the magnitude of gradient $\ell'(t)$ can be upper bounded by C (nondifferentiable loss function like $l(t) = \max(0, 1-t)$ in SVM can be made differentiable by smoothing techniques [Nesterov, 2005; Zhang *et al.*, 2013]).

Theorem 3. *Suppose we get kernel matrix* $\mathbf{K} \in \mathbb{R}^{n\times n}$ *by operating shift-invariant functions on data matrix* $\mathbf{X}^T = \{\mathbf{x}_i \in \mathbb{R}^m\}_{i=1}^n$*, and feature matrix* $\mathbf{G}^T = \{\mathbf{G}_i \in \mathbb{R}^l\}_{i=1}^n$ *by Algorithm 1 or 2. Denote by* $L(\mathbf{w}^*)$ *the optimal value of Eq. (11). Then, let* $L(\mathbf{w}_\mathbf{G}^*)$ *be the obtained optimal value by training on* $\mathbf{Z}(\mathbf{x}_i) = \mathbf{G}_i$*, and* $L(\mathbf{w}_\mathbf{K}^*)$ *the optimal value by by training on* $\mathbf{K}$ *(i.e.,*$\mathbf{Z}(\mathbf{x}_i) = \Phi(\mathbf{x}_i)$*), where* $\mathbf{w}_\mathbf{G}^* \in \mathbb{R}^l$ *and* $\mathbf{w}_\mathbf{K}^*$ *may be infinite-dimensional. Then, with limited failure probability,*

$$L(\mathbf{w}_\mathbf{G}^*) \leqslant L(\mathbf{w}_\mathbf{K}^*) + O(1/l). \qquad (12)$$

Proof. Instead of directly applying Representer Theorem [Schölkopf *et al.*, 2001] on Eq. (11), a desired tight result can be obtained using its Lagrangian duality. Rewrite Eq. (11) into a constrained convex optimization problem by introducing new variables $\mathbf{g}^T = \{g_i = \hbar(\mathbf{w}^T\mathbf{Z}(\mathbf{x}_i), y_i)\}_{i=1}^n \in \mathbb{R}^{1\times n}$, and combine the Lagrangian with KKT conditions, then we get an equivalent dual problem without duality gap, i.e., $L(\mathbf{w}^*) = \max_{\boldsymbol{\alpha}} \sum_{i=1}^n \ell_*(\alpha_i) - \frac{1}{2\lambda}\boldsymbol{\alpha}^T\mathbf{Z}(\mathbf{X})\mathbf{Z}(\mathbf{X})^T\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{y}$, where $\boldsymbol{\alpha}^T = \{\alpha_i\}_{i=1}^n \in \mathbb{R}^{1\times n}$, $\mathbf{y}^T = \{y_i\}_{i=1}^n \in \mathbb{R}^{1\times n}$, each row of $\mathbf{Z}(\mathbf{X})$ is formed by $\mathbf{Z}(\mathbf{x}_i)$, $\hbar(\mathbf{w}^T\mathbf{Z}(\mathbf{x}_i), y_i) = \mathbf{w}^T\mathbf{Z}(\mathbf{x}_i) - y_i$ without loss of generality, and $\ell_*(\alpha_i) = \inf\{-\boldsymbol{\alpha}^T\mathbf{g} + \sum_{i=1}^n \ell(g_i)/n\}$ is supposed to be finite with $\alpha_i = \ell'(g_i)/n, \forall i \in [n]$. Then we have,

$$\begin{aligned} L(\mathbf{w}^*_{\mathbf{G}}) &= \max_{\boldsymbol{\alpha}} \sum_{i=1}^n \ell_*(\alpha_i) - \frac{1}{2\lambda}\boldsymbol{\alpha}^T\mathbf{G}\mathbf{G}^T\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{y} \\ &\leqslant L(\mathbf{w}^*_{\mathbf{K}}) + \max_{\boldsymbol{\alpha}} \frac{1}{2\lambda}\boldsymbol{\alpha}^T(\mathbf{K} - \mathbf{G}\mathbf{G}^T)\boldsymbol{\alpha} \\ &\leqslant L(\mathbf{w}^*_{\mathbf{K}}) + \frac{1}{2\lambda}\|\boldsymbol{\alpha}\|\|\mathbf{K} - \mathbf{G}\mathbf{G}^T\|\|\boldsymbol{\alpha}\| \\ &\leqslant L(\mathbf{w}^*_{\mathbf{K}}) + \frac{n(C/n)^2}{2\lambda}\|\mathbf{K} - \mathbf{G}\mathbf{G}^T\| \\ &\leqslant L(\mathbf{w}^*_{\mathbf{K}}) + \frac{C^2}{2n\lambda}O(\frac{n}{l}) = L(\mathbf{w}^*_{\mathbf{K}}) + O(\frac{1}{l}), \end{aligned}$$

where λ is assumed to be a certain fixed value and the last inequality holds with limited failure probability by using Theorem 1 or 2. □

We can extend the above result to other features with different kernel approximation performances, showing that a faster convergence of the kernel approximation error can lead to faster convergence in the minimized regularized training error. This can be used as a simplified measurement to quantify the impact of kernel approximation on the learning accuracy [Cortes *et al.*, 2010; Cesa-Bianchi *et al.*, 2014]. Therefore, applying the features that ensure a good kernel approximation favors an accurate learning on the learning tasks.

3.4 Computation Analysis

In this part, we compare the computation performance on ridge regression task written in the form of Eq. (11) using the square loss function. We use ridge regression due to that it can give a closed-form solutions for the mapped features and standard kernel, then the computation time does not depend on some specific optimization techniques.

We summarize the result in Table 1, where nnz$(\cdot)$ means the number of non-zero value and t the number of test points. As we can see, with $d = O(l^2)$ set, our two algorithms take much less time, and also significantly reduce the the overall time with the dominant training time decreased.

For RKS, only a 'seed' is needed for random number generator to reproduce random sequence for the testing [Dai *et al.*, 2014]. After getting $\mathbf{w}$ by Eq. (11), we practically multiply it with $\mathbf{Q}$ of Algorithm 1 or 2 for the testing. Finally, our algorithms only stores a 'seed' and a vector $\mathbf{Q}\mathbf{w}^T$ for the testing, taking comparable storage with other methods.

	Mapping	**Training**	**Prediction**
$\mathcal{M}_1$	$O(\text{nnz}(\mathbf{X})n)$	$O(n^3)$	$O(t(m)n)$
$\mathcal{M}_2$	$O(\text{nnz}(\mathbf{X})d)$	$O(nd^2)$	$O(t(m+1)d)$
$\mathcal{M}_3$	$O(\text{nnz}(\mathbf{X})d$ $+l^2d + nld)$	$O(nl^2)$	$O(t(m+1)d$ $+dl)$
$\mathcal{M}_4$	$O(\text{nnz}(\mathbf{X})d$ $+l^2d + nd\log l)$	$O(nl^2)$	$O(t(m+1)d$ $+dl)$

Table 1: Time complexity of $\mathcal{M}_1$ to $\mathcal{M}_4$ representing 4 methods respectively, i.e., Kernel method, RKS, TEFM-G and TEFM-S.

4 Empirical Evaluation

In this section, we empirically demonstrate the superiority of the proposed methods on the Gaussian kernel matrix approximation and related regression task. We compare the following random feature map methods:

1. Random Kitchen Sinks (denoted by RKS).
2. Quasi-Monte Carlo method [Yang *et al.*, 2014] (denoted by Quasi). We use Digital Net to generate QMC sequence, as it yields the lowest approximation error and supports the high dimensional data [Yang *et al.*, 2014].
3. Compact feature maps [Hamid *et al.*, 2014] (denoted by Comp). The proposed method should not be restricted to the polynomial kernel, we thus apply it to Gaussian kernel in our experiments.
4. Fastfood method [Le *et al.*, 2013] (denoted by Ffood). We use 'Hadamard features' for its better performance.
5. Our proposed algorithms TEFM-G and TEFM-S.

We use six publicly available real-world datasets listed in Table 2, and they can be downloaded from LIBSVM website [Chang and Lin, 2011] or UCI machine learning repository. All our experiments are run on Matlab with single thread mode in order to fairly compare the running time.

Dataset	**# instances**	**# features**
Cpu	6554	21
A9a	48842	123
BlogFeedback	60021	280
SliceLocalization	53500	384
UJIIndoorLoc	21048	520
Mnist	70000	784

Table 2: Details of datasets in our experiments.

4.1 Kernel Approximation Quality

We compare the features generated by above methods on the kernel approximation. We report the approximation accuracy of each method measured by $\|\mathbf{K} - \mathbf{K}_{\text{app}}\|/\|\mathbf{K}\|$, where $\mathbf{K}_{\text{app}}$ is the approximation matrix reconstructed by the random features. To facilitate the computation of full kernel matrix $\mathbf{K}$ for the comparison purposes, we randomly sample these datasets so that only 6554, 8141, 8733, 8917, 6646 and 6006 instances are used, respectively.

We summarize the results in Figure 1. The x-axis stands for the number of features l. In our two algorithms, typically approximation error will turn down as d or l increases, however, d cannot be very large or even infinite, otherwise

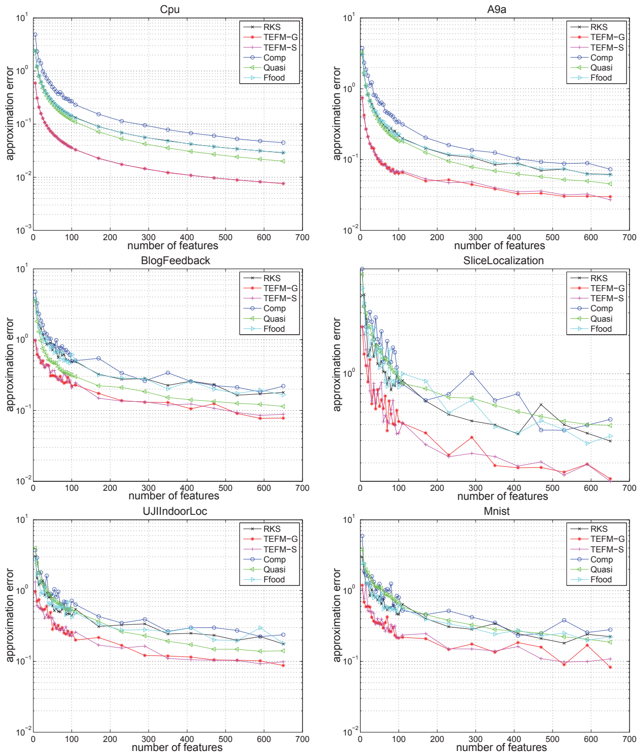

Figure 1: Kernel approximation.

our two feature generation algorithms will be time consuming. Thus, d actually affects the tradeoff between the accuracy and computation. If we keep $d = 4l$, then Figure 1 shows that our algorithms incur nearly *half* of approximation error compared to others, which almost coincides with our theoretical result. This implies that it is possible to train the leaning tasks with fewer features while maintaining the accuracy. In particular, our methods beat the counterpart 'Comp' which implements data-independent subspace embedding for original features [Hamid *et al.*, 2014]. This also reflects that data-independent subspace embedding is not suitable for the random features on shift-invariant kernels.

4.2 Performance on KRR

First, we compare the prediction ability of different features on KRR. The parameters are chosen by cross-validation. We report the relative root mean square error (RMSE) of each method in the testing datasets. The relative RMSE is defined by $\|\mathbf{y} - \mathbf{y}_o\|/\|\mathbf{y}\|$, where $\mathbf{y}_o$ is the predicted value and $\mathbf{y}$ is the ground truth. We consider the regression dataset 'SliceLocalization' and list the result in Figure 2. As we can see, using much fewer informative features, our two algorithms can obtain the same accuracy as others. Thus, training using our fewer features can keep the learning accuracy and require much less time. This also implies that, accurate learning can be achieved by applying the features that ensure a good kernel matrix approximation.

Second, we report the prediction error versus the time (*includes feature mapping time and training time*) in Figure 3. Our algorithms require slightly more feature map time.

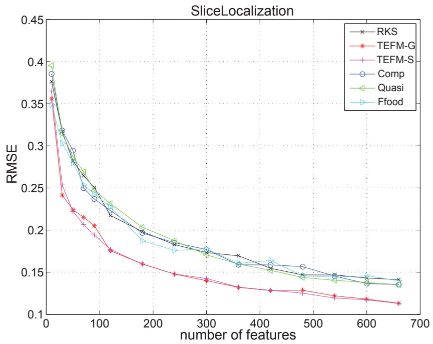

Figure 2: Accuracy comparison on regression.

They, however, use fewer features to get the same approximation performance. Therefore, this substantially decreases the training time. As can be seen, even we consider the time used for the feature map, our two algorithms still outperform the other methods with smaller RMSE achieved. Particularly, the advantage of our methods will be more evident when we need a much smaller RMSE. The reason is that, to obtain a much smaller RMSE, more features in other methods should be generated, and as the feature number grows large, the time used for training regression task will be dominant compared to the time for feature map.

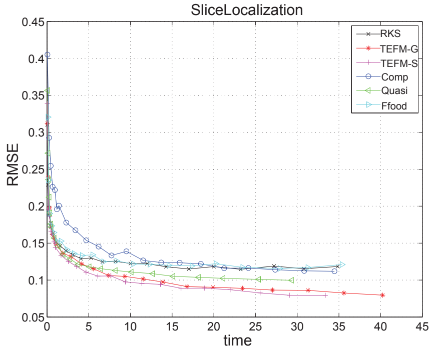

Figure 3: Time comparison on regression.

5 Conclusion

In this paper, we describe an effective feature map method that can generate better features to make the learning tasks trained accurately and more efficiently. The basic idea of our method is to combine fast data-dependent subspace embedding with the RKS. We also present two algorithms TEFM-G and TEFM-S. Our theoretical analysis indicates these two algorithms achieve better kernel approximation and faster training on learning tasks without losing precision. We report extensive empirical results of our algorithms on several real-world datasets, which support our analysis and demonstrate a good practical performance.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper. The work described in this paper was fully supported by the National Grand Fundamental Research 973 Program of China (No. 2014CB340401 and No. 2014CB340405), the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK 413213 and CUHK 14205214), and Microsoft Research Asia Regional Seed Fund in Big Data Research (Grant No. FY13-RES-SPONSOR-036).

References

[Bingham and Mannila, 2001] Ella Bingham and Heikki Mannila. Random projection in dimensionality reduction: applications to image and text data. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 245–250. ACM, 2001.

[Cesa-Bianchi *et al.*, 2014] Nicolò Cesa-Bianchi, Yishay Mansour, and Ohad Shamir. On the complexity of learning with kernels. *arXiv preprint arXiv:1411.1158*, 2014.

[Chang and Lin, 2011] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[Cortes *et al.*, 2010] Corinna Cortes, Mehryar Mohri, and Ameet Talwalkar. On the impact of kernel approximation on learning accuracy. In *International Conference on Artificial Intelligence and Statistics*, pages 113–120, 2010.

[Dai *et al.*, 2014] Bo Dai, Bo Xie, Niao He, Yingyu Liang, Anant Raj, Maria-Florina F Balcan, and Le Song. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems*, pages 3041–3049, 2014.

[Halko *et al.*, 2011] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[Hamid *et al.*, 2014] Raffay Hamid, Ying Xiao, Alex Gittens, and Dennis DeCoste. Compact random feature maps. 2014.

[Le *et al.*, 2013] Quoc Le, Tamás Sarlós, and Alex Smola. Fastfood approximating kernel expansions in loglinear time. In *Proceedings of the international conference on machine learning*, 2013.

[Lopez-paz *et al.*, 2014] David Lopez-paz, Suvrit Sra, Alex Smola, Zoubin Ghahramani, and Bernhard Schoelkopf. Randomized nonlinear component analysis. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1359–1367, 2014.

[Nelson and Nguyn, 2014] Jelani Nelson and Huy L Nguyn. Lower bounds for oblivious subspace embeddings. In *Automata, Languages, and Programming*, pages 883–894. Springer, 2014.

[Nesterov, 2005] Yu Nesterov. Smooth minimization of non-smooth functions. *Mathematical programming*, 103(1):127–152, 2005.

[Rahimi and Recht, 2007] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2007.

[Rahimi and Recht, 2009] Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in neural information processing systems*, pages 1313–1320, 2009.

[Rudin, 1990] Walter Rudin. *Fourier analysis on groups*. Number 12. John Wiley & Sons, 1990.

[Saunders *et al.*, 1998] Craig Saunders, Alexander Gammerman, and Volodya Vovk. Ridge regression learning algorithm in dual variables. In *Proceedings of the 15th International Conference on Machine Learning*, pages 515–521. Morgan Kaufmann, 1998.

[Schölkopf *et al.*, 2001] Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. A generalized representer theorem. In *Computational learning theory*, pages 416–426. Springer, 2001.

[Tropp, 2011] Joel A Tropp. Improved analysis of the subsampled randomized hadamard transform. *Advances in Adaptive Data Analysis*, 3(01n02):115–126, 2011.

[Tropp, 2015] Joel A Tropp. An introduction to matrix concentration inequalities. *arXiv preprint arXiv:1501.01571*, 2015.

[Yang *et al.*, 2011a] Haiqin Yang, Irwin King, and Michael R. Lyu. *Sparse Learning Under Regularization Framework*. LAP Lambert Academic Publishing, April 2011.

[Yang *et al.*, 2011b] Haiqin Yang, Zenglin Xu, Jieping Ye, Irwin King, and Michael R. Lyu. Efficient sparse generalized multiple kernel learning. *IEEE Transactions on Neural Networks*, 22(3):433–446, March 2011.

[Yang *et al.*, 2012] 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 *Advances in neural information processing systems*, pages 476–484, 2012.

[Yang *et al.*, 2014] Jiyan Yang, Vikas Sindhwani, Haim Avron, and Michael Mahoney. Quasi-monte carlo feature maps for shift-invariant kernels. In *Proceedings of The 31st International Conference on Machine Learning*, pages 485–493, 2014.

[Zhang *et al.*, 2013] Lijun Zhang, Mehrdad Mahdavi, Rong Jin, Tianbao Yang, and Shenghuo Zhu. Recovering the optimal solution by dual random projection. In *Conference on Learning Theory*, pages 135–157, 2013.