Automated Spectral Kernel Learning

Abstract

The generalization performance of kernel methods is largely determined by the kernel, but common kernels are stationary thus input-independent and output-independent, that limits their applications on complicated tasks. In this paper, we propose a powerful and efficient spectral kernel learning framework and learned kernels are dependent on both inputs and outputs, by using non-stationary spectral kernels and flexibly learning the spectral measure from the data. Further, we derive a data-dependent generalization error bound based on Rademacher complexity, which estimates the generalization ability of the learning framework and suggests two regularization terms to improve performance. Extensive experimental results validate the effectiveness of the proposed algorithm and confirm our theoretical results.

Introduction

Kernel methods are important non-linear approaches in statistical machine learning with complete learning frameworks and excellent statistical properties. Although kernel methods have achieved great success in many traditional applications over past decades, they show relatively inferior performance on complicated tasks nowadays. The critical and fundamental limitation of these kernels has been revealed that they are both *stationary* and *monotony* (Bengio, Delalleau, and Roux 2006). Stationary kernels only depends on the distance $\|\boldsymbol{x} - \boldsymbol{x}'\|$ thus they are independent on the value of inputs $\boldsymbol{x}$. The *monotony* property shows that the stationary kernel value decreases over distance thus ignoring long-range interdependence. For example, commonly used Gaussian and Laplacian kernels based on shift-invariant kernel functions $k(\tau)$, only depending on the distance $\tau = \boldsymbol{x} - \boldsymbol{x}'$. Corresponding feature mappings are independent on inputs but also neglect latent long-range correlations.

Spectral approaches were proposed to fully characterize all stationary kernels with a concise representation form, such as sparse spectrum kernels (Quiñonero-Candela et al. 2010), sparse mixture kernels (Wilson and Adams 2013) and random Fourier features methods to handle with large scale settings (Rahimi and Recht 2007; Le, Sarlós, and Smola 2013; Yang et al. 2014; Yu et al. 2016). With sound theoretical guarantees, namely Bochner's theorem (Rudin 1962; Stein 2012), spectral kernels are constructed from the inverse Fourier transform in the frequency domain. Although approximate spectral representation provides an efficient approach for stationary spectral kernels, the performance of random features is limited because stationary kernels are *input-independent* and *output-independent*.

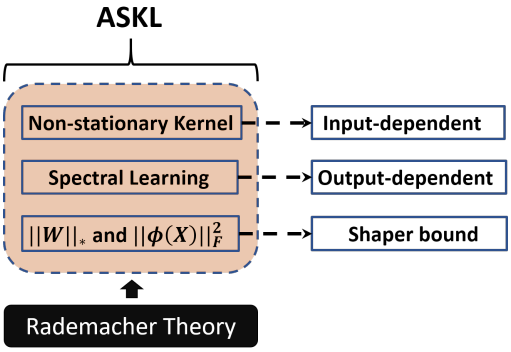

Figure 1: Overview of Automated Spectral Kernel Learning.

Similar to Bochner's theorem, Yaglom's theorem provides a more general spectral characterization approach which encompasses both stationary and non-stationary kernels via inverse Fourier transform (Yaglom 1987; Samo and Roberts 2015). Recently, due to its general and concise spectral statement, non-stationary kernels have been applied to Gaussian process regression framework (Remes, Heinonen, and Kaski 2017; Ton et al. 2018; Sun et al. 2019).

In this paper, we propose a generalized learning framework, namely automated spectral kernel learning `ASKL`, to learn feature mappings not only from inputs but also outputs. A brief overview is illustrated in Figure 1. On the algorithmic front, `ASKL` consists: (1) non-stationary kernels to obtain input-dependent features, (2) automatically spectral kernel learning to make feature output-dependent, (3) regularization terms to achieve shaper error bound. On the theoretical front, the theoretical underpinning is Rademacher complexity theory, which indicates how factors affect the performance and suggest ways to improve the algorithm.

Background

In common supervised learning cases, training samples $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)_{i=1}^n\}$ are drawn i.i.d. from a fixed but unknown distribution ρ on $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} = \mathbb{R}^d$ is the input space and $\mathcal{Y} \subseteq \mathbb{R}^K$ is the output space in single-valued ($K = 1$) or vector-valued ($K > 1$) forms. The goal is to learn an estimator $f : \mathcal{X} \to \mathcal{Y}$, which outputs K predicted labels. We define a standard hypotheses space for kernel methods

$$\mathcal{H} = \left\{ \boldsymbol{x} \to f(\boldsymbol{x}) = \boldsymbol{W}^T \phi(\boldsymbol{x}) \right\},$$

where $\boldsymbol{W} \in \mathbb{R}^{D \times K}$ is model weights, $\phi(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}^D$ is a nonlinear feature mapping. For kernel methods, $\phi(\boldsymbol{x})$ is an implicit feature mapping associated with a Mercer kernel $k(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \rangle$. To improve efficiency but also remain favorable accuracy, random Fourier features were used to approximate kernel with an explicit feature mapping $\phi(\boldsymbol{x})$ via $k(\boldsymbol{x}, \boldsymbol{x}') \approx \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') \rangle$ (Rahimi and Recht 2007).

In statistical learning theory, the supervised learning problem is to minimize the expected loss on $\mathcal{X} \times \mathcal{Y}$

$$\inf_{f \in \mathcal{H}} \mathcal{E}(f), \quad \mathcal{E}(f) = \int_{\mathcal{X} \times \mathcal{Y}} \ell(f(\boldsymbol{x}), \boldsymbol{y}) \, d\rho(\boldsymbol{x}, \boldsymbol{y}), \qquad (1)$$

where ℓ is a loss function related to specific tasks.

Stationary Kernels

The connection between the stationary kernel $k(\tau)$ and its spectral density $s(\boldsymbol{\omega})$ is revealed in Bochner's theorem via inverse Fourier transform.

Theorem 1 (Bochner's theorem)**.** *A stationary continuous kernel $k(\boldsymbol{x}, \boldsymbol{x}') = k(\boldsymbol{x} - \boldsymbol{x}')$ on $\mathbb{R}^d$ is positive definite if and only if it can be represented as*

$$k(\boldsymbol{x}, \boldsymbol{x}') = \int_{\mathbb{R}^d} e^{i\boldsymbol{\omega}^T(\boldsymbol{x} - \boldsymbol{x}')} s(\boldsymbol{\omega}) d\boldsymbol{\omega}, \qquad (2)$$

where $s(\boldsymbol{\omega})$ is a non-negative measure.

The spectral density $s(\boldsymbol{\omega})$ is a probability density function related to the corresponding kernel. From inverse Fourier transform, we find that spectral kernels are highly relevant to the probability measure $s(\boldsymbol{\omega})$. E.g. Gaussian kernel with parameter σ corresponds to Gaussian distribution $\mathcal{N}(0, 1/\sigma)$.

Non-stationary Kernels

While *stationary* and *monotony* kernels ignore input-dependent information and long-range relations, non-stationary kernels alleviate those restrictions because they depend on inputs themselves (Samo and Roberts 2015).

Theorem 2 (Yaglom's theorem)**.** *A general kernel $k(\boldsymbol{x}, \boldsymbol{x}')$ is positive definite on $\mathbb{R}^d$ is positive define if and only if it admits the form*

$$k(\boldsymbol{x}, \boldsymbol{x}') = \int_{\mathbb{R}^d \times \mathbb{R}^d} e^{i(\boldsymbol{\omega}^T \boldsymbol{x} - \boldsymbol{\omega}'^T \boldsymbol{x}')} \mu(d\boldsymbol{\omega}, d\boldsymbol{\omega}'), \qquad (3)$$

where $\mu(d\boldsymbol{\omega}, d\boldsymbol{\omega}')$ is the Lebesgue–Stieltjes measure associated to some positive semi-definite (PSD) spectral density function $s(\boldsymbol{\omega}, \boldsymbol{\omega}')$ with bounded variations.

When μ is concentrated on the diagonal $\boldsymbol{\omega} = \boldsymbol{\omega}'$, the spectral characterization of stationary kernels in the Bochner's theorem is recovered. $s(\boldsymbol{\omega}, \boldsymbol{\omega}')$ is a joint probability density.

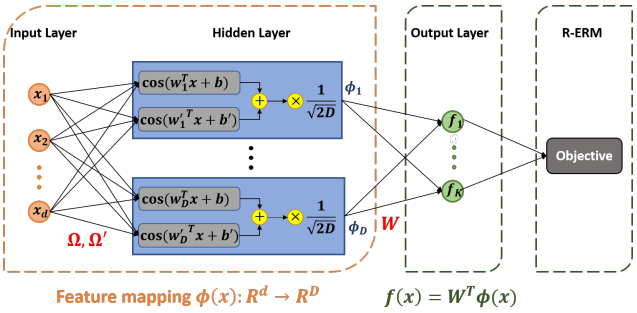

Figure 2: The architecture of learning framework

Automated Spectral Kernel Learning

In this section, we introduce the learning framework for arbitrary kernel-based supervised applications as below:

- We present learning framework with the minimization objective which combines empirical risk minimization with feature mapping and additional regularization terms.
- Spectral representation for non-stationary spectral kernels based on Yaglom's theorem is conducted.
- We apply first-order gradient approaches to solve the minimization objective. We update frequency matrices $\boldsymbol{\Omega}, \boldsymbol{\Omega}'$ together with model weights $\boldsymbol{W}$ via backpropagation.

Learning Framework

The minimization of expected loss (1) is hard to estimate in practical problems. In this paper, we combine the regularized empirical risk minimization (R-ERM) framework with two additional regularization terms

$$\underset{\boldsymbol{W}, \boldsymbol{\Omega}, \boldsymbol{\Omega}'}{\arg\min} \underbrace{\frac{1}{n} \sum_{i=1}^{n} \ell(f(\boldsymbol{x}_i), \boldsymbol{y}_i)}_{g(\boldsymbol{W})} + \lambda_1 \|\boldsymbol{W}\|_* + \lambda_2 \|\phi(\boldsymbol{X})\|_F^2, \qquad (4)$$

where both $\phi(\boldsymbol{X}) \in \mathbb{R}^{D \times n}$ on all data and $f(\boldsymbol{x}_i) = \boldsymbol{W}^T \phi(\boldsymbol{x}_i) \in \mathbb{R}^D$ use spectral representation for non-stationary kernels ϕ which is presented in detail below

$$\phi(\boldsymbol{x}) = \frac{1}{\sqrt{2D}} \left[\cos(\boldsymbol{\Omega}^T \boldsymbol{x} + \boldsymbol{b}) + \cos(\boldsymbol{\Omega}'^T \boldsymbol{x} + \boldsymbol{b}') \right].$$

The trace norm $\|\boldsymbol{W}\|_*$ and the squared Frobenius norm $\|\phi(\boldsymbol{X})\|_F^2$ exerts constraints on updating $\boldsymbol{W}$ and frequency matrices $\boldsymbol{\Omega}, \boldsymbol{\Omega}'$, respectively. Those two regularization terms especially squared Frobenius norm is rarely used in ERM. We exert them to obtain tighter Rademacher complexity bounds based on the theoretical analysis in the next section.

In the minimization objective (4), we update model weights $\boldsymbol{W}$ and frequency pairs $\boldsymbol{\Omega}, \boldsymbol{\Omega}'$ to learn feature mappings both input-dependent (non-stationary kernel) and output-dependent (automated spectral density learning). The spectral density surface $s(\boldsymbol{\omega}, \boldsymbol{\omega}')$ (joint probability density) determines the performance of spectral kernels. As shown in Figure 2, the architecture can be regarded as single hidden layer neural networks with cosine as activation and spectral density $s(\boldsymbol{\omega}, \boldsymbol{\omega}')$ is automatically learned by updating frequency matrices $\boldsymbol{\Omega}, \boldsymbol{\Omega}'$ via backpropagation (Rumelhart et al. 1988). In this network, only $\boldsymbol{W}$ and $\boldsymbol{\Omega}, \boldsymbol{\Omega}'$ are trainable.

Non-stationary Spectral Kernels Representation

According to Yaglom's theorem, to produce a positive semi-definite (PSD) kernel, the spectral density needs to be a PSD function. Therefore, we construct a PSD spectral density $s(\boldsymbol{\omega}, \boldsymbol{\omega}')$ by symmetrizing $s(\boldsymbol{\omega}, \boldsymbol{\omega}') = s(\boldsymbol{\omega}, \boldsymbol{\omega}')$ and including sufficient diagonal components $s(\boldsymbol{\omega}, \boldsymbol{\omega})$ and $s(\boldsymbol{\omega}', \boldsymbol{\omega}')$. As a result, we combine exponential components and the corresponding integrated spectral density $s(\boldsymbol{\omega}, \boldsymbol{\omega}')$

$$\mathcal{E}_{\boldsymbol{\omega},\boldsymbol{\omega}'}(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{4}\big[e^{i(\boldsymbol{\omega}^T\boldsymbol{x} - \boldsymbol{\omega}'^T\boldsymbol{x}')} + e^{i(\boldsymbol{\omega}'^T\boldsymbol{x} - \boldsymbol{\omega}^T\boldsymbol{x}')} + e^{i(\boldsymbol{\omega}^T\boldsymbol{x} - \boldsymbol{\omega}^T\boldsymbol{x}')} + e^{i(\boldsymbol{\omega}'^T\boldsymbol{x} - \boldsymbol{\omega}'^T\boldsymbol{x}')}\big].$$

The PSD kernel can be rewritten as

$$k(\boldsymbol{x}, \boldsymbol{x}') = \int_{\mathbb{R}^d \times \mathbb{R}^d} \mathcal{E}_{\boldsymbol{\omega},\boldsymbol{\omega}'}(\boldsymbol{x}, \boldsymbol{x}') s(\boldsymbol{\omega}, \boldsymbol{\omega}') d\boldsymbol{\omega} d\boldsymbol{\omega}'. \quad (5)$$

where $s(\boldsymbol{\omega}, \boldsymbol{\omega}')$ is the spectral density surface. Similar spectral representation of non-stationary kernels are also used in (Samo and Roberts 2015; Remes, Heinonen, and Kaski 2017). When $\boldsymbol{\omega} = \boldsymbol{\omega}'$, non-stationary kernel in (5) degrades into stationary kernel as in Bochner's theorem (2) and the probability density function becomes univariate $s(\boldsymbol{\omega})$.

In the stationary case, random Fourier features are used to approximate stationary kernels (Rahimi and Recht 2007). Similarly, in the non-stationary case, we can approximate (5) with Monte Carlo random sampling

$$\begin{aligned}
& k(\boldsymbol{x}, \boldsymbol{x}') = \int_{\mathbb{R}^d \times \mathbb{R}^d} \mathcal{E}_{\boldsymbol{\omega},\boldsymbol{\omega}'}(\boldsymbol{x}, \boldsymbol{x}') s(\boldsymbol{\omega}, \boldsymbol{\omega}') d\boldsymbol{\omega} d\boldsymbol{\omega}' \\
= \; & \mathbb{E}_{\boldsymbol{\omega},\boldsymbol{\omega}' \sim s}\left[\mathcal{E}_{\boldsymbol{\omega},\boldsymbol{\omega}'}(\boldsymbol{x}, \boldsymbol{x}')\right] \\
= \; & \mathbb{E}_{\boldsymbol{\omega},\boldsymbol{\omega}' \sim s} \frac{1}{4}\big[\cos(\boldsymbol{\omega}^T\boldsymbol{x} - \boldsymbol{\omega}'^T\boldsymbol{x}') + \cos(\boldsymbol{\omega}'^T\boldsymbol{x} - \boldsymbol{\omega}^T\boldsymbol{x}') \\
& + \; \cos(\boldsymbol{\omega}^T\boldsymbol{x} - \boldsymbol{\omega}^T\boldsymbol{x}') + \cos(\boldsymbol{\omega}'^T\boldsymbol{x} - \boldsymbol{\omega}'^T\boldsymbol{x}')\big] \\
\approx \; & \frac{1}{4D}\sum_{i=1}^{D}\Big[\cos(\boldsymbol{\omega}_i^T\boldsymbol{x} - \boldsymbol{\omega}_i'^T\boldsymbol{x}') + \cos(\boldsymbol{\omega}_i'^T\boldsymbol{x} - \boldsymbol{\omega}_i^T\boldsymbol{x}') \\
& + \; \cos(\boldsymbol{\omega}_i^T\boldsymbol{x} - \boldsymbol{\omega}_i^T\boldsymbol{x}') + \cos(\boldsymbol{\omega}_i'^T\boldsymbol{x} - \boldsymbol{\omega}_i'^T\boldsymbol{x}')\Big] \\
= \; & \langle \psi(\boldsymbol{x}), \psi(\boldsymbol{x}') \rangle
\end{aligned}$$

where $(\boldsymbol{\omega}_i, \boldsymbol{\omega}_i')_{i=1}^D \overset{\text{i.i.d.}}{\sim} s(\boldsymbol{\omega}, \boldsymbol{\omega}')$, D is the sampling number and random Fourier feature mapping of spectral kernel is

$$\psi(\boldsymbol{x}) = \frac{1}{\sqrt{4D}}\begin{bmatrix}\cos(\boldsymbol{\Omega}^T\boldsymbol{x}) + \cos(\boldsymbol{\Omega}'^T\boldsymbol{x}) \\ \sin(\boldsymbol{\Omega}^T\boldsymbol{x}) + \sin(\boldsymbol{\Omega}'^T\boldsymbol{x})\end{bmatrix},$$

where features mapping are $\mathbb{R}^d \to \mathbb{R}^{2D}$. To alleviate computational costs, we use the following mapping $\mathbb{R}^d \to \mathbb{R}^D$

$$\phi(\boldsymbol{x}) = \frac{1}{\sqrt{2D}}\left[\cos(\boldsymbol{\Omega}^T\boldsymbol{x} + \boldsymbol{b}) + \cos(\boldsymbol{\Omega}'^T\boldsymbol{x} + \boldsymbol{b}')\right], \quad (6)$$

where frequency matrices $\boldsymbol{\Omega}, \boldsymbol{\Omega}' \in \mathbb{R}^{d \times D}$ are integrated with $(\boldsymbol{\omega}_i, \boldsymbol{\omega}_i')_{i=1}^D$ that $\boldsymbol{\Omega} = [\boldsymbol{\omega}_1, \cdots, \boldsymbol{\omega}_D]$ and $\boldsymbol{\Omega}' = [\boldsymbol{\omega}_1', \cdots, \boldsymbol{\omega}_D']$. The phase vectors $\boldsymbol{b}, \boldsymbol{b}'$ are drawn uniformly from $[0, 2\pi]^D$. In fact, spectral kernels induced by $\psi(\boldsymbol{x})$ and $\phi(\boldsymbol{x})$ are equivalent in expectation manners.

Remark 1. *Equation (6) provides random Fourier feature mapping for non-stationary kernels. Suitable frequency matrices $\boldsymbol{\Omega}, \boldsymbol{\Omega}'$ are the key to obtain favorable performance. Traditional kernel methods generate frequency matrices sampled from the **assigned** spectral density $s(\boldsymbol{\omega}, \boldsymbol{\omega}')$, e.g frequency matrices correspond to random Gaussian matrix when approximating Gaussian kernels. In the proposed algorithm* ASKL*, frequency matrices $\boldsymbol{\Omega}, \boldsymbol{\Omega}'$ are jointly **learned** together with model weights $\boldsymbol{W}$ during training.*

Update Trainable Matrices

As shown in the above section, non-stationary spectral kernels are input-dependent. Further, we propose a general learning framework, namely Automated Spectral Kernel Learning (ASKL). The learning framework **optimizes** frequency matrices for non-stationary kernels $\boldsymbol{\Omega}, \boldsymbol{\Omega}'$ according to outputs via backpropagation. Such that the learned spectral kernels are both *input-dependent* and *output-dependent*.

The learning frame ASKL can be easily solved by first-order gradient descent methods, such as stochastic gradient descent (SGD) and its variants Adadelta (Zeiler 2012) and Adam (Kingma and Ba 2014). By backpropagation of gradients, we derive how to update estimator weights $\boldsymbol{W}$ and frequency matrices $\boldsymbol{\Omega}, \boldsymbol{\Omega}'$. In the following analysis, we consider a general case of mini-batch gradient descent with using m examples in each iteration, such that full gradient descent is the special case $m = n$ and stochastic gradient descent (SGD) when $m = 1$ one example is used.

1. Update $\boldsymbol{W}$ in Proximal Gradient Approach To minimize learning objective (4), we use first-order gradient descent algorithms update $\boldsymbol{W}$ in the direction of negative gradient. The gradient of $\boldsymbol{W}$ depends on empirical loss and trace norm in (4), but trace norm is nondifferentiable on many points for one dimension (unlike hinge loss and Relu are nondifferentiable only on zero), thus the derivative/subgradient of the trace norm cannot be applied in standard descent approaches. We employ singular value thresholding (SVT) to solve the minimization of trace norm with proximal gradient descent (Cai, Candès, and Shen 2010).

We simply the update of $\boldsymbol{W}$ in two steps and put detailed deduction process in the Appendix:
(1) Update $\boldsymbol{W}$ with SGD on empirical loss

$$\boldsymbol{Q} = \boldsymbol{W}^t - \eta \nabla g(\boldsymbol{W}^t),$$

where $\frac{1}{u}$ is the learning rate and the gradient of empirical loss on m-batch examples is

$$\begin{aligned}
\nabla g(\boldsymbol{W}) = & \frac{1}{m}\sum_{i=1}^{m}\frac{\partial \ell(f(\boldsymbol{x}_i), \boldsymbol{y})}{\partial \boldsymbol{W}} \\
= & \frac{1}{m}\sum_{i=1}^{m}\phi(\boldsymbol{x}_i) \cdot \left[\frac{\partial \ell(f(\boldsymbol{x}_i), \boldsymbol{y})}{\partial f(\boldsymbol{x}_i)}\right]^T \in \mathbb{R}^{D \times K},
\end{aligned} \quad (7)$$

and $\boldsymbol{Q}$ is an intermediate matrix and m examples are used.
(2) Update $\boldsymbol{W}$ with SVT on trace norm

$$\boldsymbol{W}^{t+1} = \boldsymbol{U}\text{diag}\left(\{\sigma_j - \lambda\eta\}_+\right)\boldsymbol{V}^T, \quad (8)$$

where $\boldsymbol{Q} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ is the singular values decomposition, $\boldsymbol{\Sigma}$ is the diagonal $\text{diag}(\{\sigma_j\}_{1 \le i \le r})$ and r is the rank of $\boldsymbol{Q}$.

2. Update $\mathbf{\Omega}, \mathbf{\Omega}'$ Using the chain rule for computing the derivative, the derivative of objective only depends on empirical loss $\ell(f(\boldsymbol{x}_i), \boldsymbol{y})$ in terms of $\mathbf{\Omega}, \mathbf{\Omega}'$. In the following, we derive the gradient of $\mathbf{\Omega}$ as an example. Both empirical risk $\ell(f(\boldsymbol{x}_i), \boldsymbol{y})$ and squared Frobenius norm $\|\phi(\boldsymbol{X})\|_F^2$ are differentiable with respect to $\mathbf{\Omega}, \mathbf{\Omega}'$. That is

$$\frac{1}{m}\sum_{i=1}^{m}\frac{\partial \ell(f(\boldsymbol{x}_i), \boldsymbol{y})}{\partial \mathbf{\Omega}} + \frac{\partial \|\phi(\boldsymbol{X})\|_F^2}{\partial \mathbf{\Omega}},$$

where derivatives w.r.t $\mathbf{\Omega}$ are

$$\frac{\partial \ell(f(\boldsymbol{x}_i), \boldsymbol{y})}{\partial \mathbf{\Omega}} = \boldsymbol{x}_i \cdot \left[\boldsymbol{D}\cdot \boldsymbol{W}\cdot \frac{\partial \ell(f(\boldsymbol{x}_i), \boldsymbol{y})}{\partial f(\boldsymbol{x}_i)}\right]^T, \quad (9)$$

$$\frac{\partial \|\phi(\boldsymbol{X})\|_F^2}{\partial \mathbf{\Omega}} = \frac{1}{m}\sum_{i=1}^{m} 2\boldsymbol{x}_i \cdot \phi^T(\boldsymbol{x}_i)\cdot \boldsymbol{D}, \quad (10)$$

and $\boldsymbol{D}$ is a diagonal matrix in $D \times D$ size filled with a vector

$$\boldsymbol{D} = \text{diag}\left\{\frac{-1/\sqrt{2D}}{\sin(\mathbf{\Omega}^T\boldsymbol{x}_i + b)}\right\}_{D\times D}.$$

Specific Loss functions For gradients in (7) and (9), only gradients in terms of loss function $\frac{\partial \ell(f(\boldsymbol{x}_i), \boldsymbol{y})}{\partial f(\boldsymbol{x}_i)}$ are uncertain. Here, we provide common loss functions and their gradients.

- Hinge loss for classification problems.
  Let the label $\boldsymbol{y} = [0, \cdots, 0, 1, 0, \cdots, 0]^T$ only one element (its category) is not zero. Hinge loss is defined as $\ell(f(\boldsymbol{x}_i), \boldsymbol{y}) = |1 - (\boldsymbol{y}^T f(\boldsymbol{x}_i) - \max_{\boldsymbol{y}'\neq \boldsymbol{y}} \boldsymbol{y}'^T f(\boldsymbol{x}_i))|_+$ and it is non-differentiable when the margin meets zero. The sub-gradient of hinge loss w.r.t the estimator is

$$\frac{\partial \ell(f(\boldsymbol{x}_i), \boldsymbol{y})}{\partial f(\boldsymbol{x}_i)} = \begin{cases}\mathbf{0}, & \boldsymbol{y}^T f(\boldsymbol{x}_i) - \max\limits_{\boldsymbol{y}'\neq \boldsymbol{y}} \boldsymbol{y}'^T f(\boldsymbol{x}_i) \geq 1,\\ \boldsymbol{y}' - \boldsymbol{y}, & \text{else.}\end{cases}$$

- Squared Loss for regression problems.
  Let $\boldsymbol{y}$ be the K-size vector-valued label where $K > 1$ for multi-label regression and $K = 1$ for univariate regression. Squared loss function is $\ell(f(\boldsymbol{x}_i), \boldsymbol{y}) = \|f(\boldsymbol{x}_i) - \boldsymbol{y}\|_2^2$. Then, the gradient of squared loss is

$$\frac{\partial \ell(f(\boldsymbol{x}_i), \boldsymbol{y})}{\partial f(\boldsymbol{x}_i)} = 2(f(\boldsymbol{x}_i) - \boldsymbol{y}).$$

Theoretical Guarantee

In this section, we analysis generalization performance for our learning model with trace norm regularization. A data-dependent excess risk bound is derived and the analysis can be regarded as statistical learning for single hidden layer neural networks with cosine as the activation function.

Definition 1. *The empirical Rademacher complexity of hypotheses space $\mathcal{H}$ is*

$$\widehat{\mathcal{R}}(\mathcal{H}) = \frac{1}{n}\,\mathbb{E}_\epsilon\left[\sup_{f\in\mathcal{H}}\sum_{i=1}^{n}\sum_{k=1}^{K}\epsilon_{ik} f_k(\boldsymbol{x}_i)\right],$$

where $f_k(\boldsymbol{x}_i)$ is the k-th value of the estimator $f(\boldsymbol{x}_i)$ with K outputs and ϵ_{ik}s are $n \times K$ independent Rademacher random variables with probability $\mathbb{P}(\epsilon_{ik} = +1) = \mathbb{P}(\epsilon_{ik} = -1) = 1/2$. Its deterministic estimate is $\mathcal{R}(\mathcal{H}) = \mathbb{E}\,\widehat{\mathcal{R}}(\mathcal{H})$.

Theorem 3 (Excess Risk Bound)**.** *Assume that $B = \sup_{f\in\mathcal{H}}\|\boldsymbol{W}\|_* < \infty$ and assume the loss function ℓ is L-Lipschitz for $\mathbb{R}^K$ equaipped with the 2-norm, with probability at least $1-\delta$, the following excess risk bound holds*

$$\mathcal{E}(\widehat{f}_n) - \mathcal{E}(f^*) \leq 4\sqrt{2}L\widehat{\mathcal{R}}(\mathcal{H}) + \mathcal{O}\Big(\sqrt{\frac{\log 1/\delta}{n}}\Big), \quad (11)$$

where $f^* \in \mathcal{H}$ is the most accurate estimator in hypotheses space, $\widehat{f}_n$ is the estimator in ERM hypotheses space and

$$\begin{aligned}\widehat{\mathcal{R}}(\mathcal{H}) &\leq \frac{B}{n}\sqrt{K\sum_{i=1}^{n}\langle\phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_i)\rangle}\\ &= \frac{B}{n}\sqrt{\frac{K}{D}\sum_{i=1}^{n}\sum_{j=1}^{D}\frac{1}{2}\Big[\cos\big((\boldsymbol{\omega}_j - \boldsymbol{\omega}_j')^T\boldsymbol{x}_i\big) + 1\Big]}.\end{aligned} \quad (12)$$

The proof is given in Appendix. The error bounds depend on Rademacher complexity term. Due to $\cos\big((\boldsymbol{\omega}_j - \boldsymbol{\omega}_j')^T\boldsymbol{x}_i\big) \leq 1$, Rademacher complexity in (12) is naturally bounded by $\widehat{\mathcal{R}}(\mathcal{H}) \leq B\sqrt{K/n}$, thus the convergence rate is

$$\mathcal{E}(\widehat{f}_n) - \mathcal{E}(f^*) \leq \mathcal{O}\left(B\sqrt{\frac{K}{n}}\right). \quad (13)$$

Based on above theoretical results, we make a some technical comments to present how the factors affects the generalization performance of the proposed algorithm and we use those factors to improve the algorithm :

- **Influence of non-stationary kernels.** As mentioned in the above section, non-stationary kernels depend on inputs themselves instead of the distance between inputs. We explore the effect of non-stationary kernels by Rademacher complexity in (12), which depends on the trace $\sum_{i=1}^{n} k(\boldsymbol{x}_i, \boldsymbol{x}_i)$. Consider the special case of stationary kernel that $\boldsymbol{\omega} = \boldsymbol{\omega}'$. The spectral representation of stationary kernels (shift-invariant kernels) holds diagonals for stationary kernels $k(\boldsymbol{x}_i, \boldsymbol{x}_i) = \cos(\boldsymbol{\omega}^T(\boldsymbol{x}_i - \boldsymbol{x}_i)) = 1$, thus the trace of kernel matrix $\sum_{i=1}^{n} k(\boldsymbol{x}_i, \boldsymbol{x}_i) = n$. While for non-stationary kernels, $k(\boldsymbol{x}_i, \boldsymbol{x}_i) = \cos((\boldsymbol{\omega} - \boldsymbol{\omega}')^T\boldsymbol{x}_i) \in [-1, 1]$. For most instances $\boldsymbol{x}_i$, the diagonals are $k(\boldsymbol{x}_i, \boldsymbol{x}_i) < 1$, so the trace $\sum_{i=1}^{n} k(\boldsymbol{x}_i, \boldsymbol{x}_i) \ll n$. Error bounds of non-stationary spectral kernels is much tighter than error bounds of stationary kernels, that guarantee non-stationary kernels achieve better performance.
- **Influence of spectral learning.** If frequency matrices $\mathbf{\Omega}, \mathbf{\Omega}'$ are just assigned according to specific spectral density $s(\boldsymbol{\omega}, \boldsymbol{\omega}')$, the non-stationary kernels are input-dependent but output-independent. By dynamically optimizing frequency matrices $\mathbf{\Omega}, \mathbf{\Omega}'$ towards to learning tasks, more powerful spectral kernels are acquired. And then spectral measure $s(\boldsymbol{\omega}, \boldsymbol{\omega}')$ of optimal kernels can be estimated form optimized $\mathbf{\Omega}, \mathbf{\Omega}'$. The learned kernels are dependent on both inputs and outputs, offering the better ability of features representation.
- **Using the trace norm $\|\boldsymbol{W}\|_*$ as regularization.** The convergence rate $B = \sup_{f\in\mathcal{H}}\|\boldsymbol{W}\|_* < \infty$ is also dependent on a constant B, that is the supremum of trace

norm $\|\boldsymbol{W}\|_*$ in terms of a specific hypotheses space. As results, the minimization of trace norm $\|\boldsymbol{W}\|_*$ is useful to reduce B and obtain better error bounds. Based on Rademacher complexity theory, the use of trace norm instead of squared Frobenius norm as regularization were also explored for linear estimators in (Yu et al. 2014; Xu et al. 2016; Li et al. 2019).

- **Using squared Frobenius norm $\|\phi(\boldsymbol{X})\|_F^2$ as regularization.** From (11), Rademacher complexity is bounded by the trace of the kernel and it can be written as

$$\sum_{i=1}^{n}\langle\phi(\boldsymbol{x}_i),\phi(\boldsymbol{x}_i)\rangle = \sum_{i=1}^{n}\|\phi(\boldsymbol{x}_i)\|_2^2 = \|\phi(\boldsymbol{X})\|_F^2.$$

  In traditional theoretical learning for Rademacher complexity, the kernel trace cannot be used to improve learning algorithm because feature mappings are constants for specific inputs. For example, the diagonals are $k(\boldsymbol{x}_i,\boldsymbol{x}_i) = 1$ for shift-variant kernels thus the trace is the number of examples. Note that, instead of assigned kernel parameters, our algorithm automatically learns optimal spectral density for kernels, thus the trace of the kernel is no longer a constant and we use it as a regularization term to improve generalization performance.

Experiments

In this section, compared with other algorithms, we evaluate the empirical behavior of our proposed algorithm ASKL on several benchmark datasets to demonstrate the effects of factors used in our algorithm, including the non-stationary spectral kernel, updating spectral density with backpropagation and additional regularization terms.

Experimental Setup

Based on random Fourier features, both our algorithm ASKL and compared methods apply nonlinear feature mapping into a fixed D-dimensional feature space where we set $D = 2000$. We apply Gaussian kernels as basic kernels because Gaussian kernels succeed in many types of data by mapping inputs into infinite-dimensional space, of which frequency matrices $\boldsymbol{\Omega}, \boldsymbol{\Omega}'$ are i.i.d. drawn from Gaussian distributions $\mathcal{N}(0,\sigma^2)$. The generalization ability of algorithms are highly dependent on different parameters on λ_1, λ_2 and Gaussian kernel parameter σ for spectral kernels. For fair comparisons, we tune those parameters to achieve optimal empirical performance for all algorithms on all dataset, by using 5-folds cross-validation and grid search over parameters candidate sets. Regularization parameters are selected in $\lambda_1, \lambda_2 \in \{10^{-10}, 10^{-9}, \cdots, 10^{-1}\}$ and Gaussian kernel parameter σ is selected from candidate set $\sigma \in \{2^{-10}, \cdots, 2^{10}\}$. Accuracy and mean squared error (MSE) are used to evaluate performance for classification and regression, respectively. We implement all algorithms based on Pytorch and use Adam as optimizer with 32 examples in a mini-batch to solve the minimization problem.

Compared Algorithms.

To assess the effectiveness of factors used in our algorithm, we compare the proposed algorithm with several relevant algorithms. As shown in which are special cases of ASKL:
(1) **SK** (Rahimi and Recht 2007): known as random Fourier features for the stationary spectral kernel. This approach directly assigned the spectral density for shift-invariant kernels and uses squared Frobenius norm as regularization term.
(2) **NSK** (Samo and Roberts 2015): Similar to **SK** but it uses spectral representation for non-stationary kernel which was introduced in (6) with assigned frequency matrices.
(3) **SKL** (Huang et al. 2014): Random Fourier features for stationary kernels and squared Frobenius norm as regularization term with updating spectral density during training.
(4) **NSKL**: A special case of ASKL with non-stationary spectral, learned spectral density. But it uses squared Frobenius norm on model weights $\|\boldsymbol{W}\|_F^2$ as regularization.

Algorithm	Kernel	Spectral	Regularizer
SK	Stationary	Assigned	Frob
NSK	Non-stationary	Assigned	Frob
SKL	Stationary	Learned	Frob
NSKL	Non-stationary	Learned	Frob
ASKL	Non-stationary	Learned	Tr + Frob

Table 1: Compared algorithms. Tr and Frob represents trace norm and squared Frobenius norm, respectively.

Datasets. We evaluate the performance of the proposed learning framework ASKL and compared algorithms based on several publicly available datasets, including both classification and regression tasks. Especially, we standardize outputs for regression tasks to $[0, 100]$ for better illustration. To obtain stable results, we run methods on each dataset 30 times with randomly partition such that 80% data for training and 20% data for testing. Further, those multiple test errors allow the estimation of the statistical significance of difference among methods. To explore the influence of factors upon convergence, we evaluate both test accuracy and objective on MNIST dataset (LeCun et al. 1998).

Empirical Results

Empirical results of all algorithms are shown in Table 2, where accuracy is used for classification tasks and root mean squared error (RMSE) is used for regression tasks. We bold results which have the best performance on each dataset, but also mark sub-optimal results with underlines which have a significant difference with the best ones, by using pairwise t-test on results of 30 times repeating data split and training.

The results in Table 2 show: (1) The proposed algorithm ASKL outperforms compared algorithms on almost all dataset that consists of our theoretical analysis. (2) The use of non-stationary kernels brings notable performance improvement but approaches based on stationary kernels still perform well on easy tasks, e.g. *shuttle* and *cpusmall*. (3) Due to the difference between assigned spectral density and learned frequency matrices, there are significant performance gaps between those two groups {SK, NSK} and {SKL, NSKL}, especially on complicated datasets *satimage* and *letter*. It confirms that learning feature mappings in both input-dependent and output-dependent ways leads to better generalization performance. (4) The proposed al-

		SK	NSK	SKL	NSKL	ASKL
Accuracy(↑)	segment	89.93±2.12	90.15±2.08	94.58±1.86	94.37±0.81	**95.02±1.54**
	satimage	74.54±1.35	75.15±1.38	83.61±1.08	83.74±1.34	**85.32±1.45**
	USPS	93.19±2.84	93.81±2.13	95.13±0.91	95.27±1.65	**97.76±1.14**
	pendigits	96.93±1.53	97.39±1.41	98.19±2.30	98.28±1.68	**99.06±1.26**
	letter	76.50±1.21	78.21±1.56	93.60±1.14	94.66±2.21	**95.70±1.74**
	porker	49.80±2.11	51.85±0.97	54.27±2.72	54.69±1.68	**54.85±1.28**
	shuttle	98.17±2.81	98.21±1.46	98.87±1.42	98.74±1.07	**98.98±0.94**
	MNIST	96.03±2.21	96.45±2.16	96.67±1.61	98.03±1.16	**98.26±1.78**
RMSE(↓)	abalone	10.09±0.42	9.71±0.28	8.35±0.28	**7.85±0.42**	7.88±0.16
	space_ga	11.86±0.26	11.58±0.42	11.40±0.18	11.39±0.46	**11.34±0.27**
	cpusmall	2.77±0.71	2.84±0.38	2.56±0.72	2.57±0.63	**2.42±0.48**
	cadata	50.31±0.92	51.47±0.32	47.67±0.33	47.71±0.30	**46.34±0.23**

Table 2: Classification accuracy (%) for classification datasets and RMSE for regression datasets. (↑) means the lager the better while (↓) indicates the smaller the better. We bold the numbers of the best method and underline the numbers of the other methods which are not significantly worse than the best one.

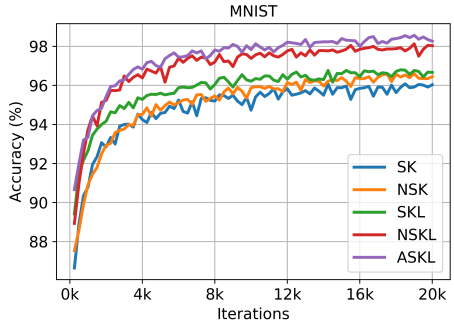

Figure 3: Accuracy curves on MNIST

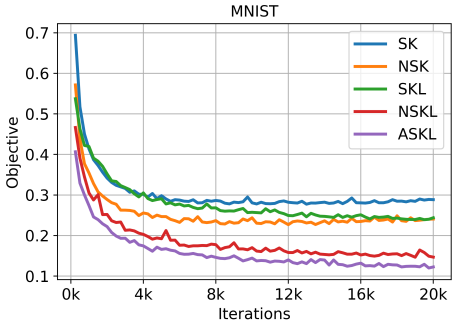

Figure 4: Objective curves on MNIST

gorithm ASKL usually provides better results than NSKL. That shows the effectiveness of regularization terms $\|\boldsymbol{W}\|_*$ and $\|\phi(\boldsymbol{X})\|_F^2$. The experimental results demonstrate the excellent performance and stability of ASKL, corroborating the theoretical analysis and excellent performance of ASKL.

During iterations, test accuracy and objective were recorded for every 200 iterations with batch size 32. Evaluation results in Figure 3 and Figure 4 show that ASKL outperforms other algorithms significantly. Accuracy curves and objective curves are correlated that the smaller objective corresponds to the higher accuracy. Figure 3 and Figure 4 empirically illustrate that ASKL achieves lower error bound than with a fast learning rate.

Conclusion and Discussion

In this paper, we propose automatically kernel learning framework, which jointly learns spectral density and the estimator. Both theoretical analysis and experiments illustrate that the framework obtains significant improvements in performance owing to three key factors: non-stationary spectral kernel, automatically optimizing frequency matrices and two regularization terms. The use of non-stationary spectral kernel makes feature mapping *input-dependent* while updating frequency matrices w.r.t labels that guarantee feature mapping is *output-dependent*, thus learned features obtain powerful representation ability. Further, we derive Rademacher complexity bounds for the algorithm. To achieve shaper bounds, we minimize two matrices norms together with empirical risk minimization framework.

Connection with deep neural networks. Because our learning framework is also a kind of single hidden layer neural networks, we provide a theoretical guarantee for a single hidden layer neural network when the activation is cosine. Those results can be extended to deep neural networks by stacking ϕ in the hierarchical structure. For example l-hidden layers are used that the spectral kernel is $k(\boldsymbol{x}, \boldsymbol{x}') = \langle \phi^l(\phi^{l-1}(\cdots \phi^1(\boldsymbol{x}))), \phi^l(\phi^{l-1}(\cdots \phi^1(\boldsymbol{x}'))) \rangle$. The outputs of other activations such as sigmoid and Relu can also be seen as random feature mapping $\varphi(\boldsymbol{x})$, corresponding kernel function is $k(\boldsymbol{x}, \boldsymbol{x}') = \langle \varphi(\boldsymbol{x}), \varphi(\boldsymbol{x}') \rangle$. It is a possible way to understand deep neural networks in kernels way based on Rademacher complexity theory for kernels.

Extensions. The training process learns suitable kernel parameters and model weights together. For the sake of readability, we reduce the algorithm and theory to a concise form. Moreover, both the algorithm and theory can be improved by local Rademacher complexity and unlabeled samples. Multi-layers of stacked random Fourier feature mapping in a hierarchical structure is a practical extension of our algorithm with theory guarantee, which is a special case of deep neural networks but it's in an interpretable way.

Appendix

Firstly, we introduce some notations used in Rademacher complexity theory. The space of loss function associated with $\mathcal{H}$ is denoted by

$$\mathcal{L} = \left\{ \ell(f(\boldsymbol{x}), \boldsymbol{y}) \;\middle|\; f \in \mathcal{H} \right\}. \tag{14}$$

Definition 2 (Rademacher complexity on loss space)**.** *Assume $\mathcal{L}$ is a space of loss functions as defined in Equation* (14)*. Then the empirical Rademacher complexity of $\mathcal{L}$ is:*

$$\widehat{\mathcal{R}}(\mathcal{L}) = \frac{1}{n}\,\mathbb{E}_\epsilon \left[\sup_{\ell \in \mathcal{L}} \sum_{i=1}^{n} \epsilon_i \ell(f(\boldsymbol{x}_i), \boldsymbol{y}_i) \right],$$

where ϵ_is are independent Rademacher random variables uniformly distributed over $\{\pm 1\}$. Its deterministic counterpart is $\mathcal{R}(\mathcal{L}) = \mathbb{E}\,\widehat{\mathcal{R}}(\mathcal{L})$.

Lemma 1 (Lemma 5 of (Cortes et al. 2016))**.** *Let the loss function ℓ be L-Lipschitz for $\mathbb{R}^K$ equaipped with the 2-norm,*

$$|\ell(f(\boldsymbol{x}), \boldsymbol{y}) - \ell(f(\boldsymbol{x}'), \boldsymbol{y}')| \le L \|f(\boldsymbol{x}) - f(\boldsymbol{x}')\|_2.$$

Then, the following contraction inequation exists

$$\mathcal{R}(\mathcal{L}) \le \sqrt{2} L \mathcal{R}(\mathcal{H}).$$

Proof of Theorem 3

Proof. A standard fact is the derivation of expected loss and empirical means can be controlled by the Rademacher averages over loss space $\mathcal{L}$ (Lemma A.5 of (Bartlett et al. 2005))

$$\mathcal{E}(\widehat{f}_n) - \mathcal{E}(f^*) \le 2 \left[\sup_{f \in \mathcal{H}} \widehat{\mathcal{E}}(f) - \mathcal{E}(f) \right] \le 4\mathcal{R}(\mathcal{L}). \tag{15}$$

Combining (15) with the contraction in Lemma 1 and connection between $\widehat{\mathcal{R}}(\mathcal{H})$ and $\mathcal{R}(\mathcal{H})$ (Lemma A.4 of (Bartlett et al. 2005)), there holds with high probability at least $1 - \delta$

$$\mathcal{E}(\widehat{f}_n) - \mathcal{E}(f^*) \le 4\sqrt{2} L \widehat{\mathcal{R}}(\mathcal{H}) + \mathcal{O}\Big(\sqrt{\frac{\log 1/\delta}{n}}\Big). \tag{16}$$

Then, we estimate empirical Rademacher complexity $\widehat{\mathcal{R}}(\mathcal{H})$

$$\begin{aligned} \widehat{\mathcal{R}}(\mathcal{H}) &= \frac{1}{n}\,\mathbb{E}_{\boldsymbol{\epsilon}} \left[\sup_{f \in \mathcal{H}} \sum_{i=1}^{n} \sum_{k=1}^{K} \epsilon_{ik} f_k(\boldsymbol{x}_i) \right] \\ &= \frac{1}{n}\,\mathbb{E}_{\boldsymbol{\epsilon}} \left[\sup_{f \in \mathcal{H}} \langle \boldsymbol{W}, \boldsymbol{\Phi}_{\boldsymbol{\epsilon}} \rangle \right] \end{aligned} \tag{17}$$

where $\boldsymbol{W}, \boldsymbol{\Phi}_{\boldsymbol{\epsilon}} \in \mathbb{R}^{D \times K}$ and $\langle \boldsymbol{W}, \boldsymbol{\Phi}_{\boldsymbol{\epsilon}} \rangle = \mathrm{Tr}(\boldsymbol{W}^T \boldsymbol{\Phi}_{\boldsymbol{\epsilon}})$, we define the matrix $\boldsymbol{\Phi}_{\boldsymbol{\epsilon}}$ as follows:

$$\boldsymbol{\Phi}_{\boldsymbol{\epsilon}} := \left[\sum_{i=1}^{n} \epsilon_{i1} \phi(\boldsymbol{x}_i), \sum_{i=1}^{n} \epsilon_{i2} \phi(\boldsymbol{x}_i), \cdots, \sum_{i=1}^{n} \epsilon_{iK} \phi(\boldsymbol{x}_i) \right].$$

Applying Hölder's inequation and $\|\boldsymbol{W}\|_*$ bounded by a constant B to (17), we can obtain

$$\begin{aligned} \widehat{\mathcal{R}}(\mathcal{H}) =\; & \frac{1}{n}\mathbb{E}_{\boldsymbol{\epsilon}} \left[\sup_{f \in \mathcal{H}} \langle \boldsymbol{W}, \boldsymbol{\Phi}_{\boldsymbol{\epsilon}} \rangle \right] \\ \le\; & \frac{1}{n}\mathbb{E}_{\boldsymbol{\epsilon}} \left[\sup_{f \in \mathcal{H}} \|\boldsymbol{W}\|_* \|\boldsymbol{\Phi}_{\boldsymbol{\epsilon}}\|_F \right] \le \frac{B}{n}\mathbb{E}_{\boldsymbol{\epsilon}} \left[\|\boldsymbol{\Phi}_{\boldsymbol{\epsilon}}\|_F \right] \\ \le\; & \frac{B}{n}\mathbb{E}_{\boldsymbol{\epsilon}} \left[\sqrt{\|\boldsymbol{\Phi}_{\boldsymbol{\epsilon}}\|_F^2} \right] \le \frac{B}{n} \sqrt{\mathbb{E}_{\boldsymbol{\epsilon}} \|\boldsymbol{\Phi}_{\boldsymbol{\epsilon}}\|_F^2}. \end{aligned} \tag{18}$$

Then, we bound $\mathbb{E}_{\boldsymbol{\epsilon}} \|\boldsymbol{\Phi}_{\boldsymbol{\epsilon}}\|_F^2$ as follows

$$\begin{aligned} \mathbb{E}_{\boldsymbol{\epsilon}} \|\boldsymbol{\Phi}_{\boldsymbol{\epsilon}}\|_F^2 \le\; & \mathbb{E}_{\boldsymbol{\epsilon}} \sum_{k=1}^{K} \Big\| \sum_{i=1}^{n} \epsilon_{ik} \phi(\boldsymbol{x}_i) \Big\|_2^2 \\ \le\; & \sum_{k=1}^{K} \mathbb{E}_{\boldsymbol{\epsilon}} \Big\| \sum_{i=1}^{n} \epsilon_{ik} \phi(\boldsymbol{x}_i) \Big\|_2^2 \\ \le\; & \sum_{k=1}^{K} \mathbb{E}_{\boldsymbol{\epsilon}} \sum_{i,k=1}^{n} \epsilon_{ik} \epsilon_{jk} \big[\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle \big] \\ =\; & K \sum_{i=1}^{n} \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_i) \rangle. \end{aligned} \tag{19}$$

The last step is due to the symmetry of $\langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_i) \rangle$ shown in (6). The result is similar to (Bartlett and Mendelson 2002; Cortes, Mohri, and Rostamizadeh 2013) Applying spectral representation in (5) of non-stationary kernels, we further bound the Rademacher complexity

$$\begin{aligned} \widehat{\mathcal{R}}(\mathcal{H}) \le\; & \frac{B}{n} \sqrt{K \sum_{i=1}^{n} \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_i) \rangle} \\ =\; & \frac{B}{n} \sqrt{\frac{K}{D} \sum_{i=1}^{n} \frac{1}{2} \Big[\cos\big((\boldsymbol{\Omega} - \boldsymbol{\Omega}')^T \boldsymbol{x}_i \big) + 1 \Big]} \end{aligned} \tag{20}$$

where $B = \sup_{f \in \mathcal{H}} \|\boldsymbol{W}\|_*$. Substituting the above inequation (20) to (16), we complete the proof. □

Singular Values Thresholding (SVT)

In each iteration, to obtain a tight surrogate of Equation (4), we keep $\|\boldsymbol{W}\|_*$ while relaxing empirical loss $g(\boldsymbol{W})$ only, that leads proximal gradient (Parikh, Boyd, and others 2014)

$$\begin{aligned} \boldsymbol{W}^{t+1} &= \arg\min_{\boldsymbol{W}} \lambda_1 \|\boldsymbol{W}\|_* + g(\boldsymbol{W}) \\ &= \arg\min_{\boldsymbol{W}} \lambda_1 \|\boldsymbol{W}\|_* + g(\boldsymbol{W}^t) \\ &+ \langle \nabla g(\boldsymbol{W}^t), \boldsymbol{W} - \boldsymbol{W}^t \rangle + \frac{1}{2\eta} \|\boldsymbol{W} - \boldsymbol{W}^t\|_F^2 \\ &= \arg\min_{\boldsymbol{W}} \lambda_1 \|\boldsymbol{W}\|_* + \frac{1}{2\eta} \|\boldsymbol{W} - (\boldsymbol{W}^t - \eta \nabla g(\boldsymbol{W}^t)\|_F^2 \\ &= \arg\min_{\boldsymbol{W}} \frac{1}{2} \|\boldsymbol{W} - \boldsymbol{Q}\|_F^2 + \eta \lambda_1 \|\boldsymbol{W}\|_* \end{aligned}$$

where $\boldsymbol{Q} = \boldsymbol{W}^t - \eta \nabla g(\boldsymbol{W}^t)$ and η is the learning rate.

Proposition 1 (Theorem 2.1 of (Cai, Candès, and Shen 2010))**.** *Let $\boldsymbol{Q} \in \mathbb{R}^{D \times K}$ with rank r and its singular values decomposition (SVD) is $\boldsymbol{Q} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$, where $\boldsymbol{U} \in \mathbb{R}^{d \times r}$ and $\boldsymbol{V} \in \mathbb{R}^{K \times r}$ have orthogonal columns, $\boldsymbol{\Sigma}$ is the diagonal diag$(\{\sigma_i\}_{1 \le i \le r})$. Then,*

$$\arg\min_{\boldsymbol{W}} \left\{ \frac{1}{2} \|\boldsymbol{W} - \boldsymbol{Q}\|_F^2 + \eta \|\boldsymbol{W}\|_* \right\} = \boldsymbol{U}\boldsymbol{\Sigma}_\eta \boldsymbol{V}^T,$$

where the diagonal is $\boldsymbol{\Sigma}_\eta = diag(\{\sigma_i - \eta\}_+)$.

Applying the SVT in Proposition 1 (Cai, Candès, and Shen 2010; Lu et al. 2015; Chatterjee and others 2015) can leads results in (8) to update $\boldsymbol{W}$ twice in each iteration.

References

Bartlett, P. L., and Mendelson, S. 2002. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research* 3(Nov):463–482.

Bartlett, P. L.; Bousquet, O.; Mendelson, S.; et al. 2005. Local rademacher complexities. *The Annals of Statistics* 33(4):1497–1537.

Bengio, Y.; Delalleau, O.; and Roux, N. L. 2006. The curse of highly variable functions for local kernel machines. In *Advances in neural information processing systems*, 107–114.

Cai, J.-F.; Candès, E. J.; and Shen, Z. 2010. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization* 20(4):1956–1982.

Chatterjee, S., et al. 2015. Matrix estimation by universal singular value thresholding. *The Annals of Statistics* 43(1):177–214.

Cortes, C.; Kuznetsov, V.; Mohri, M.; and Yang, S. 2016. Structured prediction theory based on factor graph complexity. In *Advances in Neural Information Processing Systems 29 (NIPS)*, 2514–2522.

Cortes, C.; Mohri, M.; and Rostamizadeh, A. 2013. Multi-class classification with maximum margin multiple kernel. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 46–54.

Huang, P.-S.; Avron, H.; Sainath, T. N.; Sindhwani, V.; and Ramabhadran, B. 2014. Kernel methods match deep neural networks on timit. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 205–209. IEEE.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Le, Q.; Sarlós, T.; and Smola, A. 2013. Fastfood-approximating kernel expansions in loglinear time. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, volume 85.

LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P.; et al. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324.

Li, J.; Liu, Y.; Yin, R.; and Wang, W. 2019. Multi-class learning using unlabeled samples : Theory and algorithm. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*.

Lu, C.; Zhu, C.; Xu, C.; Yan, S.; and Lin, Z. 2015. Generalized singular value thresholding. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI)*, 1805–1811.

Parikh, N.; Boyd, S.; et al. 2014. Proximal algorithms. *Foundations and Trends® in Optimization* 1(3):127–239.

Quiñonero-Candela, J.; Rasmussen, C. E.; Figueiras-Vidal, A. R.; et al. 2010. Sparse spectrum gaussian process regression. *Journal of Machine Learning Research* 11(Jun):1865–1881.

Rahimi, A., and Recht, B. 2007. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems 21 (NIPS)*, 1177–1184.

Remes, S.; Heinonen, M.; and Kaski, S. 2017. Nonstationary spectral kernels. In *Advances in Neural Information Processing Systems 30 (NIPS)*, 4642–4651.

Rudin, W. 1962. *Fourier analysis on groups*, volume 121967. Wiley Online Library.

Rumelhart, D. E.; Hinton, G. E.; Williams, R. J.; et al. 1988. Learning representations by back-propagating errors. *Cognitive modeling* 5(3):1.

Samo, Y.-L. K., and Roberts, S. 2015. Generalized spectral kernels. *arXiv preprint arXiv:1506.02236*.

Stein, M. L. 2012. *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media.

Sun, S.; Zhang, G.; Shi, J.; and Grosse, R. 2019. Functional variational bayesian neural networks. *arXiv preprint arXiv:1903.05779*.

Ton, J.-F.; Flaxman, S.; Sejdinovic, D.; and Bhatt, S. 2018. Spatial mapping with gaussian processes and nonstationary fourier features. *Spatial statistics* 28:59–78.

Wilson, A., and Adams, R. 2013. Gaussian process kernels for pattern discovery and extrapolation. In *International Conference on Machine Learning*, 1067–1075.

Xu, C.; Liu, T.; Tao, D.; and Xu, C. 2016. Local rademacher complexity for multi-label learning. *IEEE Transactions on Image Processing* 25(3):1495–1507.

Yaglom, A. M. 1987. Correlation theory of stationary and related random functions. *Volume I: Basic Results.* 526.

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 (ICML)*, 485–493.

Yu, H.-F.; Jain, P.; Kar, P.; and Dhillon, I. 2014. Large-scale multi-label learning with missing labels. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 593–601.

Yu, F. X. X.; Suresh, A. T.; Choromanski, K. M.; Holtmann-Rice, D. N.; and Kumar, S. 2016. Orthogonal random features. In *Advances in Neural Information Processing Systems 29 (NIPS)*, 1975–1983.

Zeiler, M. D. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.