Copula-Nested Spectral Kernel Network

Jinyue Tian [1 2] **Hui Xue** [1 2] **Yanfang Xue** [1 2] **Pengfei Fang** [1 2]

Abstract

Spectral Kernel Networks (SKNs) emerge as a promising approach in machine learning, melding solid theoretical foundations of spectral kernels with the representation power of hierarchical architectures. At its core, the spectral density function plays a pivotal role by revealing essential patterns in data distributions, thereby offering deep insights into the underlying framework in real-world tasks. Nevertheless, prevailing designs of spectral density often overlook the intricate interactions within data structures. This phenomenon consequently neglects expanses of the hypothesis space, thus curtailing the performance of SKNs. This paper addresses the issues through a novel approach, the **Co**pula-Nested Spectral **Ke**rnel **Net**work (**CokeNet**). Concretely, we first redefine the spectral density with the form of copulas to enhance the diversity of spectral densities. Next, the specific expression of the copula module is designed to allow the excavation of complex dependence structures. Finally, the unified kernel network is proposed by integrating the corresponding spectral kernel and the copula module. Through rigorous theoretical analysis and experimental verification, CokeNet demonstrates superior performance and significant advancements over SOTA algorithms in the field.

1. Introduction

Kernel methods are an essential class of machine learning approaches (Shawe-Taylor & Cristianini, 2004). However, with the ever-growing scale of datasets in the machine learning community, the computation cost of kernel methods significantly increases. To overcome these limitations, researchers have proposed several approaches to improve the scalability of traditional kernels (Williams & Seeger, 2000; Rahimi & Recht, 2007; Yang et al., 2014; Liu et al., 2021). For instance, Random Fourier Features (RFF) (Rahimi & Recht, 2007), as a widely adopted method, proposed an explicit kernel mapping based on the spectral representation of stationary kernels, which inspires many researchers to delve deeper into the exploration of spectral kernels (Lázaro-Gredilla et al., 2010; Wilson & Adams, 2013; Sinha & Duchi, 2016). To further enhance the representation ability of spectral kernels, some integrated them into the hierarchical architectures, resulting in spectral kernel networks (SKNs) (Xue et al., 2019; Li et al., 2022; Xu et al., 2022), which share the benefits of reduced computation cost along with the improved capability of representation while maintaining solid theoretical foundations.

In the SKNs, the spectral density serves as a crucial determinant of their properties. It is refined through the back-propagation procedure of the SKN, encapsulating the intrinsic characteristics in the data. However, the selection and formulation of spectral density remain underexplored. Previous works prefer to utilize the spectral density function of traditional kernels (Rahimi & Recht, 2007; Xue et al., 2019) or assume the spectral density to be a linear combination of Gaussian probability density functions (Wilson & Adams, 2013; Samo & Roberts, 2015; Remes et al., 2017). The former is more akin to approximating an existing kernel, rather than creating a novel data-based kernel. The latter fails to investigate complex architectures between variables, such as the dependence structures. Both of these methods limit the exploration of the hypothesis space. To adjust the spectral densities, Avron *et al.* related the spectral density to an appropriately defined ridge leverage function (Avron et al., 2017), but the modified spectral density can only be used in the 1-dimensional Gaussian kernel. Li *et al.* employed the empirical ridge leverage score distribution and proposed an algorithm to approximate the distribution and the leverage weights (Li et al., 2019). However, the distribution is not learnable and hence unable to dynamically mine the structure in the data. Xue and Wu derived a prior-posterior bridge to strengthen the uncertainty of spectral density (Xue & Wu, 2020), but the bridge is manually chosen, reducing the flexibility in the parametric form of the spectral density. The insufficient exploration of spectral den-

[1]School of Computer Science and Engineering, Southeast University, Nanjing, 210096, China [2]Key Laboratory of New Generation Artificial Intelligence Technology and Its Interdisciplinary Applications (Southeast University), Ministry of Education, China. Correspondence to: Hui Xue <hxue@seu.edu.cn>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

sity prevents SKNs from fully demonstrating their inherent, powerful data mining capabilities.

In this paper, we propose an expressive and flexible method named **Co**pula-Nested Spectral **Ke**rnel **Net**work (CokeNet). The core idea is to introduce copula networks into the design of the spectral density based on Sklar's theorem. This scheme can investigate a broader range of hypothesis space of SKNs and capture the complicated relations between variables in the data. Concretely, we first redefine the spectral density as a multiplication of copulas and pseudo probability density by employing Sklar's theorem. Secondly, we construct the copula module. Particularly, we select the Archimedean copulas due to their simplicity and ability to model complex distributions. Berstein's theorem is then utilized to further enrich the choice of generators in Archimedean copulas, which leads to a copula module in hierarchical architectures. Finally, we insert the redefined spectral density and the copula module into the spectral kernel mapping, resulting in CokeNet. With the copula and the kernel both in the representation of networks, we can train the network in an end-to-end manner and flexibly fit the network to the data, learning the spectral density most suitable for the given task. Our contributions are:

- We propose a learnable copula-nested spectral kernel network, CokeNet. By introducing the copula networks into the construction of spectral kernels, the proposed model can explore wider expanses of hypothesis spaces and capture the complicated relations in the data, thus deriving the most task-appropriate spectral density.
- We theoretically analyze the improvements of CokeNet. This includes a broadening in the variety of spectral densities, an enhanced capability for data structure extraction and an augmented generalization ability.
- We conduct experiments on synthetic datasets and real-world datasets to evaluate the effectiveness of our method. The experimental results on the synthetic data show that CokeNet can capture the dependence structure between data variables. Results of the real-world experiments demonstrate the superiority of CokeNet compared to state-of-the-art relevant methods.

2. Related Work

Spectral Kernel Based on Bochner's theorem, Rahimi and Recht proposed the RFF method based on the spectral representation of stationary kernels (Rahimi & Recht, 2007). Its core idea is generating an explicit expression of the kernel mapping to approximate the implicit mapping of the target kernel, resulting in a succinct expression of cosine functions. This work motivates many researchers to further explore spectral kernels (Wilson & Adams, 2013; Sinha & Duchi, 2016; Lázaro-Gredilla et al., 2010). Some researchers generalized spectral kernels to non-stationary scenarios. Remes *et al*. define spectral density as a combination of bivariate Gaussian components and present a family of non-stationary and non-monotonic kernels, which can learn input-dependent and long-range covariance between inputs (Remes et al., 2017). Ton *et al*. obtain the sparse spectrum kernel by solving a more general spectral characterization of non-stationary kernels (Ton et al., 2018). Samo and Roberts leverage Wiener's Tauberian theorem and Yaglom's theorem to construct families of kernels that can approximate arbitrarily well any bounded continuous non-stationary kernels (Samo & Roberts, 2015). To further enhance the representation ability of spectral kernels, ones combine spectral kernels and various deep architectures, resulting in SKNs (Xue et al., 2019; Li et al., 2022; Xu et al., 2022). These methods have not only strong mathematical guarantees of kernel methods but also the power to model sophisticated data patterns brought by deep learning. Furthermore, Xue *et al*. generalize the SKNs on the real number domain to the complex number domain by taking both the real and imaginary parts of the spectral kernel mapping into account (Xue et al., 2023). Now, SKNs are not merely a plug-in within kernel methods. Their improved integrability and diverse architectures enable them to be used as standalone learners.

Copula Theory Copula is a powerful statistical tool for capturing the dependence structure between random variables, introduced by Sklar (Sklar, 1959). Along with the introduction of the notion of copula, Sklar's theorem points out that any d-dimensional continuous joint distribution is a product of d marginal distribution functions and a single d-dimensional copula, which reduces the complex issue of estimating the multivariate distribution to a much easier problem of estimating the univariate distribution and their dependence structure. Hence, various copulas have been developed and widely applied in fields where dependence matters, such as economics (Oh & Patton, 2018), biology (Disegna et al., 2017), medicine (Genest & Rivest, 1993) *et al*. However, the diversity of copula functions brings the challenge of selecting or estimating a well-suited and tractable copula. To tackle this issue, many copula generative models have been proposed (Ling et al., 2020; Ng et al., 2021; Janke et al., 2021). These methods enhance the ability to tailor copula functions to specific datasets, enabling more precise modeling of complex dependencies.

3. Preliminary

In this section, we introduce some necessary concepts about spectral kernels and copulas. Throughout the paper, the vectors are denoted by bold letters (*e.g.* $\boldsymbol{\omega}$) while the scalars are not (*e.g.* i).

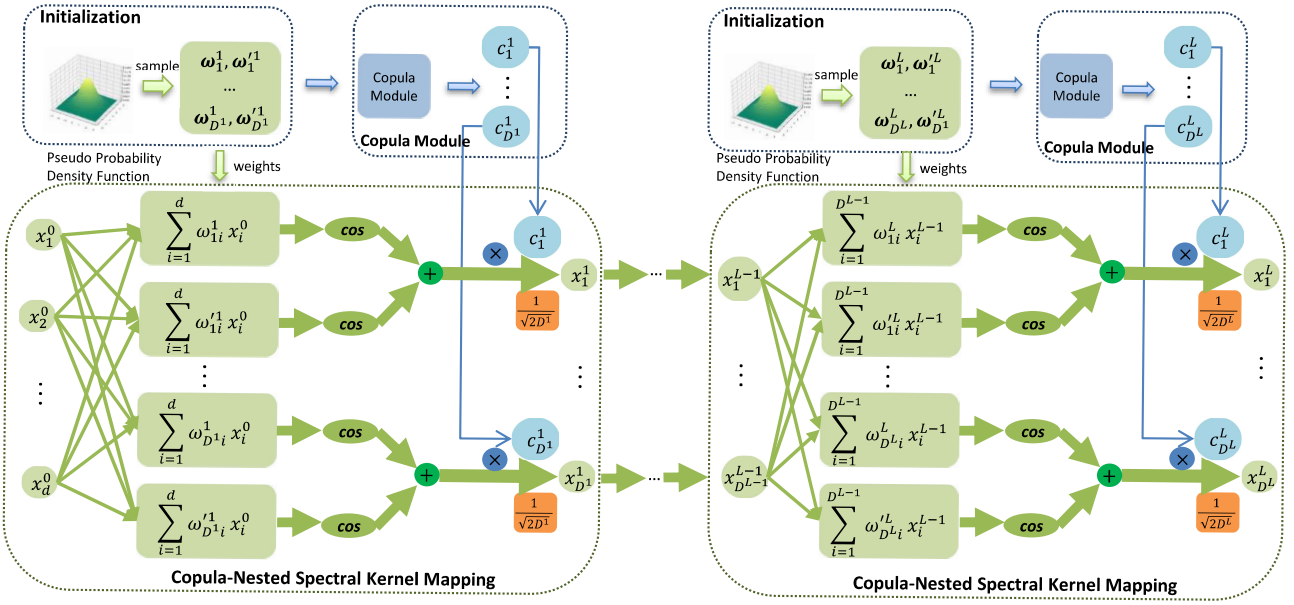


Figure 1. The overall structure of the CokeNet. First, we initial the weights according to the pseudo probability density function and pass them to the Copula Module. Then, the copulas are passed into the spectral kernel mapping. Finally, the copula-nested spectral kernel mapping makes each layer of CokeNet.

3.1. Spectral Kernel

According to Yaglom's theorem (Yaglom, 1987), a non-stationary kernel $k(\boldsymbol{x}, \boldsymbol{x}')$ is positive definite in $\mathbb{R}^d$ if and only if it has the form

$$k(\boldsymbol{x}, \boldsymbol{x}') = \int_{\mathbb{R}^d \times \mathbb{R}^d} e^{i(\boldsymbol{\omega}^\top \boldsymbol{x} - \boldsymbol{\omega}'^\top \boldsymbol{x}')} \mu(d\boldsymbol{\omega}, d\boldsymbol{\omega}'), \quad (1)$$

where $\mu(d\boldsymbol{\omega}, d\boldsymbol{\omega}')$ is the Lebesgue-Stieltjes measure associated to some positive definite function $p(\boldsymbol{\omega}, \boldsymbol{\omega}')$ with bounded variation. And they satisfy that $\mu(d\boldsymbol{\omega}, d\boldsymbol{\omega}') = p(\boldsymbol{\omega}, \boldsymbol{\omega}')d\boldsymbol{\omega}d\boldsymbol{\omega}'$. Hence, $\frac{p(\boldsymbol{\omega}, \boldsymbol{\omega}')}{\int p(\boldsymbol{\omega}, \boldsymbol{\omega}) d\boldsymbol{\omega} d\boldsymbol{\omega}'}$ is a probability density function. Without loss of generality, we can assume that $\int p(\boldsymbol{\omega}, \boldsymbol{\omega}) d\boldsymbol{\omega} d\boldsymbol{\omega}' < \infty$ and $p(\boldsymbol{\omega}, \boldsymbol{\omega}')$ is a probability density function. There exists a one-to-one correspondence between k and p based on Fourier duality.

$$\begin{aligned} k(\boldsymbol{x}, \boldsymbol{x}') &= \int_{\mathbb{R}^d \times \mathbb{R}^d} e^{i(\boldsymbol{\omega}^\top \boldsymbol{x} - \boldsymbol{\omega}'^\top \boldsymbol{x}')} p(\boldsymbol{\omega}, \boldsymbol{\omega}') d\boldsymbol{\omega} d\boldsymbol{\omega}', \\ p(\boldsymbol{\omega}, \boldsymbol{\omega}') &= \int_{\mathbb{R}^d \times \mathbb{R}^d} e^{-i(\boldsymbol{\omega}^\top \boldsymbol{x} - \boldsymbol{\omega}'^\top \boldsymbol{x}')} k(\boldsymbol{x}, \boldsymbol{x}') d\boldsymbol{x} d\boldsymbol{x}', \end{aligned} \quad (2)$$

where p is the spectral density of the kernel and $\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{\omega}, \boldsymbol{\omega}' \in \mathbb{R}^d$. Thereby, k is also called the spectral kernel.

3.2. Copula

Copulas are powerful mathematical tools that fully depict the dependence structure among random variables, offering great flexibility in building multivariate models. They are defined as

Definition 3.1. A function $C : [0, 1]^d \to [0, 1]$ is a copula if and only if the following properties hold

(i.) for every $j \in \{1, \ldots, d\}$, $C(\boldsymbol{u}) = u_j$ when all elements of $\boldsymbol{u}$ are equal to 1 with the exception of the j-th one that is equal to $u_j \in [0, 1]$;

(ii.) $C(\boldsymbol{u}) \leq C(\boldsymbol{v})$ for all $\boldsymbol{u}, \boldsymbol{v} \in [0, 1]^d, \boldsymbol{u} \leq \boldsymbol{v}$;

(iii.) C is d-increasing.

These properties guarantee copulas to be a special kind of cumulative distribution function. Their corresponding probability density function, namely the copula density function is defined as

$$c(u_1, \ldots, u_d) = \frac{\partial C(u_1, \ldots, u_d)}{\partial u_1 \ldots \partial u_d}. \quad (3)$$

4. CokeNet

In this section, we first provide the overall architecture of our CokeNet of three parts. Subsequently, we introduce each part in detail.

Overall Architecture CokeNet comprises three parts: the redefinition of spectral density, the construction of copula modules, and the copula-nested spectral kernel mapping.

The overall architecture is shown in Figure 1. Concretely, the spectral density is defined as

$$p(\boldsymbol{\omega}, \boldsymbol{\omega}') = c(\boldsymbol{\omega}, \boldsymbol{\omega}')\hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega}'), \tag{4}$$

where c is the integrated copula module and $\hat{p}$ represents the pseudo sampling distribution for initialization. Then, we obtain the copula module by representing a family of Archimedean copulas C in a network form. Combining the novel spectral kernel density $p(\boldsymbol{\omega}, \boldsymbol{\omega}')$ and copula module, we derive the spectral kernel mapping Φ as one layer of CokeNet. Stacking L layers of spectral kernel mapping, our CokeNet is formulated as

$$\text{CokeNet}(x) = \Phi^L(\ldots\Phi^2(\Phi^1(\boldsymbol{x}))), \tag{5}$$

where Φ^l is the copula-nested spectral kernel mapping in l-th layer.

4.1. Copula-Nested Spectral Density

Theorem 4.1. *(Sklar's Theorem) (Jaworski et al., 2010) Let F be a d-dimensional distribution function with univariate margins $F_1, F_2, \ldots, F_d$. Let A_j denote the range of F_j, $A_j = F_j(\mathbb{R})(j = 1, 2 \ldots, d)$. Then there exists a copula C such that for all $(x_1, x_2, \ldots, x_d) \in \mathbb{R}^d$,*

$$F(x_1, x_2, \ldots, x_d) = C(F_1(x_1), F_2(x_2), \ldots, F_d(x_d)), \tag{6}$$

where C is uniquely determined on $A_1 \times A_2 \times \ldots \times A_d$. Hence, it is unique when $F_1, F_2, \ldots, F_d$ are all continuous.

By Sklar's theorem, copulas serve as a link to build a joint distribution with marginals. They depict the dependence structure between margins. Furthermore, Corollary 4.2 explains how the copula density function connects probability density functions.

Corollary 4.2. *(Jaworski et al., 2010) Follow the notation in Theorem 4.1. Let f be a d-dimensional probability density function with univariate margins $f_1, f_2, \ldots, f_d$. Let A_j denote the range of f_j, $A_j = f_j(\mathbb{R})(j = 1, 2, \ldots, d)$. Then there exists a copula C such that for all $(x_1, \ldots, x_d) \in \mathbb{R}^d$,*

$$f(x_1, x_2, \ldots, x_d) = c(F_1(x_1), \ldots, F_d(x_d)) \prod_{k=1}^{d} f_k(x_k),$$
$$c(F_1(x_1), \ldots, F_d(x_d)) = \frac{\partial C(F_1(x_1), \ldots, F_d(x_d))}{\partial F_1(x_1) \ldots \partial F_d(x_d)}, \tag{7}$$

where c is the copula density function uniquely determined on $A_1 \times A_2 \times \ldots \times A_d$. It satisfies that c is unique when $f_1, f_2, \ldots, f_d$ are all continuous.

Look back at Equation (2), by Corollary 4.2, we formulate the spectral density function p as

$$\begin{aligned} p(\boldsymbol{\omega}, \boldsymbol{\omega}') &= c(\boldsymbol{\omega}, \boldsymbol{\omega}')\hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega}') \\ &= c(P_1(\omega_1) \ldots, P'_d(\omega'_d)) \prod_{j=1}^{d} p_j(\omega_j) p'_j(\omega'_j), \end{aligned} \tag{8}$$

where ω_j, ω'_j is the j-th element of the vector $\boldsymbol{\omega}, \boldsymbol{\omega}' \in \mathbb{R}^d$ respectively. p_j and p'_j are the corresponding probability density functions. P_j and P'_j are the corresponding cumulative distribution functions. c is the copula density and $\hat{p}$ is the pseudo probability density function. Note that when $c(\boldsymbol{\omega}, \boldsymbol{\omega}') \equiv 1$, the spectral density degenerates to the scenario where each feature is independent of each other.

4.2. Copula Module

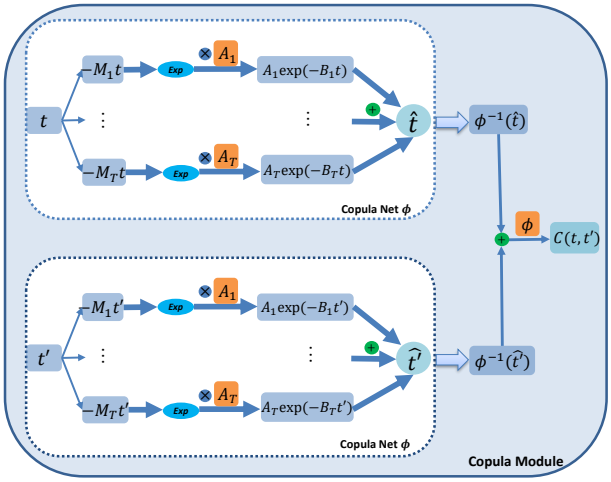


Figure 2. The structure of copula module.

Motivated by ACNet (Ling et al., 2020), we select the family of Archimedean copulas to link the marginal probability density functions to a joint probability density function due to its simple form and ability to model complex distributions. The Archimedean copulas are given by

$$C(u_1, \ldots, u_d) = \phi(\phi^{-1}(u_1) + \phi^{-1}(u_2) + \ldots + \phi^{-1}(u_d)), \tag{9}$$

where ϕ is the generator of C. And $\phi : [0, \infty) \to [0, 1]$ is d-monotone, *i.e.* $(-1)^k \phi^{(k)}(t) \geq 0$ for all $k \leq d$, $t \geq 0$.

From Equation (9), it can be seen that the generator ϕ determines the characteristics of Archimedean copulas. Hence, an important problem is the selection of ϕ since there are so many functions that satisfy the d-monotone constraint. If we strengthen the condition in Equation (9) to require that the generator is totally monotone, *i.e.* $(-1)^k \phi^{(k)}(t) \geq 0$ for all nonnegative integers k and all $t \geq 0$. These functions can be expressed in a unified form via Theorem 4.3.

Theorem 4.3. *(Berstein's theorem) (Bernstein, 1929) A function ϕ is totally monotone if and only if ϕ is the Laplace transform of a positive random variable M, i.e. $\phi(t) = \int_0^\infty e^{-Mt}p(M)dM$ and $P(M > 0) = 1$.*

Hence, by Berstein's theorem and Monte Carlo approximation, the generator ϕ can be modified as

$$\begin{aligned}\phi(t) &= \int_0^\infty e^{-Mt}p(M)dM = E_{M\sim P(M)}[e^{-Mt}] \\ &\approx \frac{1}{T}\sum_{k=1}^{T} exp(-M_k t),\end{aligned} \tag{10}$$

where T is the number of samplings and $M_k > 0$. Since the positive linear combination of monotone functions remains monotone, we can derive a more general form of Equation (10)

$$\phi(t) = \sum_{k=1}^{T} A_k exp(-M_k t), \tag{11}$$

where $A_k \geq 0$. It can be seen as a one-layer neural network with exponential function as its activation function. We name the network as the Copula Net. Note that ϕ can be extended to deep architecture, but that is not the focus of this paper. For more information, please refer to (Ling et al., 2020).

Subsequently, by Equation (9), we generate copulas with respect to the variables $(\boldsymbol{\omega}, \boldsymbol{\omega}') \in \mathbb{R}^{2d}$

$$C(\boldsymbol{\omega}, \boldsymbol{\omega}') = \sum_{k=1}^{T} A_k exp(-M_k(\sum_{j=1}^{d}[\phi^{-1}(\omega_j) + \phi^{-1}(\omega_j')])), \tag{12}$$

where ω_j, ω_j is the j-th element of $\boldsymbol{\omega}, \boldsymbol{\omega'}$ respectively. ϕ is defined in Equation (11). ϕ^{-1} is its inverse, which can be calculated by numerical methods such as Newton's method and bisection method, *i.e.* by solving for t in $\phi(t) - \omega_i = 0$. We present a simple case in Figure 2, where $t, t' \in \mathbb{R}$ are scalars, $\hat{t} = \phi(t)$. All steps in the copula module theoretically guarantee that Equation (12) is an efficient approximation of copulas. Further, the derivatives of the copula, namely the copula density function c, can be obtained by automatic differentiation libraries such as PyTorch (Paszke et al., 2019).

The copula is structured within the framework of a network architecture instead of a conventional statistical format, offering two significant advantages. Firstly, there is no need to choose the generator manually. According to Theorem 4.3, the exact parametric form of the generator relies on the optimization of the approximation (Equation (10)) and the selection of hyper-parameters in the copula network. This significantly improves the flexibility in the architecture of the generator, consequently enhancing the agility of the copulas. Secondly, using the expression of networks makes the copula more task-oriented. Through gradient descent, CokeNet optimizes the parameters in the copula module so that the consequent output is a reliable prediction. Therefore, the copulas are formed in a most task-oriented way.

4.3. Copula-Nested Spectral Kernel Mapping

Using the novel spectral density in Equation (8) and our elaborate copula density c, the spectral representation in Equation (2) can be written as

$$k(\boldsymbol{x}, \boldsymbol{x}') = \int_{\mathbb{R}^d\times\mathbb{R}^d} e^{i(\boldsymbol{\omega}^\top \boldsymbol{x} - \boldsymbol{\omega}'^\top \boldsymbol{x}')} c(\boldsymbol{\omega}, \boldsymbol{\omega}')\hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega}')d\boldsymbol{\omega}d\boldsymbol{\omega}'. \tag{13}$$

To loosen the symmetry restriction of spectral kernel k, we rewrite Equation (13) as

$$k(\boldsymbol{x}, \boldsymbol{x}') = \int_{\mathbb{R}^d\times\mathbb{R}^d} \tau(\boldsymbol{\omega}, \boldsymbol{\omega}', \boldsymbol{x}, \boldsymbol{x}')c(\boldsymbol{\omega}, \boldsymbol{\omega}')\hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega}')d\boldsymbol{\omega}d\boldsymbol{\omega}', \tag{14}$$

where $\tau(\boldsymbol{\omega}, \boldsymbol{\omega}', \boldsymbol{x}, \boldsymbol{x}')$ is defined as

$$\begin{aligned}&\frac{1}{8}[e^{i(\boldsymbol{\omega}^\top \boldsymbol{x} - \boldsymbol{\omega}'^\top \boldsymbol{x}')} + e^{i(\boldsymbol{\omega}'^\top \boldsymbol{x} - \boldsymbol{\omega}^\top \boldsymbol{x}')} \\ &+ e^{i(-\boldsymbol{\omega}^\top \boldsymbol{x} + \boldsymbol{\omega}'^\top \boldsymbol{x}')} + e^{i(-\boldsymbol{\omega}'^\top \boldsymbol{x} + \boldsymbol{\omega}^\top \boldsymbol{x}')} \\ &+ e^{i(\boldsymbol{\omega}^\top \boldsymbol{x} - \boldsymbol{\omega}^\top \boldsymbol{x}')} + e^{i(\boldsymbol{\omega}'^\top \boldsymbol{x} - \boldsymbol{\omega}'^\top \boldsymbol{x}')} \\ &+ e^{i(-\boldsymbol{\omega}^\top \boldsymbol{x} + \boldsymbol{\omega}^\top \boldsymbol{x}')} + e^{i(-\boldsymbol{\omega}'^\top \boldsymbol{x} + \boldsymbol{\omega}'^\top \boldsymbol{x}')}].\end{aligned} \tag{15}$$

Applying Euler's formula that $e^{i\boldsymbol{x}} = \cos(\boldsymbol{x}) + i\sin(\boldsymbol{x})$, we rewrite $\tau(\boldsymbol{\omega}, \boldsymbol{\omega}', \boldsymbol{x}, \boldsymbol{x}')$ as

$$\begin{aligned}&\frac{1}{4}[\cos(\boldsymbol{\omega}^\top \boldsymbol{x} - \boldsymbol{\omega}'^\top \boldsymbol{x}') + \cos(\boldsymbol{\omega}'^\top \boldsymbol{x} - \boldsymbol{\omega}^\top \boldsymbol{x}') \\ &+ \cos(\boldsymbol{\omega}^\top \boldsymbol{x} - \boldsymbol{\omega}^\top \boldsymbol{x}') + \cos(\boldsymbol{\omega}'^\top \boldsymbol{x} - \boldsymbol{\omega}'^\top \boldsymbol{x}')].\end{aligned} \tag{16}$$

We deem that $(\boldsymbol{\omega}, \boldsymbol{\omega}')$ follow the pseudo probability density function $\hat{p}$. Following Monte Carlo approximation and Equation (38), we have

$$\begin{aligned}k(\boldsymbol{x}, \boldsymbol{x}') &= \mathbb{E}_{(\boldsymbol{\omega},\boldsymbol{\omega}')\sim\hat{P}}[\tau(\boldsymbol{\omega}, \boldsymbol{\omega}', \boldsymbol{x}, \boldsymbol{x}')c(\boldsymbol{\omega}, \boldsymbol{\omega}')] \\ &\approx \frac{1}{D}\sum_{m=1}^{D} \tau(\boldsymbol{\omega}_m, \boldsymbol{\omega}_m', \boldsymbol{x}, \boldsymbol{x}')c(\boldsymbol{\omega}_m, \boldsymbol{\omega}_m') \\ &= \Psi(\boldsymbol{x})^\top \Psi(\boldsymbol{x}'),\end{aligned} \tag{17}$$

where

$$\Psi(\boldsymbol{x}) = \frac{1}{\sqrt{4D}}\begin{bmatrix}\cos(\boldsymbol{\omega}_1^\top \boldsymbol{x}) + \cos(\boldsymbol{\omega}_1'^\top \boldsymbol{x}) \\ \cdots \\ \cos(\boldsymbol{\omega}_D^\top \boldsymbol{x}) + \cos(\boldsymbol{\omega}_D'^\top \boldsymbol{x}) \\ \sin(\boldsymbol{\omega}_1^\top \boldsymbol{x}) + \sin(\boldsymbol{\omega}_1'^\top \boldsymbol{x}) \\ \cdots \\ \sin(\boldsymbol{\omega}_D^\top \boldsymbol{x}) + \sin(\boldsymbol{\omega}_D'^\top \boldsymbol{x})\end{bmatrix} \cdot \begin{bmatrix}c(\boldsymbol{\omega}_1, \boldsymbol{\omega}_1') \\ \cdots \\ c(\boldsymbol{\omega}_D, \boldsymbol{\omega}_D') \\ c(\boldsymbol{\omega}_1, \boldsymbol{\omega}_1') \\ \cdots \\ c(\boldsymbol{\omega}_D, \boldsymbol{\omega}_D')\end{bmatrix}. \tag{18}$$

D is the number of random features, and $(\boldsymbol{\omega}_i, \boldsymbol{\omega}_i') \sim \hat{P}$ are the weights sampled in the i-th random feature.

Further, we replace Ψ with Φ, given by

$$\Phi(\boldsymbol{x}) = \frac{1}{\sqrt{2D}} \begin{bmatrix} \cos(\boldsymbol{\omega}_1^\top \boldsymbol{x}) + \cos(\boldsymbol{\omega}_1'^\top \boldsymbol{x}) \\ \cdots \\ \cos(\boldsymbol{\omega}_D^\top \boldsymbol{x}) + \cos(\boldsymbol{\omega}_D'^\top \boldsymbol{x}) \end{bmatrix} \cdot \begin{bmatrix} c(\boldsymbol{\omega}_1, \boldsymbol{\omega}_1') \\ \cdots \\ c(\boldsymbol{\omega}_D, \boldsymbol{\omega}_D') \end{bmatrix}, \tag{19}$$

since $\mathbb{E}[\Phi^\top(\boldsymbol{x})\Phi(\boldsymbol{x}')] = \mathbb{E}[\Psi^\top(\boldsymbol{x})\Psi(\boldsymbol{x}')]$. The explicit mapping defined by Equation (19) enables the kernel to be regarded as a neural network with cosine function as its activation function. Its parameters can be updated via algorithms such as gradient descent. Furthermore, the CokeNet is formulated as

$$\text{CokeNet}(\boldsymbol{x}) = \Phi^L(\Phi^{L-1}(\ldots \Phi^2(\Phi^1(\boldsymbol{x}))), \tag{20}$$

where Φ^l identifies the mapping of the l-th layer. The corresponding **Co**pula-nested spectral **ke**rnel (**Coke**) is

$$\hat{k}(\boldsymbol{x}, \boldsymbol{x}') = \langle \Phi^L(\ldots(\Phi^1(\boldsymbol{x})), \Phi^L(\ldots(\Phi^1(\boldsymbol{x}'))\rangle. \tag{21}$$

5. Analysis

By incorporating copulas into SKNs, our proposed model has great improvements in its representation and flexibility. The explicit description of dependence between variables enhances the diversity of the spectral densities which makes it more convenient for users to add prior information about the data. Further, integrating copulas strengthens the generalization of the model. We discuss the advantages in detail.

5.1. The Diversity and Uncertainty of Spectral Densities

We theoretically explain the capacity of CokeNet to increase the uncertainty of spectral densities via 1) the entropy of the weights and 2) the Wigner distribution of the copula-nested spectral kernel. Furthermore, we illustrate that this improvement makes CokeNet fall into local minima less easily compared to plain SKN.

1) The entropy of the spectral density function. Equation (19) indicates that the spectral density, including the pseudo probability density function $\hat{p}$ as well as the copula module, determines the parameter space of CokeNet, $\boldsymbol{\Theta}_{\text{CokeNet}} = \{(\boldsymbol{\Theta}_c, \boldsymbol{\omega}_i, \boldsymbol{\omega}_i') | (\boldsymbol{\omega}_i, \boldsymbol{\omega}_i') \sim \hat{P}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_i' \in \mathbb{R}^d, \boldsymbol{\Theta}_c \text{ are parameters in copula module}\}$. Similarly, the parameter space of the SKN without copulas is $\boldsymbol{\Theta}_{\text{SKN}} = \{(\boldsymbol{\omega}_i, \boldsymbol{\omega}_i') | (\boldsymbol{\omega}_i, \boldsymbol{\omega}_i') \sim P, \boldsymbol{\omega}_i, \boldsymbol{\omega}_i' \in \mathbb{R}^d\}$. Therefore, the design of spectral densities strongly influences the performance of SKNs.

In previous works, the spectral density p is either the one-to-one corresponding spectral density (defined in Equation (2)) of traditional kernels or a mixture of Gaussian components, leading to the following disadvantages: firstly, using spectral densities of existing kernels is similar to kernel approximation rather than kernel designing. Secondly, empirically choosing Gaussian components limits the selection of spectral kernels. In practice, many functions satisfying the condition described in Equation (2) are not taken into scope during the construction of spectral densities, which leaves a large part of the hypothesis space unexplored. Furthermore, the preconceived assumption may not be most task-appropriate and hinder the optimization process. The dependence structure of the weights is neglected under this assumption.

Therefore, we consider incorporating copulas in the construction of spectral densities, which improves the diversity and uncertainty of spectral densities and depicts dependence relations. We demonstrate this improvement via the entropy of the weights $(\boldsymbol{\omega}, \boldsymbol{\omega'})$. By Equation (8), we know that $(\boldsymbol{\omega}, \boldsymbol{\omega'}) \sim p(\boldsymbol{\omega}, \boldsymbol{\omega'}) = c(\boldsymbol{\omega}, \boldsymbol{\omega'})\hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega'})$, so the entropy is defined and formulated as

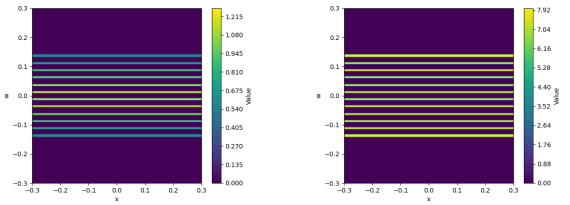
$$\begin{aligned} \underset{(\boldsymbol{\omega},\boldsymbol{\omega'})\sim P}{H}(\boldsymbol{\omega}, \boldsymbol{\omega'}) &= -\int p(\boldsymbol{\omega}, \boldsymbol{\omega'}) log(p(\boldsymbol{\omega}, \boldsymbol{\omega'})) d\boldsymbol{\omega} d\boldsymbol{\omega'} \\ &= \underset{(\boldsymbol{\omega},\boldsymbol{\omega'})\sim C}{H}(\boldsymbol{\omega}, \boldsymbol{\omega'}) + \underset{(\boldsymbol{\omega},\boldsymbol{\omega'})\sim \hat{P}}{H}(\boldsymbol{\omega}, \boldsymbol{\omega'}). \end{aligned} \tag{22}$$

It is trivial that the entropy of the weights is determined by the marginal distribution of each weight and the copula function. When the copula functions vary dynamically according to the weights, they increase the entropy of the weights, which indicates the enhancement of the diversity and uncertainty in spectral densities. Hence, by involving copulas in the construction of spectral densities, sampled weights in CokeNet contain more information, and the parameter space is enriched.

2) Wigner distribution of the copula-nested spectral kernel. Wigner transform (Flandrin, 1998) is a mathematical tool in time-frequency analysis, demonstrating the relation between input and frequency. The Wigner distribution function of a kernel k is defined as $W_k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$

$$W_k(\boldsymbol{x}, \boldsymbol{\omega}) = \int_{\mathbb{R}^d} k(\boldsymbol{x} + \frac{\boldsymbol{\tau}}{2}, x - \frac{\boldsymbol{\tau}}{2}) e^{-2i\pi \boldsymbol{\omega}^\top \boldsymbol{\tau}} d\boldsymbol{\tau}. \tag{23}$$

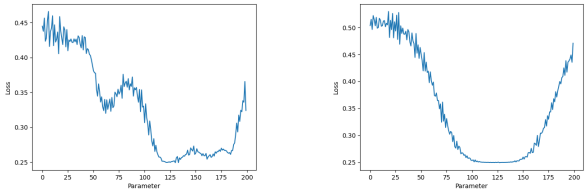
Note that when the kernel is stationary, the Wigner distribution function reduces to the spectral density. It reveals the frequency structure of the kernel and provides insight into the properties of the spectral density. We demonstrate the Wigner distribution of Coke and plain spectral kernel in Figure 3 and leave the detailed formula derivation and explanation in the appendix. In brief, both Coke and plain spectral kernel imply several sinusoidal signals. But given a fixed input x, the Wigner distribution of Coke exhibits a clear value change with frequency w (denoted in the color change in the figure), yet this variation is not apparent on the plain spectral kernel. This indicates that the copula-nested spectral density exhibits higher diversity.



(a) Coke (b) Plain Spectral Kernel

Figure 3. The Wigner distribution of Coke(a) and SK(b).

3) CokeNet falls into local minima less easily compared to plain SKN. A wider parameter space as well as the hypothesis space enables the model to explore deeper. It can prevent the model from local minima. Figure 4 demonstrates the changes of loss surface according to parameters in an experiment described in Section 6.1. The loss surface of the SKN is rougher and fluctuates more with multiple minima. While there is only one minimum in the loss surface of CokeNet. It indicates that the proposed model is less likely to fall into the local minima during optimization.



(a) SKN (b) CokeNet

Figure 4. The loss surface of the SKN (a) and CokeNet (b).

5.2. Description of Dependence Structures

Equation (22) shows that copulas bring extra information into the spectral densities. By further analyzing the increase in entropy brought by copulas, we obtain that

$$\underset{(\boldsymbol{\omega},\boldsymbol{\omega}')\sim C}{H}(\boldsymbol{\omega},\boldsymbol{\omega}') = -\mathrm{KL}(p(\boldsymbol{\omega},\boldsymbol{\omega}')||\hat{p}(\boldsymbol{\omega},\boldsymbol{\omega}')). \tag{24}$$

The entropy of c indicates the Kullback-Leibler divergence of using $\hat{p}$ to approximate p. The difference between p and $\hat{p}$ is the dependence structure between variables. Thus, the extra information indicated by Equation (22) is the internal dependence of weights, verifying its capacity to tackle complicated relations between variables.

From another perspective, since the interactions between variables are explicitly described by copulas, we can incorporate reliable prior information into the construction of models via the copula module. For instance, when the data appear to be Gaussian, the pseudo probability density functions can be set to be Gaussian components, and the copula module is set to be a constant. While the data appear to have tail dependence, it is hard to tackle by Gaussian components. We can use the Clayton copula to pre-train the copula module, which exhibits tail dependencies to describe the interaction. The involved prior information can guide the learning process and lead to a better alignment between the model and existing patterns, enhancing the interpretability.

5.3. The Generalization Ability

The integration of copulas improves the generalization of SKNs. We show its improvements via the following theorems. Define the empirical Rademacher complexity.

Definition 5.1. The empirical Rademacher complexity of $\mathcal{F} = \{f|f \text{ is a binary classifier}\}$ is defined as

$$\hat{\mathcal{R}}(\mathcal{F}) = \mathbb{E}_\sigma[\sup_{f\in\mathcal{F}}(\frac{1}{N}\sum_{i=1}^{N}\sigma_i f(\boldsymbol{x}_i))], \tag{25}$$

where $\sigma_1, \ldots, \sigma_m$ are independent random variables uniformly chosen from $\{-1, +1\}$ and $\{\boldsymbol{x}_i\}_{i=1}^N$ is the dataset.

By Equation (19), the reproducing kernel Hilbert space (RKHS) of CokeNet is

$$\mathcal{G} = \{\Phi(\cdot)|\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m \in \mathbb{R}^d, \Theta_c\}, \tag{26}$$

where Θ_c are the parameters in the copula modules.

Note that the plain spectral kernel mapping is

$$\tilde{\Phi}(\boldsymbol{x}) = \frac{1}{\sqrt{2D}}\begin{bmatrix}\cos(\boldsymbol{\omega}_1^\top\boldsymbol{x}) + \cos(\boldsymbol{\omega}_1'^\top\boldsymbol{x}) \\ \cdots \\ \cos(\boldsymbol{\omega}_D^\top\boldsymbol{x}) + \cos(\boldsymbol{\omega}_D'^\top\boldsymbol{x})\end{bmatrix}. \tag{27}$$

And its RKHS is defined as

$$\tilde{\mathcal{G}} = \{\tilde{\Phi}(\cdot)|\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m \in \mathbb{R}^d\}. \tag{28}$$

We have the following theorem considering the empirical Rademacher complexities of these hypothesis spaces.

Theorem 5.2. *Following the notation in Equation (19) and Equation (27) and considering a dataset $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$, the empirical Rademacher complexity of $\mathcal{G}$ is bounded by*

$$\hat{\mathcal{R}}(\mathcal{G}) \leq \frac{1}{N}[\sum_{i=1}^{N}\sum_{m=1}^{D} 2c^2(\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m)[1+cos((\boldsymbol{\omega}_m-\boldsymbol{\omega}'_m)^\top\boldsymbol{x}_i)]]^{1/2}. \tag{29}$$

The empirical Rademacher complexity of $\tilde{\mathcal{G}}$ is bounded by

$$\hat{\mathcal{R}}(\tilde{\mathcal{G}}) \leq \frac{1}{N}[\sum_{i=1}^{N}\sum_{m=1}^{D} 2[1 + cos((\boldsymbol{\omega}_m - \boldsymbol{\omega}'_m)^\top\boldsymbol{x}_i)]]^{1/2}. \tag{30}$$

Proof. The proof is given in the Appendix. □

Theorem 5.3. *(Shalev-Shwartz & Ben-David, 2014) Given the datasets $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ and the hypothesis space $\mathcal{H}$, assume the loss function ℓ is l-Lipstchitz and $\ell(\cdot) < \infty$. With probability at least $1-\delta$, the following risk bound holds*

$$\epsilon(f^*) - \hat{\epsilon}(f) \leq 2l\hat{\mathcal{R}}(\mathcal{H}) + \mathcal{O}(\sqrt{\frac{log(1/\delta)}{N}}), \tag{31}$$

where $f \in \mathcal{H}$ and $f^* \in \mathcal{H}$ is the most accurate estimator in the hypothesis space. ϵ denotes the expected risk and $\hat{\epsilon}$ denotes the empirical risk.

By Theorem 5.2 and Theorem 5.3, we can guarantee that the proposed model has better generalization ability by constraining that $||c^2(\boldsymbol{\omega}, \boldsymbol{\omega}')|| \leq 1$, which can be done by adding a regularizer to the loss function

$$\ell(y, \hat{y}) = \text{loss}(y, \hat{y}) + \lambda(1 - ||c^2(\boldsymbol{\omega}, \boldsymbol{\omega}')||), \tag{32}$$

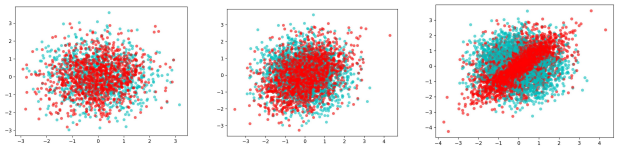
where λ is the regularization parameter. $\hat{y}$ is the prediction and y is the groundtruth.

6. Experiment

In this section, systematical experiments are performed to evaluate our proposed CokeNet. We first empirically demonstrate the efficacy of CokeNet in depicting relations between variables in the synthetic data. Then, we evaluate the performance of CokeNet compared with several state-of-the-art algorithms on six real-world datasets. All the experiments are implemented with PyTorch (Paszke et al., 2019).

6.1. Synthetic Data

To verify the capacity to capture the dependence structures between variables of CokeNet, we elaborate a series of synthetic data and conduct comparison experiments and ablation experiments.



(a) 0.1 (b) 0.5 (c) 0.9

Figure 5. Data points of different dependence degrees. The dependence degrees of blue points are all 0 while the dependence degree of red points is 0.1 in (a), 0.5 in (b) and 0.9 in (c). Note that with an increase in the degree of dependence, the mixing of differently colored points becomes less distinct, thereby simplifying the classification task.

Table 1. Classification accuracy of synthetic data. The best results are highlighted in **bold**. And DD stands for dependence degree.

DD	SKN	COKENET	COKENET-P	COKENET-R
0.01	0.5616	**0.5800**	0.5250	0.5383
0.1	0.5366	**0.5616**	0.5516	0.5300
0.3	0.5433	**0.5750**	0.5483	0.5300
0.5	0.5916	**0.5950**	0.5816	0.5750
0.7	0.6516	**0.6600**	0.6583	0.6400
0.9	0.7616	**0.7650**	0.7566	0.7583
0.999	0.9700	**0.9750**	**0.9750**	0.9716

Data We generate 7 pairs of 2-dimensional synthetic data points with different degrees of dependences. Each pair contains two groups of data points: one is a group of individually independent data points sampled from $\mathcal{N}(0, 1)$ and the other is a group of data points sampled from $\mathcal{N}(0, \boldsymbol{\Sigma})$, where the $\Sigma = \begin{bmatrix} 1 & \alpha \\ \alpha & 1 \end{bmatrix}$. The α, regarded as the dependence degree of each pair, is set to be $0.1, 0.3, 0.5, 0.7, 0.9, 0.01, 0.999$ in each pair respectively. Then a binary classification task is conducted in each pair. We illustrate three pairs of synthetic data points (dependence degree:0.1, 0.5 and 0.9) in Figure 5 to show the complexity of the task associated with the dependence degree. In Figure 5(a), the red and blue points are intermixed with no clear separation or pattern. In Figure 5(b), it starts to show that the variables of red points are linearly correlated. While in Figure 5(c), the dependence relation is very obvious. As the dependence degree increases from Figure 5(a) to Figure 5(c), the complexity and challenge of the classification task simultaneously drops.

Compared Methods We compared the performances of (1) our method (**CokeNet**), (2) plain spectral kernel network without copulas (**SKN**), and several variants of our method: (3) **CokeNet-P**: replace the copula module $c(\boldsymbol{\omega}, \boldsymbol{\omega}')$ with parameters of the same dimension but with no dependence with $\boldsymbol{\omega}$, (4) **CokeNet-R**: change the copula network to a neural network with ReLU function as its activation function. The architectures of all networks are set to be $2 \times 4 \times 4 \times 2$, with a softmax function at the end for classification.

Results Table 1 represents the results of all 7 pairs. The best results are highlighted in **bold**. The performance of all models improves with the increase in dependence degree, which is associated with the decrease in task complexity. CokeNet consistently leads across almost all tasks. The performance gap between CokeNet and other methods is more obvious at lower dependence degrees, which suggests the effectiveness of CokeNet in dealing with dependent data even when the dependence structure is nontrivial. The ablation experiment between CokeNet, CokeNet-R and CokeNet-P indicates that adding arbitrary parameters to the SKN cannot effectively capture the complex dependent relations.

Table 2. Classification accuracy in real-world datasets. The best results are highlighted in **bold**.

	DSKN	ASKL	CosNet	GRFF	SRFF	CokeNet-P	CokeNet-R	CokeNet
DistalPhalanxOutlineCorrect	0.7789	0.7608	0.7717	0.6131	0.7754	0.7282	0.7355	**0.8007**
Earthquakes	0.7482	0.6762	0.6043	0.7482	0.6763	0.7482	0.7482	**0.7553**
HandOutlines	0.9081	0.8378	0.8838	0.6459	0.9108	0.9081	0.9054	**0.9135**
DistalPhalanxTW	0.6474	0.6474	0.6043	0.6331	0.6331	0.6187	0.6402	**0.6618**
ProximalPhalanxOutlineCorrect	0.8797	0.8659	0.8729	0.6838	0.8866	0.9037	0.9003	**0.9072**
ProximalPhalanxTW	0.7804	0.8048	0.8049	0.7854	0.7805	0.7951	0.7707	**0.8146**

6.2. Real-World Data

To demonstrate the efficacy of CokeNet, we conduct comparison and ablation experiments in several real-world datasets.
Datasets We systematically evaluate the performance of CokeNet on six standard classification tasks from UCI repository (Blake, 1998).

Compared Methods We compare the proposal with several mainstream spectral kernel methods, as follows: **DSKN** (Xue et al., 2019): Deep Spectral Kernel Network which embedded non-stationary spectral kernel into deep architectures; **ASKL** (Li et al., 2020): Automated Spectral Kernel Learning which incorporates the process of finding suitable kernels and model training in a learning framework; **CosNet** (Xue et al., 2023): Complex-valued spectral kernel network, which generalizes spectral kernel mapping in real number domain to complex number domain; **GRFF** (Fang et al., 2023): Generative Random Fourier Features, a one-stage kernel learning approach that models some latent distribution of the kernel via a generative network based on the random Fourier features; **SRFF** (Zhang et al., 2017): Stacked kernel network that learns a hierarchy of RKHS functions via random Fourier feature representation.

Implementation Details Following the common practice, Gaussian probability density function is set to be the spectral density of all kernel methods. We denote the number of features in the dataset as n_{feature}, the number of classes as n_{classes}. The scale of all networks is uniformly set to $n_{\text{feature}} \times 512 \times 256 \times n_{\text{classes}}$. The architecture of the neural network, with ReLU functions as its activation function in CokeNet-R, is set to be as same as the copula net. Detailed information about the setting in each dataset is in the Appendix. All method is trained by ADAM (Kingma & Ba, 2015) using mean squared error (MSE) loss. The learning rate is 0.001 without weight decay. Accuracy is the measurement. Table 2 demonstrates the results of real-world datasets. The best results are highlighted in **bold**.

Results Overall, CokeNet consistently outperforms other models across all datasets. GRFF exhibits varied performances because of its progressive training strategy. In the backpropagation on a single batch, this scheme first only updates the last layer and the corresponding generator. Then parameters are added to the training sequence layer by layer, starting from the penultimate layer. Updating parameters multiple times during a single backpropagation process incurs unnecessary disturbances. The ablation studies between CokeNet, CokeNet-R and CokeNet-P indicate the efficacy of adding copula modules compared to arbitrary parameters and arbitrary neural networks. Additionally, while CokeNet-R has a lot more parameters than CokeNet-P, the performances of CokeNet-R are not always better. This suggests that merely making the architectures of the copula complicated does not guarantee better performance.

7. Conclusion

In this paper, we propose CokeNet, a copula-nested spectral kernel network. In CokeNet, we first redefine spectral densities in the form of copulas. Secondly, we generalize Archimedean copulas to a hierarchical architecture and form the copula module. Subsequently, copula-nested spectral kernel mapping is obtained by integrating the copula module into the novel spectral density. CokeNet significantly expands the diversity of the hypothesis space and allows for an excavation of the complex dependencies inherent in data variables, overcoming the limitations posed by traditional spectral kernel networks. Theoretical analysis verifies the increase in the uncertainty of spectral densities, an improved data description capacity and a better generalization ability. The experimental results affirm the superiority of CokeNet over relevant state-of-the-art algorithms.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. 62076062 and 62306070) and the Social Development Science and Technology Project of Jiangsu Province (No. BE2022811). Furthermore, the work was also supported by the Big Data Computing Center of Southeast University.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Avron, H., Kapralov, M., Musco, C., Musco, C., Velingker, A., and Zandieh, A. Random fourier features for kernel ridge regression: Approximation bounds and statistical guarantees. In *Proceedings of the 34th International Conference on Machine Learning ICML*, volume 70, pp. 253–262, Sydney, NSW, Australia, 2017.

Bernstein, S. Sur les fonctions absolument monotones. *Acta Mathematica*, 52:1–66, 1929.

Blake, C. Uci repository of machine learning databases. *http://www. ics. uci. edu/˜ mlearn/MLRepository. html*, 1998.

Disegna, M., D'Urso, P., and Durante, F. Copula-based fuzzy clustering of spatial time series. *Spatial Statistics*, 21:209–225, 2017.

Fang, K., Liu, F., Huang, X., and Yang, J. End-to-end kernel learning via generative random fourier features. *Pattern Recognition*, 134:109057, 2023.

Flandrin, P. *Time-frequency/time-scale analysis*. Academic press, 1998.

Genest, C. and Rivest, L.-P. Statistical inference procedures for bivariate archimedean copulas. *Journal of the American statistical Association*, 88(423):1034–1043, 1993.

Janke, T., Ghanmi, M., and Steinke, F. Implicit generative copulas. In Ranzato, M., Beygelzimer, A., Dauphin, Y. N., Liang, P., and Vaughan, J. W. (eds.), *In Advances in Neural Information Processing Systems 34*, volume 34, pp. 26028–26039, virtual, 2021.

Jaworski, P., Durante, F., Hardle, W. K., and Rychlik, T. *Copula theory and its applications*, volume 198. 2010.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR*, San Diego, CA, USA, 2015.

Lázaro-Gredilla, M., Quinonero-Candela, J., Rasmussen, C. E., and Figueiras-Vidal, A. R. Sparse spectrum gaussian process regression. *The Journal of Machine Learning Research*, 11:1865–1881, 2010.

Li, J., Liu, Y., and Wang, W. Automated spectral kernel learning. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence, AAAI 2020*, volume 34, pp. 4618–4625, 2020.

Li, J., Liu, Y., and Wang, W. Convolutional spectral kernel learning with generalization guarantees. *Artificial Intelligence*, 313:103803, 2022.

Li, Z., Ton, J.-F., Oglic, D., and Sejdinovic, D. Towards a unified analysis of random fourier features. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 97, pp. 3905–3914, Long Beach, California, USA, 2019.

Ling, C. K., Fang, F., and Kolter, J. Z. Deep archimedean copulas. In *Advances in Neural Information Processing Systems 33*, volume 33, pp. 1535–1545, Virtual, 2020.

Liu, F., Huang, X., Chen, Y., and Suykens, J. A. Random features for kernel approximation: A survey on algorithms, theory, and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):7128–7148, 2021.

Ng, Y., Hasan, A., Elkhalil, K., and Tarokh, V. Generative archimedean copulas. In de Campos, C. P., Maathuis, M. H., and Quaeghebeur, E. (eds.), *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence, UAI 2021, Virtual Event, 27-30 July 2021*, volume 161 of *Proceedings of Machine Learning Research*, pp. 643–653. AUAI Press, 2021.

Oh, D. H. and Patton, A. J. Time-varying systemic risk: Evidence from a dynamic copula model of cds spreads. *Journal of Business & Economic Statistics*, 36(2):181–195, 2018.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. 32:8024–8035, 2019.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems 20*, volume 20, pp. 1177–1184, Vancouver, British Columbia, Canada, 2007.

Remes, S., Heinonen, M., and Kaski, S. Non-stationary spectral kernels. In *Advances in Neural Information Processing Systems 30*, volume 30, pp. 4642–4651, Long Beach, CA, USA, 2017.

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

Shalev-Shwartz, S. and Ben-David, S. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Shawe-Taylor, J. and Cristianini, N. *Kernel methods for pattern analysis*. 2004.

Sinha, A. and Duchi, J. C. Learning kernels with random features. In *Advances in Neural Information Processing Systems 29*, pp. 1298–1306, Barcelona, Spain, 2016.

Sklar, M. Fonctions de répartition à n dimensions et leurs marges. In *Annales de l'ISUP*, volume 8, pp. 229–231, 1959.

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

Williams, C. and Seeger, M. Using the nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems 13*, pp. 682–688, Denver, CO, USA, 2000.

Wilson, A. and Adams, R. Gaussian process kernels for pattern discovery and extrapolation. In *Proceedings of the 30th International Conference on Machine Learning, ICML*, volume 28, pp. 1067–1075, Atlanta, GA, USA, 2013.

Xu, P., Wang, Y., Chen, X., and Tian, Z. Deep kernel learning networks with multiple learning paths. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pp. 4438–4442, Virtual and Singapore, 2022.

Xue, H. and Wu, Z.-F. Baker-nets: Bayesian random kernel mapping networks. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence, IJCAI*, pp. 3073–3079, 2020.

Xue, H., Wu, Z.-F., and Sun, W.-X. Deep spectral kernel learning. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence, IJCAI*, pp. 4019–4025, Macao, China, 2019.

Xue, Y., Fang, P., Tian, J., Zhu, S., et al. Cosnet: A generalized spectral kernel network. In *Advances in Neural Information Processing Systems 37*, 2023.

Yaglom, A. M. *Correlation Theory of Stationary and Related Random Functions, Volume I: Basic Results*, volume 131. 1987.

Yang, J., Sindhwani, V., Avron, H., and Mahoney, M. Quasi-monte carlo feature maps for shift-invariant kernels. In *Proceedings of the 31th International Conference on Machine Learning, ICML*, volume 32, pp. 485–493, Beijing, China, 2014.

Zhang, S., Li, J., Xie, P., Zhang, Y., Shao, M., Zhou, H., and Yan, M. Stacked kernel network. *arXiv preprint arXiv:1711.09219*, 2017.

Appendix

In the Appendix, we provide:

- Detailed derivation of equations and theorems.
- The Wigner distribution functions of kernels.
- Loss surface of SKN and CokeNet in the experiments described in Section 6.1.
- Extensive experiments.
- Additional information about the real-world classification experiments.

A. The Detailed Derivation of Equation (22)

The entropy of weights $(\boldsymbol{\omega}, \boldsymbol{\omega}')$ is defined as

$$\underset{(\boldsymbol{\omega},\boldsymbol{\omega}')\sim P}{H}(\boldsymbol{\omega}, \boldsymbol{\omega}') = -\int p(\boldsymbol{\omega}, \boldsymbol{\omega}')log(p(\boldsymbol{\omega}, \boldsymbol{\omega}'))d\boldsymbol{\omega}d\boldsymbol{\omega}', \tag{33}$$

where $(\boldsymbol{\omega}, \boldsymbol{\omega}') \sim P$. We define the joint probability density function of $(\boldsymbol{\omega}, \boldsymbol{\omega}')$ as $c(\boldsymbol{\omega}, \boldsymbol{\omega}')\hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega}')$ and $\hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega}') = p(\boldsymbol{\omega})p(\boldsymbol{\omega}')$ is the product of the marginal probability density of $\boldsymbol{\omega}$ and $\boldsymbol{\omega}'$.

Therefore, Equation (33) can be written as

$$\begin{aligned}
\underset{(\boldsymbol{\omega},\boldsymbol{\omega}')\sim P}{H}(\boldsymbol{\omega}, \boldsymbol{\omega}') &= -\int p(\boldsymbol{\omega}, \boldsymbol{\omega}')log(p(\boldsymbol{\omega}, \boldsymbol{\omega}'))d\boldsymbol{\omega}d\boldsymbol{\omega}' \\
&= -\int c(\boldsymbol{\omega}, \boldsymbol{\omega}')\hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega}')[log(c(\boldsymbol{\omega}, \boldsymbol{\omega}')) + log(\hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega}'))]d\boldsymbol{\omega}d\boldsymbol{\omega}' \\
&= -\int c(\boldsymbol{\omega}, \boldsymbol{\omega}')log(c(\boldsymbol{\omega}, \boldsymbol{\omega}'))d\boldsymbol{\omega}d\boldsymbol{\omega}' - \int \hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega}')log(\hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega}'))d\boldsymbol{\omega}d\boldsymbol{\omega}' \\
&= \underset{(\boldsymbol{\omega},\boldsymbol{\omega}')\sim C}{H}(\boldsymbol{\omega}, \boldsymbol{\omega}') + \underset{(\boldsymbol{\omega},\boldsymbol{\omega}')\sim P}{H}(\boldsymbol{\omega}, \boldsymbol{\omega}').
\end{aligned} \tag{34}$$

B. The Detailed Derivation of Equation (24)

Given that $p(\boldsymbol{\omega}, \boldsymbol{\omega}') = c(\boldsymbol{\omega}, \boldsymbol{\omega}')\hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega}')$, we have

$$\begin{aligned}
\mathrm{KL}(p(\boldsymbol{\omega}, \boldsymbol{\omega}')||\hat{p}((\boldsymbol{\omega}, \boldsymbol{\omega}')) &= \int p(\boldsymbol{\omega}, \boldsymbol{\omega}')log(\frac{p(\boldsymbol{\omega}, \boldsymbol{\omega}')}{\hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega}')})d\boldsymbol{\omega}d\boldsymbol{\omega}' \\
&= \int c(\boldsymbol{\omega}, \boldsymbol{\omega}')\hat{p}(\boldsymbol{\omega}, \boldsymbol{\omega}')log(c(\boldsymbol{\omega}, \boldsymbol{\omega}'))d\boldsymbol{\omega}d\boldsymbol{\omega}' \\
&= \int c(\boldsymbol{\omega}, \boldsymbol{\omega}')log(c(\boldsymbol{\omega}, \boldsymbol{\omega}'))d\boldsymbol{\omega}d\boldsymbol{\omega}' \\
&= -\underset{(\boldsymbol{\omega},\boldsymbol{\omega}')\sim C}{H}(\boldsymbol{\omega}, \boldsymbol{\omega}').
\end{aligned} \tag{35}$$

C. The Wigner Distribution of Kernels.

Recall that the Wigner distribution function of a kernel k is defined as $W_k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$

$$W_k(\boldsymbol{x}, \boldsymbol{\omega}) = \int_{\mathbb{R}^d} k(\boldsymbol{x} + \frac{\boldsymbol{\tau}}{2}, x - \frac{\boldsymbol{\tau}}{2})e^{-2i\pi\boldsymbol{\omega}^\top\boldsymbol{\tau}}d\boldsymbol{\tau}. \tag{36}$$

And the copula-nested spectral kernel can be written as

$$k(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}_{(\boldsymbol{\omega},\boldsymbol{\omega}')\sim\hat{P}}[\tau(\boldsymbol{\omega}, \boldsymbol{\omega}', \boldsymbol{x}, \boldsymbol{x}')c(\boldsymbol{\omega}, \boldsymbol{\omega}')] \approx \frac{1}{D}\sum_{m=1}^{D}\tau(\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m, \boldsymbol{x}, \boldsymbol{x}')c(\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m), \tag{37}$$

where $\tau(\boldsymbol{\omega}, \boldsymbol{\omega}', \boldsymbol{x}, \boldsymbol{x}')$ is

$$\frac{1}{4}[cos(\boldsymbol{\omega}^\top \boldsymbol{x} - \boldsymbol{\omega}'^\top \boldsymbol{x}') + cos(\boldsymbol{\omega}'^\top \boldsymbol{x} - \boldsymbol{\omega}^\top \boldsymbol{x}') + cos(\boldsymbol{\omega}^\top \boldsymbol{x} - \boldsymbol{\omega}^\top \boldsymbol{x}') + cos(\boldsymbol{\omega}'^\top \boldsymbol{x} - \boldsymbol{\omega}'^\top \boldsymbol{x}')]. \tag{38}$$

By inserting Equation (37) into Equation (36), and set $\boldsymbol{A}_m = \boldsymbol{\omega}_m - \boldsymbol{\omega}_m$, $\boldsymbol{B}_m = \frac{\boldsymbol{\omega}_m + \boldsymbol{\omega}_m}{2}$, $W_k(\boldsymbol{x}, \boldsymbol{\omega})$ of Coke is

$$\begin{aligned}
&\frac{1}{D}\sum_{m=1}^{D}\int_{\mathbb{R}^d} \tau(\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m, \boldsymbol{x} + \frac{\boldsymbol{\tau}}{2}, \boldsymbol{x} - \frac{\boldsymbol{\tau}}{2}) c(\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m) e^{-2i\pi\boldsymbol{\omega}^\top \boldsymbol{\tau}} d\boldsymbol{\tau} \\
&= \frac{1}{4D}\sum_{m=1}^{D}\int_{\mathbb{R}^d} [cos(\boldsymbol{A}_m^\top \boldsymbol{x} + \boldsymbol{B}_m^\top \boldsymbol{\tau}) + cos(-\boldsymbol{A}_m^\top \boldsymbol{x} + \boldsymbol{B}_m^\top \boldsymbol{\tau}) + cos(\boldsymbol{\omega}_m^\top \boldsymbol{\tau}) + cos(\boldsymbol{\omega}_m'^\top \boldsymbol{\tau})] c(\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m) e^{-2i\pi\boldsymbol{\omega}^\top \boldsymbol{\tau}} d\boldsymbol{\tau} \\
&= \frac{1}{4D}\sum_{m=1}^{D} c(\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m) \int_{\mathbb{R}^d} [cos(\boldsymbol{A}_m^\top \boldsymbol{x} + \boldsymbol{B}_m^\top \boldsymbol{\tau}) + cos(-\boldsymbol{A}_m^\top \boldsymbol{x} + \boldsymbol{B}_m^\top \boldsymbol{\tau}) + cos(\boldsymbol{\omega}_m^\top \boldsymbol{\tau}) + cos(\boldsymbol{\omega}_m'^\top \boldsymbol{\tau})] e^{-2i\pi\boldsymbol{\omega}^\top \boldsymbol{\tau}} d\boldsymbol{\tau}.
\end{aligned} \tag{39}$$

For simplicity of understanding, we break down the calculation of the integral into four parts. By Euler's formula, the calculation of the first part $\int_{\mathbb{R}^d} cos(\boldsymbol{A}_m^\top \boldsymbol{x} + \boldsymbol{B}_m^\top \boldsymbol{\tau}) e^{-2i\pi\boldsymbol{\omega}^\top \boldsymbol{\tau}} d\boldsymbol{\tau}$ is as follows

$$\begin{aligned}
&\frac{1}{2}\int_{\mathbb{R}^d} [e^{i(\boldsymbol{A}_m^\top \boldsymbol{x} + \boldsymbol{B}_m^\top \boldsymbol{\tau})} + e^{-i(\boldsymbol{A}_m^\top \boldsymbol{x} + \boldsymbol{B}_m^\top \boldsymbol{\tau})}] e^{-2i\pi\boldsymbol{\omega}^\top \boldsymbol{\tau}} d\boldsymbol{\tau} \\
&= \frac{1}{2}[\int_{\mathbb{R}^d} e^{i(\boldsymbol{A}_m^\top \boldsymbol{x} + \boldsymbol{B}_m^\top \boldsymbol{\tau})} e^{-2i\pi\boldsymbol{\omega}^\top \boldsymbol{\tau}} d\boldsymbol{\tau} + \int_{\mathbb{R}^d} e^{-i(\boldsymbol{A}_m^\top \boldsymbol{x} + \boldsymbol{B}_m^\top \boldsymbol{\tau})}] e^{-2i\pi\boldsymbol{\omega}^\top \boldsymbol{\tau}} d\boldsymbol{\tau}] \\
&= \frac{1}{2}[e^{i\boldsymbol{A}_m^\top \boldsymbol{x}} \int_{\mathbb{R}^d} e^{i(\boldsymbol{B}_m - 2\pi\boldsymbol{\omega})^\top \boldsymbol{\tau}} d\boldsymbol{\tau} + e^{-i\boldsymbol{A}_m^\top \boldsymbol{x}} \int_{\mathbb{R}^d} e^{-i(\boldsymbol{B}_m + 2\pi\boldsymbol{\omega})^\top \boldsymbol{\tau}} d\boldsymbol{\tau}] \\
&= \frac{1}{2}(2\pi)^d [e^{i\boldsymbol{A}_m^\top \boldsymbol{x}} \delta(\boldsymbol{B}_m - 2\pi\boldsymbol{\omega}) + e^{-i\boldsymbol{A}_m^\top \boldsymbol{x}} \delta(\boldsymbol{B}_m + 2\pi\boldsymbol{\omega})],
\end{aligned} \tag{40}$$

where $\delta(\cdot)$ is the dirac delta function. Similarly, that of the second part is the same. And we can obtain the integral of the third part is

$$\frac{1}{2}(2\pi)^d [\delta(\boldsymbol{\omega}_m - 2\pi\boldsymbol{\omega}) + \delta(\boldsymbol{\omega}_m + 2\pi\boldsymbol{\omega})]. \tag{41}$$

While that of the fourth part is

$$\frac{1}{2}(2\pi)^d [\delta(\boldsymbol{\omega}'_m - 2\pi\boldsymbol{\omega}) + \delta(\boldsymbol{\omega}'_m + 2\pi\boldsymbol{\omega}). \tag{42}$$

Combining these four parts together, we obtain the Wigner distribution of the Coke

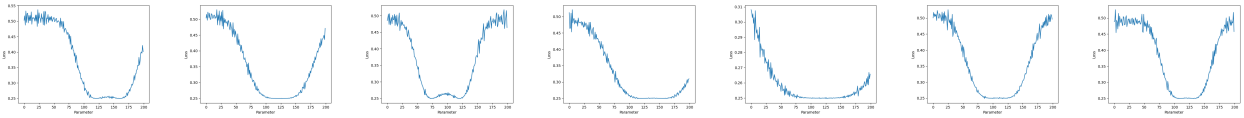
$$\begin{aligned}
\frac{(2\pi)^d}{8D}\sum_{m=1}^{D} c(\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m) [&2e^{i\boldsymbol{A}_m^\top \boldsymbol{x}} \delta(\boldsymbol{B}_m - 2\pi\boldsymbol{\omega}) + 2e^{-i\boldsymbol{A}_m^\top \boldsymbol{x}} \delta(\boldsymbol{B}_m + 2\pi\boldsymbol{\omega}) \\
&+ \delta(\boldsymbol{\omega}_m - 2\pi\boldsymbol{\omega}) + \delta(\boldsymbol{\omega}_m + 2\pi\boldsymbol{\omega}) + \delta(\boldsymbol{\omega}'_m - 2\pi\boldsymbol{\omega}) + \delta(\boldsymbol{\omega}'_m + 2\pi\boldsymbol{\omega})].
\end{aligned} \tag{43}$$

And we can obtain the Wigner distribution of plain spectral kernel by setting $c(\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m) \equiv 1$.

In terms of the figures in Figure 3, for the simplicity of visualization, we consider the simple scenario where $d = 1$, *i.e.* $\omega_m,\ \omega'_m,\ A_m,\ B_m \in \mathbb{R}$. And set $x = ai,\ a \in \mathbb{R}$, i is the imaginary unit.

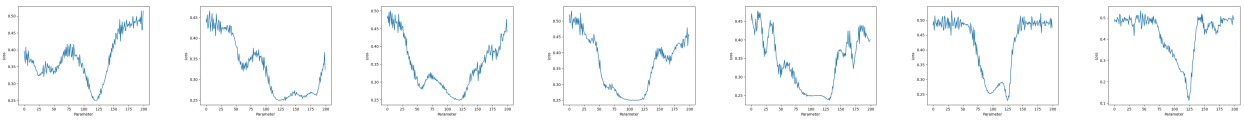
D. Loss Surface

In Figure 4, we demonstrate the change of loss according to parameters in the experiment in Section 6.1. Here we represent the loss surface of CokeNet and SKN on the 7 pairs of synthetic data. Note that the figures in Figure 4 are the results of the dependence degree 0.1.



(a) 0.01 (b) 0.1 (c) 0.3 (d) 0.5 (e) 0.7 (f) 0.9 (g) 0.999

Figure 6. Loss surface of CokeNet on synthetic data of different dependence degrees.



(a) 0.01 (b) 0.1 (c) 0.3 (d) 0.5 (e) 0.7 (f) 0.9 (g) 0.999

Figure 7. Loss surface of SKN on synthetic data of different dependence degrees.

E. Extensive Experiments

To further verify the superiority of CokeNet, we conduct regression tasks on several UCI datasets and an image classification task on the CIAFR10 datasets compared to some relevant algorithms. The results are represented in the following tables.

Table 3. The MSE loss of DSKN, CokeNet, CosNet, SRFF on several regression datasets. The best results are highlighted in **bold**.

Dataset	DSKN	CokeNet	CosNet	SRFF
power	0.4812	**0.1004**	0.1021	0.1009
concrete	0.8803	**0.7977**	0.8891	0.8924
yacht	1.9578	**1.6857**	2.3417	2.1974
airfoil	0.1128	**0.0980**	0.1070	0.4986
boston	0.4938	**0.4492**	0.5476	0.7136
wine_red	0.6058	**0.5967**	0.6560	0.6423
wine_white	0.7598	**0.7476**	0.7525	0.8347

Table 4. The accuracy of DSKN, CokeNet, CosNet, SRFF on CIFAR10. The best results are highlighted in **bold**.

Dataset	DSKN	CokeNet	CosNet	SRFF
Accuracy	0.8148	**0.8172**	0.6631	0.7359

F. Proof of Theorem 5.2

Theorem F.1. *Following the notation in Equation (27) and Equation (19) and considering a dataset* $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$*, the empirical Rademacher complexity of* $\mathcal{G}$ *is bounded by*

$$\hat{\mathcal{R}}(\mathcal{G}) \leq \frac{1}{N}[\sum_{i=1}^{N}\sum_{m=1}^{D} 2c^2(\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m)[1 + cos((\boldsymbol{\omega}_m - \boldsymbol{\omega}'_m)^\top \boldsymbol{x}_i)]^{1/2}. \tag{44}$$

The empirical Rademacher complexity of $\tilde{\mathcal{G}}$ *is bounded by*

$$\hat{\mathcal{R}}(\tilde{\mathcal{G}}) \leq \frac{1}{N}[\sum_{i=1}^{N}\sum_{m=1}^{D} 2[1 + cos((\boldsymbol{\omega}_m - \boldsymbol{\omega}'_m)^\top \boldsymbol{x}_i)]^{1/2}. \tag{45}$$

Proof. Following the same notation as above and the definition of empirical Rademacher complexity (Equation (25)), we obtain that

$$\hat{\mathcal{R}}(\mathcal{G}) = \mathbb{E}_\sigma[\sup_{\Phi\in\mathcal{G}}(\frac{1}{N}\sum_{i=1}^{N}\sigma_i\Phi(\boldsymbol{x}_i))] \leq \frac{1}{N}\mathbb{E}_\sigma[\sup_{\Phi\in\mathcal{G}}(||\sum_{i=1}^{N}\sigma_i\Phi(\boldsymbol{x}_i))||]. \tag{46}$$

Since $||\sum_{i=1}^{N}\sigma_i\Phi(\boldsymbol{x}_i))||^2 = \sum_{i=1}^{N}\sum_{j=1}^{N}\sigma_i\sigma_j\Phi^\top(\boldsymbol{x}_i)\Phi(\boldsymbol{x}_j)$, we have

$$\hat{\mathcal{R}}(\mathcal{G}) \leq \frac{1}{N}\mathbb{E}_\sigma[\sup_{\Phi\in\mathcal{G}}(\sum_{i=1}^{N}\sum_{j=1}^{N}\sigma_i\sigma_j\Phi^\top(\boldsymbol{x}_i)\Phi(\boldsymbol{x}_j))^{1/2}] = \frac{1}{N}\sup_{\Phi\in\mathcal{G}}(\sum_{i=1}^{N}\Phi^\top(\boldsymbol{x}_i)\Phi(\boldsymbol{x}_i))^{1/2}. \tag{47}$$

Next, we use Equation (18), which is equivalent to Equation (19),

$$\begin{aligned}\Phi^\top(\boldsymbol{x})\Phi(\boldsymbol{x}) &= \sum_{m=1}^{D} c^2(\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m)[(cos(\boldsymbol{\omega}_m^\top\boldsymbol{x}) + cos(\boldsymbol{\omega}_m'^\top\boldsymbol{x}))^2 + (sin(\boldsymbol{\omega}_m^\top\boldsymbol{x}) + sin(\boldsymbol{\omega}_m'^\top\boldsymbol{x}))^2] \\ &= \sum_{m=1}^{D} 2c^2(\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m)[1 + cos((\boldsymbol{\omega}_m - \boldsymbol{\omega}'_m)^\top\boldsymbol{x})].\end{aligned} \tag{48}$$

Insert Equation (48) into Equation (47), we obtain that

$$\hat{\mathcal{R}}(\mathcal{G}) \leq \frac{1}{N}[\sum_{i=1}^{N}\sum_{m=1}^{D} 2c^2(\boldsymbol{\omega}_m, \boldsymbol{\omega}'_m)[1 + cos((\boldsymbol{\omega}_m - \boldsymbol{\omega}'_m)^\top\boldsymbol{x}_i)]]^{1/2}. \tag{49}$$

By setting $c(\boldsymbol{\omega}, \boldsymbol{\omega}') = 1$, we obtain the bound for the plain non-stationary spectral kernel. □

G. Additional Information of Real-World Classification Experiments

Table 5. Detailed information about the real-world experiments

Dataset	Type	Input.Dim	Train.Num	Test.Num	Classes
DistalPhalanxOutlineCorrect	Image	80	400	139	3
Earthquakes	Sensor	512	322	139	2
HandOutlines	Image	2709	1000	370	2
DistalPhalanxTW	Image	80	400	139	6
ProximalPhalanxOutlineCorrect	Image	80	600	291	2
ProximalPhalanxTW	Image	80	400	205	6

Table 6. Architecture of networks in each dataset.

Dataset	Architecture
DistalPhalanxOutlineCorrect	$80 \times 512 \times 256 \times 3$
Earthquakes	$512 \times 512 \times 256 \times 2$
HandOutlines	$2709 \times 512 \times 256 \times 2$
DistalPhalanxTW	$80 \times 512 \times 256 \times 6$
ProximalPhalanxOutlineCorrect	$80 \times 512 \times 256 \times 2$
ProximalPhalanxTW	$80 \times 512 \times 256 \times 6$