Flow-based Spectral Kernel Learning for Nonstationary Attention

Zi Yang[1,*] Ying Li[2,*] Alejandro Lancho[3]
Darío C. Larese[3] Pablo M. Olmos[3,†] Michael Minyi Zhang[1,†]

[1]School of Computing and Data Science, The University of Hong Kong
[2]Department of Statistics and Data Science, National University of Singapore
[3]Department of Signal Theory and Communications, Universidad Carlos III de Madrid

ziyang2023@connect.hku.hk ying-li@nus.edu.sg
alancho@ing.uc3m.es dcabezas@tsc.uc3m.es
pamartin@ing.uc3m.es mzhang18@hku.hk

[*]Equal contribution. [†]Corresponding authors.

Abstract

Kernelized attention provides a useful perspective for understanding self-attention as a similarity-based aggregation mechanism, but existing approaches often rely on fixed or stationary kernels, limiting their ability to adapt the similarity structure to complex token interactions. In this paper, we propose flow-based spectral kernel attention (FSKA), a framework for learning expressive nonstationary attention kernels. FSKA formulates kernelized attention through spectral density learning and parameterizes the resulting bivariate spectral density with normalizing flows. This design enables flexible and regularizable kernel learning while remaining compatible with scalable linear-attention computation. We further show that expressive kernel learning enables a simplified attention design with removed query–key projections and fixed orthogonal value projections, resulting in a more parameter-efficient architecture. Controlled experiments on classification, ablation, and scalability profiling show that FSKA achieves competitive or improved performance over existing attention mechanisms, while reducing projection-related trainable parameters by approximately 40%–47% compared with most baselines and maintaining favorable linear-attention memory scaling. Code Available: https://github.com/ziyang18/FSKA.

1 Introduction

Transformers have become the dominant architecture for sequence modeling in natural language processing, vision, and multimodal learning, largely due to the success of self-attention [Vaswani et al., 2017, Brown et al., 2020, Dosovitskiy et al., 2021]. At its core, self-attention computes pairwise token similarities and uses them to aggregate information. This similarity-based view has motivated a growing line of work that interprets attention as a kernel machine [Tsai et al., 2019]. Under this kernel perspective, different kernel choices induce different similarity measures for token aggregation, with standard softmax attention corresponding to one particular kernel.

This kernel view has led to substantial progress on efficient attention. Methods such as random feature attention (RKA) and the Performer approximate the softmax kernel using random features, enabling linear-time attention with respect to sequence length [Peng et al., 2021, Choromanski et al., 2021]. However, these approaches typically rely on fixed or stationary kernels, whose similarity measures do not adapt to the input data. As a result, they are often limited in modeling complex or context dependent (nonstationary) interactions between tokens.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

To overcome the rigidity of fixed kernels, several recent works have explored learnable kernel attention under the Bochner's theorem. These methods learn the spectral density of the kernel and approximate the resulting kernel using random Fourier features (RFFs) for scalable attention. For example, GMM-RKS parameterizes the spectral density with Gaussian mixtures, while implicit kernel attention (IKAN) learns the spectral density through neural-network (NN) [Pal Chowdhury et al., 2022, Song et al., 2021, Li et al., 2019]. Although these methods improve flexibility, GMM-RKS can become costly when many components are needed to model complex spectra, and IKAN lacks an explicit density for principled spectral regularization and may suffer from overfitting or spectral collapse (see Figure 1). In summary, existing approaches often fall between one of two extremes: fixed kernels are efficient but too restrictive, whereas highly flexible kernels are more expressive but often lack explicit spectral densities for regularization or can become computationally burdensome.

In this paper, we address this gap by proposing "flow-based spectral kernel attention" (FSKA), a flexible and efficient framework for learning nonstationary attention kernels. Our main contributions are summarized as follows:

- We formulate nonstationary attention kernel learning as spectral density learning. To obtain flexible and stable kernels, we parameterize the spectral density with normalizing flows and regularize it with a structured spectral prior. The resulting kernel remains compatible with RFF-based linear attention.
- We show that expressive kernel learning enables a simplified attention design with removed query–key projections and fixed orthogonal value projections, resulting in a more parameter-efficient architecture.
- We conduct experiments across regression and classification tasks, together with ablation and scalability studies. FSKA achieves competitive or improved performance across these settings, reduces projection-related trainable parameters by approximately 40%–47% compared with most baselines, and maintains a low-growth memory profile close to linear-attention baselines.

2 Background

This section briefly reviews self-attention in Section 2.1, its kernel interpretation in Section 2.2, and RFFs for scalable kernel attention in Section 2.3.

2.1 Self-Attention

Given an input sequence $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_L] \in \mathbb{R}^{L \times d_x}$ consisting of L tokens with input embedding dimension d_x, the self-attention mechanism [Vaswani et al., 2017] computes pairwise interactions through learned similarity scores. Specifically, the inputs are linearly projected into queries, keys, and values:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q, \qquad \mathbf{K} = \mathbf{X}\mathbf{W}_K, \qquad \mathbf{V} = \mathbf{X}\mathbf{W}_V, \tag{1}$$

where $\mathbf{W}_Q \in \mathbb{R}^{d_x \times d_q}$, $\mathbf{W}_K \in \mathbb{R}^{d_x \times d_k}$, and $\mathbf{W}_V \in \mathbb{R}^{d_x \times d_v}$ are learnable projection matrices. For dot-product attention, we take $d_q = d_k$, so that $\mathbf{Q}\mathbf{K}^\top$ is well-defined. The attention weights are computed by

$$\mathbf{A} = \mathrm{softmax}\Big(\mathbf{Q}\mathbf{K}^\top / \sqrt{d_k}\Big), \tag{2}$$

and the output is given by $\mathbf{Y} = \mathbf{A}\mathbf{V}$. Multi-head self-attention (MHSA) extends this formulation by computing attention over H heads with independent projections and concatenating their outputs. Overall, self-attention can be viewed as a learned similarity mechanism between tokens.

2.2 Kernel View of Attention

A key observation by Tsai et al. [2019] is that self-attention admits a kernel interpretation. Given a non-negative kernel function $\kappa(\cdot, \cdot)$, the generalized attention output at position i can be written as

$$\mathrm{Attn}(\mathbf{X})_i = \frac{\sum_{j=1}^{L} \kappa(\mathbf{q}_i, \mathbf{k}_j)\, \mathbf{v}_j}{\sum_{j'=1}^{L} \kappa(\mathbf{q}_i, \mathbf{k}_{j'})}, \tag{3}$$

where $\mathbf{q}_i = \mathbf{x}_i\mathbf{W}_Q$, $\mathbf{k}_j = \mathbf{x}_j\mathbf{W}_K$, and $\mathbf{v}_j = \mathbf{x}_j\mathbf{W}_V$. When $\kappa(\mathbf{q}, \mathbf{k}) = \exp\big(\mathbf{q}\mathbf{k}^\top/\sqrt{d_k}\big)$, Eq. (3) reduces to standard softmax attention. This formulation reveals that attention is fundamentally a kernel smoother operating on token representations. However, computing Eq. (3) naively requires $\mathcal{O}(L^2)$ time and memory, which limits scalability to long sequences [Katharopoulos et al., 2020].

2.3 Random Fourier Features for Kernel Attention

Random Fourier features (RFF) [Rahimi and Recht, 2007] provide a scalable approximation to shift-invariant kernels by expressing them as inner products in a randomized feature space. By Bochner's theorem, any continuous, shift-invariant kernel $\kappa(\mathbf{x}, \mathbf{y}) = \kappa(\mathbf{x} - \mathbf{y})$ can be represented as the Fourier transform of a non-negative spectral density. In practice, RFF approximates such kernels by sampling random frequencies from the associated spectral distribution and constructing an explicit feature map.

Concretely, let $\boldsymbol{\phi} : \mathbb{R}^{d_x} \to \mathbb{R}^{2M}$ be a nonlinear feature map defined as

$$\boldsymbol{\phi}(\mathbf{x}) = \frac{1}{\sqrt{M}} \left[\sin\big(\mathbf{x}\mathbf{w}_1^\top\big), \ldots, \sin\big(\mathbf{x}\mathbf{w}_M^\top\big), \cos\big(\mathbf{x}\mathbf{w}_1^\top\big), \ldots, \cos\big(\mathbf{x}\mathbf{w}_M^\top\big)\right], \tag{4}$$

where the random vectors $\{\mathbf{w}_m\}_{m=1}^M, \mathbf{w}_m \in \mathbb{R}^{1\times d_x}$ are independently sampled from a spectral density $p(\mathbf{w})$ associated with the target kernel. Under this construction, the inner product of random features provides an unbiased approximation to the kernel:

$$\mathbb{E}_{\mathbf{w}}\left[\boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\phi}(\mathbf{y})\right] = \kappa(\mathbf{x}, \mathbf{y}). \tag{5}$$

This approximation allows the attention kernel to be written as $\kappa(\mathbf{q}, \mathbf{k}) \approx \boldsymbol{\phi}(\mathbf{q})^\top \boldsymbol{\phi}(\mathbf{k})$, which enables linear-time attention [Peng et al., 2021, Choromanski et al., 2021]. However, for methods such as RKA and Performer, the random frequencies are drawn from a predetermined distribution $p(\mathbf{w})$, so the induced spectral density cannot adapt to the data. Moreover, the resulting kernel is stationary, depending only on $\mathbf{q} - \mathbf{k}$ rather than input-dependent patterns. As a result, such RFF-based attention mechanisms are not sufficiently flexible to capture complex or nonstationary patterns in real-world sequences.

3 Method

In this section, we first introduce the nonstationary attention kernel and its random-feature approximation in Section 3.1. We then propose flow-based spectral density modeling for flexible and regularized kernel learning in Section 3.2. Finally, we show how kernel expressiveness simplifies attention design by removing query–key projections, using fixed value projections in Section 3.3.2, and extending the resulting module to multi-head attention in Section 3.4.

3.1 Nonstationary Attention Kernel

Most existing kernelized attention methods rely on stationary kernels (e.g., $k(\mathbf{x}_1, \mathbf{x}_2) = \kappa(\mathbf{x}_1 - \mathbf{x}_2)$) derived from Bochner's theorem, which limits expressiveness and prevents attention from capturing input-dependent, nonstationary patterns. An effective attention kernel should therefore be flexible enough to model complex similarities while remaining scalable with RFFs for linear-time computation. To this end, we build on a universal Bochner theorem that establishes a duality between continuous kernels and bivariate spectral measures [Yang et al., 2025].

Theorem 3.1 (Universal Bochner's Theorem)**.** *A complex-valued bounded continuous kernel $k(\mathbf{x}_1, \mathbf{x}_2)$ on $\mathbb{R}^{d_x}$ is the covariance function of a mean square continuous complex-valued random process on $\mathbb{R}^{d_x}$ if and only if*

$$\begin{aligned} k(\mathbf{x}_1, \mathbf{x}_2) = \frac{1}{4}\int \Big(&\exp\big(i(\mathbf{x}_1\boldsymbol{\omega}_1^\top - \mathbf{x}_2\boldsymbol{\omega}_2^\top)\big) + \exp\big(i(\mathbf{x}_1\boldsymbol{\omega}_2^\top - \mathbf{x}_2\boldsymbol{\omega}_1^\top)\big) \\ &+ \exp\big(i(\mathbf{x}_1\boldsymbol{\omega}_1^\top - \mathbf{x}_2\boldsymbol{\omega}_1^\top)\big) + \exp\big(i(\mathbf{x}_1\boldsymbol{\omega}_2^\top - \mathbf{x}_2\boldsymbol{\omega}_2^\top)\big) \Big)\, u(\mathrm{d}\boldsymbol{\omega}_1, \mathrm{d}\boldsymbol{\omega}_2). \end{aligned} \tag{6}$$

where u is the Lebesgue-Stieltjes measure associated with some function $p(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$. When $\boldsymbol{\omega}_1 = \boldsymbol{\omega}_2$, this theorem reduces to Bochner's theorem.

Theorem 3.1 shows that learning a nonstationary continuous kernel can be viewed as learning its associated bivariate spectral density $p(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$. Moreover, this formulation enables the design of flexible kernels that allow attention mechanisms to capture input-dependent patterns.

Next, we state that this duality allows us to derive a general closed-form random feature approximation for arbitrary continuous kernels, thereby enabling linear-time attention mechanisms.

Theorem 3.2. *Let $\boldsymbol{\phi}(\mathbf{x})$ be the randomized feature map of $k(\mathbf{x}_1, \mathbf{x}_2)$, defined as follows:*

$$\boldsymbol{\phi}(\mathbf{x}) = \sqrt{\frac{1}{2M}} \begin{bmatrix} \cos\left(\mathbf{x}\boldsymbol{\omega}_1^{(1:M/2)\top}\right) + \cos\left(\mathbf{x}\boldsymbol{\omega}_2^{(1:M/2)\top}\right) \\ \sin\left(\mathbf{x}\boldsymbol{\omega}_1^{(1:M/2)\top}\right) + \sin\left(\mathbf{x}\boldsymbol{\omega}_2^{(1:M/2)\top}\right) \end{bmatrix} \in \mathbb{R}^M, \tag{7}$$

where the spectral frequencies $\{[\boldsymbol{\omega}_1^{(m)}, \boldsymbol{\omega}_2^{(m)}]\}_{m=1}^{M/2} \subset \mathbb{R}^{1\times 2d_x}$ are independent and identically distributed random row vectors drawn from the spectral density $p(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$. Then, the kernel $k(\mathbf{x}_1, \mathbf{x}_2)$ admits the unbiased estimator

$$k(\mathbf{x}_1, \mathbf{x}_2) \;\approx\; \boldsymbol{\phi}(\mathbf{x}_1)^\top \boldsymbol{\phi}(\mathbf{x}_2). \tag{8}$$

Proof. See Appendix A.1 for the full proof. □

Theorem 3.2 converts kernel evaluation into feature inner products, providing the key step toward scalable linear attention.

We next derive a nonstationary attention mechanism based on RFFs, which is both flexible and scalable. The nonstationary attention output at position i is defined as

$$\begin{aligned} \mathrm{Attn}(\mathbf{X})_i &= \frac{\sum_{j=1}^L k(\mathbf{q}_i, \mathbf{k}_j)\, \mathbf{v}_j}{\sum_{j'=1}^L k(\mathbf{q}_i, \mathbf{k}_{j'})} \approx \frac{\sum_{j=1}^L \boldsymbol{\phi}(\mathbf{q}_i)^\top \boldsymbol{\phi}(\mathbf{k}_j)\, \mathbf{v}_j}{\sum_{j'=1}^L \boldsymbol{\phi}(\mathbf{q}_i)^\top \boldsymbol{\phi}(\mathbf{k}_{j'})} \\ &= \frac{\boldsymbol{\phi}(\mathbf{q}_i)^\top \sum_{j=1}^L \left(\boldsymbol{\phi}(\mathbf{k}_j)\, \mathbf{v}_j\right)}{\boldsymbol{\phi}(\mathbf{q}_i)^\top \sum_{j'=1}^L \boldsymbol{\phi}(\mathbf{k}_{j'})}. \end{aligned} \tag{9}$$

Here, $k(\cdot,\cdot)$ denotes the proposed nonstationary kernel and $\boldsymbol{\phi}(\cdot)$ is the random feature map defined in Eq. (7). The last equality follows by exchanging the order of the inner products and summations, which implies the attention can be computed in linear time with respect to the sequence length L.

However, the sine and cosine components within the $\boldsymbol{\phi}$ in Eq. (9) can take both positive and negative values. Under a finite number of random features, this may cause cancellation in the linear-attention denominator $\boldsymbol{\phi}(\mathbf{q}_i)^\top \sum_{j=1}^L \boldsymbol{\phi}(\mathbf{k}_j)$, leading to small or even negative normalization values. As shown in Appendix B.1, increasing the random feature dimension M reduces the measured fraction of negative or relatively small denominators overall, suggesting that this issue can be alleviated with more samples. Nevertheless, near-zero denominators may still occur in practical finite-sample settings, especially when M is limited for computational efficiency. As a simple way to address this issue, we apply a positive feature transformation before computing attention:

$$\widetilde{\boldsymbol{\phi}}(\mathbf{x}) = g(\boldsymbol{\phi}(\mathbf{x})), \qquad g(u) = \begin{cases} u + 1, & u > 0, \\ \exp(u), & u \le 0, \end{cases} \tag{10}$$

where g corresponds to the exponential linear unit (ELU) shifted by one, i.e., $g(u) = \mathrm{ELU}(u) + 1$. This ensures $\widetilde{\boldsymbol{\phi}}(\mathbf{x}) \ge 0$ element-wise. We then use $\widetilde{\boldsymbol{\phi}}$ in place of $\boldsymbol{\phi}$ in Eq. (9), which yields a nonnegative denominator by construction and improves numerical stability. The induced affinity is $\widetilde{k}_\theta(x, y) = \widetilde{\phi}_\theta(x)^\top \widetilde{\phi}_\theta(y)$. Theorem 3.2 applies only to the signed features before ELU+1; it does not establish unbiasedness of the transformed affinity or the normalized attention ratio. Appendix B.1.2 characterizes this transformed kernel and quantifies its difference from the signed-feature similarity.

3.2 Flow-based Spectral Density Modeling

As established in Section 3.1, learning expressive nonstationary attention kernels can be reformulated as learning a bivariate spectral density $p(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$. Prior work such as GMM-RKS [Pal Chowdhury et al., 2022] models the spectral distribution using Gaussian mixture models. Although this improves

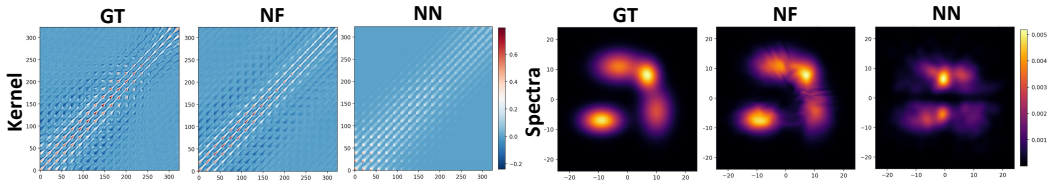


Figure 1: Visualization of kernels and spectral densities $p_(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$ under NF-based and NN-based implicit parameterizations. NF-based spectral modeling recovers more structured spectra and induces more faithful kernels.

expressiveness over fixed kernels, the flexibility and computational cost depend heavily on the number of mixture components, which may limit scalability in practice. IKAN [Song et al., 2021] further generalizes kernelized attention through implicit neural parameterization. However, the absence of an explicit spectral density hinders principled regularization, and may lead to spectral collapse, thereby degrading model performance. As shown in Figure 1, NN-based implicit spectral parameterization may fail to recover the underlying spectral structure, leading to poorly learned kernels.

To address these limitations, we need to design a spectral modeling framework that is expressive and scalable while retaining an explicit and regularizable spectral representation. We therefore introduce a flow-based approach for spectral density learning. Normalizing flows (NFs) define flexible probability distributions by transforming a simple base distribution through a sequence of invertible and differentiable mappings. Formally, we first sample a latent variable

$$\mathbf{z} \sim p_0(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I}), \qquad \mathbf{z} \in \mathbb{R}^{2d_x}, \tag{11}$$

and obtain the spectral pair through an invertible transformation

$$(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) = F_\theta(\mathbf{z}), \qquad \boldsymbol{\omega}_1, \boldsymbol{\omega}_2 \in \mathbb{R}^{d_x}. \tag{12}$$

where F_θ denotes a composition of N_f invertible and differentiable transformations. By the change-of-variables formula, the induced spectral density is given by

$$p_\theta(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) = p_0(\mathbf{z}) \left| \det \left(\frac{\partial F_\theta(\mathbf{z})}{\partial \mathbf{z}} \right) \right|^{-1}. \tag{13}$$

In this work, we parameterize F_θ with RealNVP transformations, which provide an efficient architecture with tractable Jacobian determinants [Dinh et al., 2017]. Therefore, sampling spectral pairs for random feature construction reduces to drawing $\mathbf{z} \sim p_0(\mathbf{z})$ and applying $(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) = F_\theta(\mathbf{z})$.

To avoid overfitting the spectral density, we further introduce a spectral regularizer. Specifically, we regularize the learned spectral density $p_\theta(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$ using the NG-SM spectral density as a reference prior $p_{\mathrm{NGSM}}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$, which is defined as a bivariate Gaussian mixture [Yang et al., 2025]. The regularization term is

$$\mathcal{L}_{\mathrm{spec}} = \mathrm{KL}\left(p_\theta(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) \,\|\, p_{\mathrm{NGSM}}(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2) \right), \tag{14}$$

and is estimated by Monte Carlo samples from $p_\theta(\boldsymbol{\omega}_1, \boldsymbol{\omega}_2)$. This term acts as a soft regularizer that stabilizes spectral learning while keeping the flow parameterization flexible.

Finally, with the flexible and stable flow-based spectral kernel attention (FSKA) introduced above, we further ask *whether the expressiveness of the kernel can simplify the attention architecture itself*. We turn to this question next.

3.3 Simplifying Attention via Kernel Expressiveness

Standard attention uses learnable query, key, and value projections together with a fixed kernel function to model token interactions, resulting in a large number of projection parameters and reduced efficiency in long-sequence settings. We next simplify FSKA by removing the query and key projections and fixing the value projection. Specifically, the learned kernel is expressive enough to model token similarity directly on the input representations, so explicit query and key projections can be removed. For the value branch, we use a simple fixed projection to provide a stable aggregation space, while leaving adaptive token interactions to the learned kernel.

3.3.1 Removing Query and Key Projections

In standard attention, input representations $\mathbf{X}$ are first projected into query and key spaces by learnable matrices, and the attention matrix is determined by a fixed similarity function evaluated on $(\mathbf{q}_i, \mathbf{k}_j)$. In other words, token interactions are jointly determined by the learned projections and the predefined similarity function.

In contrast, our formulation models token interactions directly with a flexible kernel defined on the input space. When the kernel family is sufficiently expressive, the projected similarity can be represented directly as kernel acting on the original input representations $\mathbf{X}$. This enables removing explicit query and key projections while preserving the expressiveness of kernel-based similarity modeling. We formalize this observation below.

Theorem 3.3 (Absorption of Query–Key Projections)**.** *Let $\mathcal{X} \subset \mathbb{R}^{d_x}$ be a compact set. Let $\mathcal{K}$ be a family of functions $\tilde{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that is uniformly dense in $C(\mathcal{X} \times \mathcal{X})$.*

Let $k \in C(\mathcal{U} \times \mathcal{V})$ be any continuous similarity function, where $\mathcal{U}$ and $\mathcal{V}$ are compact sets. Let $\mathbf{W}_Q \in \mathbb{R}^{d_x \times d_q}$ and $\mathbf{W}_K \in \mathbb{R}^{d_x \times d_k}$ be linear mappings such that $\mathcal{X}\mathbf{W}_Q \subset \mathcal{U}$ and $\mathcal{X}\mathbf{W}_K \subset \mathcal{V}$. Define

$$f(\mathbf{x}_1, \mathbf{x}_2) = k(\mathbf{x}_1\mathbf{W}_Q, \mathbf{x}_2\mathbf{W}_K). \tag{15}$$

Then, for any $\epsilon > 0$, there exists $\tilde{k}_\epsilon \in \mathcal{K}$ such that

$$\sup_{\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}} \left| f(\mathbf{x}_1, \mathbf{x}_2) - \tilde{k}_\epsilon(\mathbf{x}_1, \mathbf{x}_2) \right| \leq \epsilon. \tag{16}$$

Proof. See Appendix A.2 for the full proof. □

Theorem 3.3 is conditional on its uniform-density assumption and does not establish that a finite implementation satisfies that assumption. Under this assumption, any similarity function induced by linear projections can be uniformly approximated by a kernel defined directly on the original input representations. This implies that explicit query and key projections are not strictly necessary for modeling pairwise similarities in expressive kernel attention, and introduce additional parameters.

Motivated by this observation, we remove query and key projections and compute attention directly on the input representations:

$$\mathrm{Attn}(\mathbf{X})_i = \frac{\sum_{j=1}^{L} \widetilde{k}_\theta(\mathbf{x}_i, \mathbf{x}_j)\, \mathbf{v}_j}{\sum_{j'=1}^{L} \widetilde{k}_\theta(\mathbf{x}_i, \mathbf{x}_{j'})}. \tag{17}$$

This simplification of the attention mechanism reduces projection-related parameters by shifting token interaction modeling from learned projections to the expressive kernel attention mechanism.

3.3.2 Value Representation with Fixed Projections

After removing query and key projections, the attention weights are computed directly from the input representations through the learned kernel. A natural question is whether the value branch in Eq. (17) can also be simplified by directly using the original token representations as values. However, directly using $\mathbf{X}$ provides only a single value representation space. In a multi-head setting, this may limit representation diversity across heads and lead to redundant outputs, even if different heads apply distinct kernel attention. Therefore, the value branch cannot be removed in the same way as query and key projections.

The reason is that the value branch plays a different role from the query and key branches. Query and key projections affect similarity computation and can therefore be absorbed by a sufficiently expressive learned kernel. By contrast, the value branch defines the feature space in which token information is aggregated. Directly using $\mathbf{X}$ forces all heads to aggregate in the same original representation space, whereas value projections provide different feature subspaces to capture complementary sequence information. However, these value projections do not need to be learnable. Since the learned kernel already adaptively determines the aggregation weights over the values, the value projection mainly provides the feature space in which the selected token information is represented.

Motivated by this observation, we design a non-trainable value projection $\mathbf{W}_V \in \mathbb{R}^{d_x \times d_v}$:

$$\mathrm{Attn}(\mathbf{X})_i = \sum_{j=1}^{L} \alpha_{ij}^{\theta} \mathbf{x}_j \mathbf{W}_V, \qquad \alpha_{ij}^{\theta} = \frac{\widetilde{k}_\theta(\mathbf{x}_i, \mathbf{x}_j)}{\sum_{j'=1}^{L} \widetilde{k}_\theta(\mathbf{x}_i, \mathbf{x}_{j'})}. \tag{18}$$

Here, the positive-feature affinity $\widetilde{k}_\theta$ determines the input-dependent similarity structure and assigns adaptive aggregation weights, while the fixed projection $\mathbf{W}_V$ determines how token information is represented before aggregation.

Finally, we show that FSKA can be simplified with expressive kernel attention by removing the query and key projections and fixing the value projection, since the learned kernel already models token similarities directly while the value projection mainly provides the representation space for aggregation. This simplified design reduces projection-related parameter size while preserving the expressive capacity of the attention mechanism. This formulation also motivates our multi-head design: different heads use head-specific learned kernels to induce diverse input-dependent similarity patterns, together with fixed orthogonal value projections that aggregate information in complementary feature subspaces.

3.4 Multi-Head Kernel Attention

We next extend the proposed kernel attention to the multi-head setting. We assign each head a head-specific learned kernel together with a fixed orthogonal value projection, so that different heads can induce diverse input-dependent similarity patterns while aggregating information in complementary feature subspaces.

Formally, for the h-th head, the attention weights are defined as

$$\alpha_{ij}^{(h)} = \frac{\widetilde{k}_{\theta_h}(\mathbf{x}_i, \mathbf{x}_j)}{\sum_{j'=1}^{L} \widetilde{k}_{\theta_h}(\mathbf{x}_i, \mathbf{x}_{j'})}, \tag{19}$$

where each head's spectral density is parameterized by a normalizing flow as in Section 3.2, and $\widetilde{k}_{\theta_h}$ denotes the affinity after the positive feature transformation in Section 3.1. The corresponding head output is

$$\mathrm{Attn}^{(h)}(\mathbf{X})_i = \sum_{j=1}^{L} \alpha_{ij}^{(h)} \mathbf{x}_j \mathbf{W}_V^{(h)}, \tag{20}$$

where $\mathbf{W}_V^{(h)} \in \mathbb{R}^{d_x \times d_v}$ is a fixed value projection.

To preserve diversity in the value space without introducing learnable value projections, we design the value projections of different heads to be mutually orthogonal:

$$\left(\mathbf{W}_V^{(h)}\right)^\top \mathbf{W}_V^{(h')} = \mathbf{0}, \qquad h \neq h'. \tag{21}$$

This orthogonality encourages different heads to aggregate information in distinct value subspaces, thereby reducing redundancy among head outputs.

The final multi-head output is obtained by concatenating all head outputs:

$$\mathrm{MHAttn}(\mathbf{X})_i = \mathrm{Concat}\big(\mathrm{Attn}^{(1)}(\mathbf{X})_i, \ldots, \mathrm{Attn}^{(H)}(\mathbf{X})_i\big). \tag{22}$$

This design preserves multi-head diversity through two mechanisms: head-specific kernel attention and orthogonal fixed value subspaces. Compared with standard multi-head attention, this design removes explicit query and key projections and avoids learnable value projections, thereby reducing projection-related parameters while preserving diverse token interactions through head-specific learned kernels and complementary value subspaces.

4 Experiments

In this section, we first examine the contribution of each design component in Section 4.1, then report sentence and full-document classification results in Sections 4.2 and 4.3, and finally analyze

Table 1: Ablation study on architectural design choices for attention mechanisms on MNIST classification. We report test accuracy (Acc). ✓ indicates that the corresponding component is used, ✗ indicates that it is removed, and – indicates that the component is not applicable.

Model	QK Proj.	Learnable V	Flexible Kernel	Positive RF	Acc ↑
Transformer	✓	✓	✗	–	0.8556
Transformer w/o QKV	✗	✗	✗	–	0.5267
IKAN	✓	✓	✓	✗	0.8656
Ours (FSKA)	✗	✗	✓	✓	**0.8722**
+ Direct RFF	✗	✗	✓	✗	0.3956
+ Learnable V	✗	✓	✓	✓	0.8725

parameter efficiency and memory scaling in Section 4.4 to evaluate practical efficiency. Additional analyses, including PRF stability and similarity differences, nonstationary regression, hyperparameter sensitivity, and LRA-style benchmark settings, are provided in Appendix B. The new document-classification protocol and supplementary spectral-learning, query–key, and regularization ablations are collected in Appendices B.6 and B.7.

4.1 Ablation Study

We start by conducting an ablation study on MNIST to isolate the effects of the main architectural choices in our attention module. The results are reported in Table 1. Overall, the ablation supports two central claims of our method: FSKA improves model performance by learning expressive nonstationary kernels for attention through a flow-based spectral density parameterization. Moreover, the flexible learned kernel enables a simpler projection design while preserving the expressive capacity of the model.

First, our method achieves the best performance among the compared methods, demonstrating the benefit of explicit flow-based spectral kernel learning. Unlike the standard Transformer, which relies on a fixed softmax kernel, and IKAN, whose kernel is learned implicitly without an explicit spectral density, our method directly parameterizes a bivariate spectral density with normalizing flows. This yields a flexible and regularizable nonstationary kernel attention mechanism, improving the model's ability to capture complex token interactions.

Second, the ablation confirms that an expressive kernel enables a simpler projection design. Removing projection branches from a standard Transformer causes a large performance drop, because the fixed softmax kernel alone is insufficient for adaptive similarity modeling. In contrast, our model removes query–key projections while maintaining strong performance, as the learned kernel directly models token interactions in the input representation space. Moreover, making the value projection learnable brings little additional benefit over the fixed-value variant. These results support our claim that query–key projections become less necessary, and fixed value projections are sufficient, once adaptive routing is governed by a sufficiently expressive kernel.

In addition, PRFs are crucial for stable linearized attention when the number of random features is limited, e.g., $M = 96$. Direct RFF features severely degrade performance because their signed components may cancel in the attention denominator and destabilize normalization. Our positive feature map prevents this issue by ensuring nonnegative features.

4.2 Sentence Classification

We further evaluate the proposed FSKA on sentence-level classification tasks. We conduct experiments on six benchmark datasets, including DBPedia, AGNews, TREC, RT-Polarity, SST-2, and SST-5, covering topic classification, question classification, and sentiment classification.[1] We compare with the standard Transformer, MIKAN, Performer, RKA, GMM-RKS, KPCA, and MetaLA [Vaswani et al., 2017, Song et al., 2021, Choromanski et al., 2021, Peng et al., 2021, Pal Chowdhury et al., 2022, Teo and Nguyen, 2024, Chou et al., 2024], and report F1-macro over five

[1] Details of the datasets and baselines are provided in Appendix B.4.

Table 2: F1-macro (%) on text classification benchmark datasets. Mean and standard deviation are computed over five runs. The best result per column is bolded and the second-best is underlined.

MODEL	Venue	DBPedia	AGNews	TREC	RT-Polarity	SST-2	SST-5
METRIC		F1-macro (↑)					
Transformer	NeurIPS 2017	98.13 ± 0.15	90.43 ± 0.07	$\underline{86.14 \pm 0.93}$	70.52 ± 0.45	76.07 ± 1.10	35.11 ± 1.46
MIKAN	AAAI 2021	97.74 ± 0.01	$\underline{91.15 \pm 0.24}$	85.47 ± 1.07	71.30 ± 0.92	76.04 ± 0.86	32.84 ± 0.39
Performer	ICLR 2021	98.12 ± 0.09	90.23 ± 0.31	85.70 ± 0.57	70.88 ± 0.62	77.24 ± 0.41	35.92 ± 0.96
RKA	ICLR 2021	$\underline{98.26 \pm 0.01}$	90.80 ± 0.28	85.56 ± 0.75	71.15 ± 0.72	77.08 ± 1.46	36.10 ± 0.49
GMM-RKS	TMLR 2022	96.76 ± 0.21	90.58 ± 0.41	85.36 ± 0.91	71.05 ± 1.02	76.81 ± 0.89	36.42 ± 0.84
KPCA	NeurIPS 2024	97.59 ± 0.14	90.20 ± 0.49	85.90 ± 0.96	$\mathbf{72.16 \pm 1.58}$	$\underline{77.62 \pm 1.13}$	$\underline{36.97 \pm 1.36}$
MetaLA	NeurIPS 2024	97.24 ± 0.15	91.14 ± 0.36	85.07 ± 1.02	70.28 ± 1.71	77.03 ± 0.69	35.27 ± 1.41
Ours	–	$\mathbf{98.33 \pm 0.13}$	$\mathbf{91.20 \pm 0.37}$	$\mathbf{86.21 \pm 1.02}$	$\underline{72.08 \pm 1.44}$	$\mathbf{78.74 \pm 1.15}$	$\mathbf{38.16 \pm 1.40}$

runs in Table 2.[2] Overall, FSKA achieves the strongest or highly competitive performance across the six datasets. Notably, FSKA improves the Transformer from 76.07 to 78.74 on SST-2 and from 35.11 to 38.16 on SST-5.

These results demonstrate the benefit of explicit flow-based spectral kernel learning. Unlike the standard Transformer and fixed-form attention variants such as Performer, RKA, KPCA, and MetaLA, whose similarity structures are largely determined by predefined attention forms, FSKA directly learns an explicit bivariate spectral density with NFs. This yields a flexible and regularizable nonstationary kernel, allowing the model to better adapt token interactions to different classification tasks.

Compared with learnable kernel methods such as GMM-RKS and MIKAN, FSKA also shows more consistent performance. GMM-RKS relies on a Gaussian-mixture spectral parameterization, which may suffer from increasingly difficult optimization as the number of mixture components grows. MIKAN learns kernelized attention implicitly, but lacks an explicit and regularizable spectral density representation. In contrast, FSKA combines explicit spectral modeling with the expressiveness of normalizing flows, leading to stronger overall performance across both Accuracy and F1-macro.

4.3 Document Classification with Full Sequences

We extend the evaluation to SCOTUS, BBC News, 20 Newsgroups (20NG), and arXiv. SCOTUS opinions contain approximately 5–9k tokens and arXiv papers average approximately 6k tokens; BBC News and 20NG provide complementary medium-length documents, averaging approximately 380 and 300 words, respectively. Each tokenized document is provided in full to every model, without truncation, chunking, or sliding-window segmentation. In addition to the attention baselines, we include FlashAttention [Dao et al., 2022] and Mamba [Gu and Dao, 2023]. Evaluation details and the available FSKA settings are given in Appendix B.6. Table 3 reports accuracy and macro-averaged F1, as mean $\pm$ standard deviation over five independent runs.

FSKA obtains the highest mean macro-F1 on all four datasets and the highest mean accuracy on BBC News, 20NG, and arXiv. On SCOTUS, Mamba achieves the highest accuracy (59.84%), while FSKA attains a higher macro-F1 (39.68% versus 37.91%). The macro-F1 margins over the strongest competing baseline are 1.77, 1.13, 0.39, and 1.57 percentage points on SCOTUS, BBC News, 20NG, and arXiv, respectively. These results show competitive document classification performance across the evaluated sequence lengths; the reported mean differences alone do not establish statistical significance.

4.4 Parameter Efficiency and Memory Scalability

We further evaluate the practical efficiency of our design by analyzing trainable parameter allocation and training memory growth under an LRA-style benchmark [Tay et al., 2020]. [3] All methods use the same backbone architecture and are measured under identical training conditions and hardware environment, with details provided in Appendix B.4.

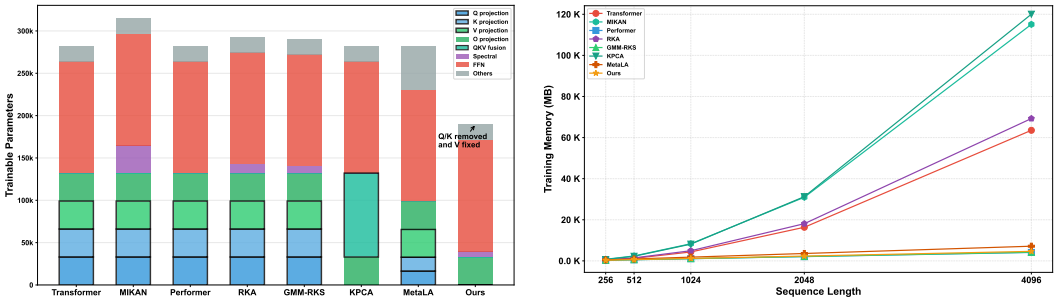
As shown in Fig. 2, FSKA achieves a favorable efficiency–scalability profile. In Fig. 2a, FSKA uses roughly 40%–47% fewer trainable parameters than most baseline attention modules. This reduction

[2] Additional accuracy results are reported in Appendix B.4, Table 6.

[3] Additional details are provided in Appendix B.5.

Table 3: Full-document classification results (%). Values are mean ± standard deviation over five independent runs. Bold denotes the highest mean within each column. F1 denotes macro-F1. Documents are processed in full; BBC News and 20NG are shorter than SCOTUS and arXiv.

Model	SCOTUS		BBC News		20NG		arXiv	
	Acc.	F1	Acc.	F1	Acc.	F1	Acc.	F1
Transformer	55.12±1.24	35.76±0.91	88.43±1.52	88.01±1.57	24.91±0.73	24.18±0.69	75.62±1.61	74.90±1.66
FlashAttention	55.68±0.79	36.21±0.87	88.91±0.49	88.47±0.54	25.46±0.70	24.92±0.67	76.18±0.58	75.41±0.63
Performer	52.38±0.77	31.05±0.88	83.74±1.26	82.96±1.21	25.33±0.81	24.97±0.78	69.81±1.04	68.93±1.09
Mamba	**59.84±0.68**	37.91±0.78	89.34±0.46	88.96±0.51	28.91±0.63	28.47±0.66	76.92±0.54	76.24±0.59
RKA	53.06±0.99	31.72±1.26	85.18±0.48	84.71±0.55	28.62±0.64	28.34±0.67	70.15±1.58	69.42±1.62
GMM-RKS	39.44±0.93	12.31±0.82	66.92±0.75	65.81±0.79	27.13±0.44	23.62±0.31	73.08±0.51	72.47±0.28
KPCA-Scaled	56.37±0.71	37.42±0.83	89.56±0.43	89.18±0.49	22.74±0.68	22.01±0.72	76.84±0.55	76.09±0.60
MIKAN	40.28±0.86	17.05±0.74	84.31±0.59	83.92±0.63	18.46±0.71	17.83±0.68	64.92±0.70	64.15±0.73
FSKA (Ours)	58.91±1.12	**39.68±0.71**	**90.76±0.39**	**90.31±0.44**	**30.24±0.57**	**28.86±0.61**	**78.43±0.48**	**77.81±0.53**



(a) Trainable parameter breakdown by component.

(b) Training memory versus sequence length.

Figure 2: Parameter efficiency and memory scalability under the LRA-style benchmark. Left: different methods allocate trainable parameters differently across projection, spectral, and feed-forward components. Right: training memory growth with increasing sequence length.

mainly comes from eliminating the learnable query and key projections and replacing the learnable value projection with a fixed mapping, while retaining adaptive similarity modeling through the learned nonstationary kernel.

In terms of memory, FSKA follows a low-growth trend close to linear-attention baselines. As shown in Fig. 2b, when the sequence length increases to 4096, FSKA keeps training memory below 10K MB, whereas quadratic-style attention mechanisms grow to around 80K–120K MB. This favorable scaling comes from the RFF-based kernel attention design, which avoids explicitly forming the full quadratic attention matrix.

5 Conclusion

To address the limited flexibility of fixed or stationary attention kernels, we proposed FSKA, a flow-based spectral kernel attention framework for learning expressive nonstationary attention kernels. FSKA parameterizes the spectral density with normalizing flows and regularizes it with a structured spectral prior, enabling flexible kernel learning while remaining compatible with RFF-based linear attention. We further showed that expressive kernel learning can simplify attention by removing query–key projections and using fixed orthogonal value projections. Experiments show that FSKA achieves competitive or improved performance on sentence and full-document classification tasks. It also reduces parameter sizes by approximately 40%–47% compared with most baselines and maintains favorable linear-attention memory scaling on long-sequence benchmarks. Overall, these results highlight FSKA as an expressive and parameter-efficient attention mechanism.

Acknowledgments and Disclosure of Funding

Ying Li acknowledges support from Singapore MOE AcRF Tier 2 grants MOE-T2EP20123-0010 and MOE-T2EP20223-0011.

Alejandro Lancho has received funding from the Comunidad de Madrid's 2023 Cesar Nombela program under Grant Agreement No. 2023-T1/COM-29065. Additionally, he has received funding from the Ministerio de Ciencia, Innovación y Universidades, Spain, under Grant Agreement No. PID2023-148856OA-I00 (funded by MICIU/AEI/10.13039/501100011033 and by ERDF/UE).

Pablo M. Olmos has been supported by the Ministerio de Ciencia, Innovación y Universidades, Spain, under Grant PID2024-157856NB-I00 (CARTESIAN), funded by MCIN/AEI/10.13039/501100011033 and, as appropriate, by "ERDF A way of making Europe"; by the ELLIS Unit Madrid (European Laboratory for Learning and Intelligent Systems); and by the Comunidad de Madrid under grant IND2024/TIC-34728 and the IDEA-CM project (TEC-2024/COM-89).

The contribution of Michael Zhang is funded by the Research Grants Council under the General Research Fund #17312826 and by the University of Hong Kong Seed Fund for PI Research – Basic Research #2402101367.

AI Use Statement

Generative AI tools were used to polish the language of this manuscript.

Reproducibility Statement

Mathematical assumptions and proofs are provided in Appendix A.1 and Appendix A.2. Experimental descriptions, data preparation, configuration details, and supplementary comparisons are collected in Appendix B, including document classification in Appendix B.6. Code Available: `https://github.com/ziyang18/FSKA`.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. In *International Conference on Learning Representations*, 2021.

Yuhong Chou, Man Yao, Kexin Wang, Yuqi Pan, Ruijie Zhu, Yiran Zhong, Yu Qiao, Jibin Wu, Bo Xu, and Guoqi Li. Metala: Unified optimal linear approximation to softmax attention map. *Advances in Neural Information Processing Systems*, 37:71034–71067, 2024.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022. URL `https://arxiv.org/abs/2205.14135`.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. *International Conference on Learning Representations*, 2017.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023. URL `https://arxiv.org/abs/2312.00752`.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR, 2020.

Chun-Liang Li, Wei-Cheng Chang, Youssef Mroueh, Yiming Yang, and Barnabas Poczos. Implicit kernel learning. *Artificial Intelligence and Statistics*, pages 2007–2016, 2019.

Xin Li and Dan Roth. Learning question classifiers. *International Conference on Computational Linguistics*, 2002.

Sankalan Pal Chowdhury, Adamos Solomou, Avinava Dubey, and Mrinmaya Sachan. Learning the transformer kernel. *Transactions on Machine Learning Research*, 2022.

Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *Association for Computational Linguistics*, pages 115–124, 2005.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. Random feature attention. In *International Conference on Learning Representations*, 2021.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in Neural Information Processing Systems*, 20, 2007.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. *Empirical Methods in Natural Language Processing*, pages 1631–1642, 2013.

Kyungwoo Song, Yohan Jung, Dongjun Kim, and Il-Chul Moon. Implicit kernel attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9713–9721, 2021.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006*, 2020.

Rachel S Teo and Tan M Nguyen. Unveiling the hidden structure of self-attention via kernel principal component analysis. *Advances in Neural Information Processing Systems*, 37:101393–101427, 2024.

Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. Transformer dissection: A unified understanding for Transformer's attention via the lens of kernel. 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Zi Yang, Ying Li, Zhidi Lin, Michael Minyi Zhang, and Pablo M Olmos. Multi-view oriented GPLVM: Expressiveness and efficiency. In *Advances in Neural Information Processing Systems*, 2025.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

A Additional Proofs

A.1 Proof of Theorem 3.2

Proof. By Theorem 3.1, the nonstationary kernel admits the spectral representation

$$k(\mathbf{x}_1,\mathbf{x}_2)=\frac{1}{4}\int\Big(\exp\big(i(\mathbf{x}_1\boldsymbol{\omega}_1^\top-\mathbf{x}_2\boldsymbol{\omega}_2^\top)\big)+\exp\big(i(\mathbf{x}_1\boldsymbol{\omega}_2^\top-\mathbf{x}_2\boldsymbol{\omega}_1^\top)\big) \\ +\exp\big(i(\mathbf{x}_1\boldsymbol{\omega}_1^\top-\mathbf{x}_2\boldsymbol{\omega}_1^\top)\big)+\exp\big(i(\mathbf{x}_1\boldsymbol{\omega}_2^\top-\mathbf{x}_2\boldsymbol{\omega}_2^\top)\big)\Big)\,u(\mathrm{d}\boldsymbol{\omega}_1,\mathrm{d}\boldsymbol{\omega}_2). \tag{23}$$

For real-valued kernels, we take the real part of Eq. (23). When u admits a density $p(\boldsymbol{\omega}_1,\boldsymbol{\omega}_2)$, this gives

$$k(\mathbf{x}_1,\mathbf{x}_2)=\frac{1}{4}\mathbb{E}_{p(\boldsymbol{\omega}_1,\boldsymbol{\omega}_2)}\Big[\cos\big(\mathbf{x}_1\boldsymbol{\omega}_1^\top-\mathbf{x}_2\boldsymbol{\omega}_2^\top\big)+\cos\big(\mathbf{x}_1\boldsymbol{\omega}_2^\top-\mathbf{x}_2\boldsymbol{\omega}_1^\top\big) \\ +\cos\big(\mathbf{x}_1\boldsymbol{\omega}_1^\top-\mathbf{x}_2\boldsymbol{\omega}_1^\top\big)+\cos\big(\mathbf{x}_1\boldsymbol{\omega}_2^\top-\mathbf{x}_2\boldsymbol{\omega}_2^\top\big)\Big]. \tag{24}$$

Let $\{(\boldsymbol{\omega}_1^{(m)},\boldsymbol{\omega}_2^{(m)})\}_{m=1}^{M/2}$ be i.i.d. sampled row vectors from $p(\boldsymbol{\omega}_1,\boldsymbol{\omega}_2)$. The corresponding Monte Carlo estimator is

$$\widehat{k}(\mathbf{x}_1,\mathbf{x}_2)=\frac{1}{2M}\sum_{m=1}^{M/2}\Big[\cos\Big(\mathbf{x}_1\boldsymbol{\omega}_1^{(m)\top}-\mathbf{x}_2\boldsymbol{\omega}_2^{(m)\top}\Big)+\cos\Big(\mathbf{x}_1\boldsymbol{\omega}_2^{(m)\top}-\mathbf{x}_2\boldsymbol{\omega}_1^{(m)\top}\Big) \\ +\cos\Big(\mathbf{x}_1\boldsymbol{\omega}_1^{(m)\top}-\mathbf{x}_2\boldsymbol{\omega}_1^{(m)\top}\Big)+\cos\Big(\mathbf{x}_1\boldsymbol{\omega}_2^{(m)\top}-\mathbf{x}_2\boldsymbol{\omega}_2^{(m)\top}\Big)\Big]. \tag{25}$$

Using the trigonometric identity $\cos(a-b)=\cos(a)\cos(b)+\sin(a)\sin(b)$, we can rewrite Eq. (25) as

$$\widehat{k}(\mathbf{x}_1,\mathbf{x}_2)=\frac{1}{2M}\sum_{m=1}^{M/2}\Big[\Big(\cos\Big(\mathbf{x}_1\boldsymbol{\omega}_1^{(m)\top}\Big)+\cos\Big(\mathbf{x}_1\boldsymbol{\omega}_2^{(m)\top}\Big)\Big) \\ \times\Big(\cos\Big(\mathbf{x}_2\boldsymbol{\omega}_1^{(m)\top}\Big)+\cos\Big(\mathbf{x}_2\boldsymbol{\omega}_2^{(m)\top}\Big)\Big) \\ +\Big(\sin\Big(\mathbf{x}_1\boldsymbol{\omega}_1^{(m)\top}\Big)+\sin\Big(\mathbf{x}_1\boldsymbol{\omega}_2^{(m)\top}\Big)\Big) \\ \times\Big(\sin\Big(\mathbf{x}_2\boldsymbol{\omega}_1^{(m)\top}\Big)+\sin\Big(\mathbf{x}_2\boldsymbol{\omega}_2^{(m)\top}\Big)\Big)\Big]. \tag{26}$$

Now define the randomized feature map

$$\boldsymbol{\phi}(\mathbf{x})=\sqrt{\frac{1}{2M}}\begin{bmatrix}\cos\Big(\mathbf{x}\boldsymbol{\omega}_1^{(1:M/2)\top}\Big)+\cos\Big(\mathbf{x}\boldsymbol{\omega}_2^{(1:M/2)\top}\Big)\\ \sin\Big(\mathbf{x}\boldsymbol{\omega}_1^{(1:M/2)\top}\Big)+\sin\Big(\mathbf{x}\boldsymbol{\omega}_2^{(1:M/2)\top}\Big)\end{bmatrix}\in\mathbb{R}^M. \tag{27}$$

Then Eq. (26) can be written compactly as

$$\widehat{k}(\mathbf{x}_1,\mathbf{x}_2)=\boldsymbol{\phi}(\mathbf{x}_1)^\top\boldsymbol{\phi}(\mathbf{x}_2). \tag{28}$$

Finally, since the spectral pairs are sampled independently from $p(\boldsymbol{\omega}_1,\boldsymbol{\omega}_2)$, the Monte Carlo estimator is unbiased:

$$\mathbb{E}\left[\boldsymbol{\phi}(\mathbf{x}_1)^\top\boldsymbol{\phi}(\mathbf{x}_2)\right]=k(\mathbf{x}_1,\mathbf{x}_2). \tag{29}$$

Therefore, the kernel $k(\mathbf{x}_1,\mathbf{x}_2)$ admits the unbiased random-feature estimator $\boldsymbol{\phi}(\mathbf{x}_1)^\top\boldsymbol{\phi}(\mathbf{x}_2)$. □

A.2 Proof of Theorem 3.3

Proof. Since $\mathcal{X}$ is compact and the linear mappings defined by $\mathbf{W}_Q$ and $\mathbf{W}_K$ are continuous, their images under right-multiplication

$$\mathcal{X}\mathbf{W}_Q=\{\mathbf{x}\mathbf{W}_Q:\mathbf{x}\in\mathcal{X}\},\qquad \mathcal{X}\mathbf{W}_K=\{\mathbf{x}\mathbf{W}_K:\mathbf{x}\in\mathcal{X}\}$$

are compact subsets of $\mathcal{U}$ and $\mathcal{V}$, respectively.

Now define the map

$$T:\mathcal{X}\times\mathcal{X}\to\mathcal{U}\times\mathcal{V},\qquad T(\mathbf{x}_1,\mathbf{x}_2)=(\mathbf{x}_1\mathbf{W}_Q,\mathbf{x}_2\mathbf{W}_K).$$

Because the mappings $\mathbf{x} \mapsto \mathbf{x}\mathbf{W}_Q$ and $\mathbf{x} \mapsto \mathbf{x}\mathbf{W}_K$ are continuous, the product map T is also continuous. Since $k \in C(\mathcal{U} \times \mathcal{V})$, the composition

$$f(\mathbf{x}_1, \mathbf{x}_2) = k\big(\mathbf{x}_1\mathbf{W}_Q, \mathbf{x}_2\mathbf{W}_K\big) = (k \circ T)(\mathbf{x}_1, \mathbf{x}_2)$$

is continuous on $\mathcal{X} \times \mathcal{X}$. Hence,

$$f \in C(\mathcal{X} \times \mathcal{X}).$$

By the assumption that $\mathcal{K}$ is uniformly dense in $C(\mathcal{X} \times \mathcal{X})$, for any $\epsilon > 0$ and any function $f \in C(\mathcal{X} \times \mathcal{X})$, there exists $\tilde{k}_\epsilon \in \mathcal{K}$ such that

$$\|f - \tilde{k}_\epsilon\|_\infty = \sup_{\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}} \left| f(\mathbf{x}_1, \mathbf{x}_2) - \tilde{k}_\epsilon(\mathbf{x}_1, \mathbf{x}_2) \right| \leq \epsilon.$$

Substituting the definition of f yields

$$\sup_{\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}} \left| k(\mathbf{x}_1\mathbf{W}_Q, \mathbf{x}_2\mathbf{W}_K) - \tilde{k}_\epsilon(\mathbf{x}_1, \mathbf{x}_2) \right| \leq \epsilon.$$

This proves that the projected similarity function induced by query and key projections can be uniformly approximated by a kernel defined directly on the original input space. □

B Experimental Details and Additional Experiments

All experiments were conducted on a cloud server equipped with 2 NVIDIA Tesla V100 GPUs (16GB memory each), a 10-core Intel Xeon Platinum 81xx series CPU, 64GB RAM, and 200GB storage.

B.1 Additional Analysis on Positive Random Features

We provide an additional analysis of the denominator stability issue in linearized kernel attention. For a query token q_i, the denominator of random-feature attention is given by

$$D_i = \boldsymbol{\phi}(q_i)^\top \sum_{j=1}^{L} \boldsymbol{\phi}(k_j), \tag{30}$$

where $\boldsymbol{\phi}(\cdot)$ denotes the random feature map. Since the sine and cosine components in the direct random Fourier feature construction can take both positive and negative values, finite-sample cancellation may occur in D_i. This can lead to very small or even unstable normalization values, especially when the number of random features M is limited.

To study this effect, we simulate the denominator distribution under different random feature dimensions

$$M \in \{32, 64, 128, 256, 512, 1024, 2048, 4096\}.$$

For each value of M, we compute the linear-attention denominator D_i using direct random Fourier features and collect its empirical distribution over tokens and random trials. The original plotting script measures the fraction of negative or relatively small denominators:

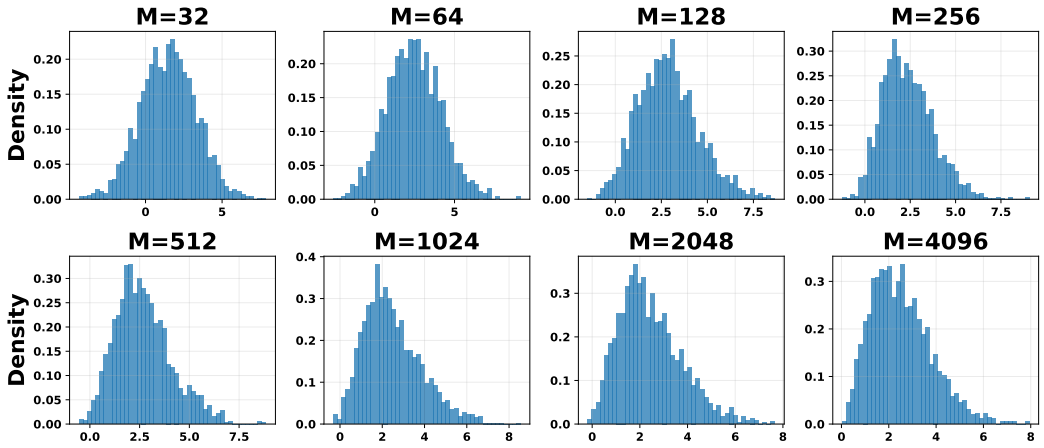
$$\widehat{P}\big(D_i < 0 \text{ or } |D_i| < 10^{-3}\,\mathrm{median}(|D|)\big)\,. \tag{31}$$

The median is computed over batch and token denominators separately for each feature count and method. This statistic differs from an absolute near-zero threshold. The diagnostic uses 32 batches of 64 queries and keys in dimension 16, independently drawn from $\mathcal{N}(0, 0.25I)$, with Gaussian RBF frequencies $\omega \sim \mathcal{N}(0, I)$. It does not train a flow or a Transformer. Here the plotted M counts frequency pairs, yielding $2M$ feature coordinates. We compare direct RFF features with the positive random feature (PRF) transformation used in our model,

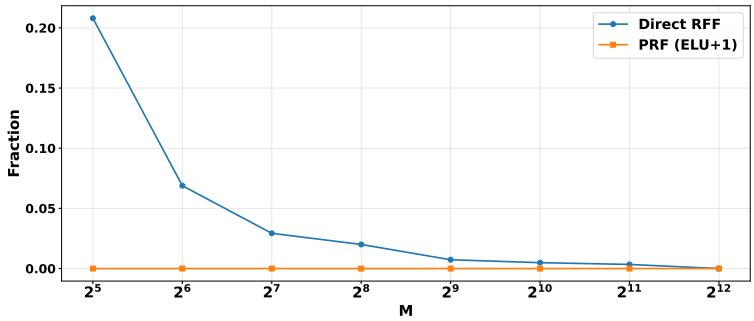
$$\widetilde{\boldsymbol{\phi}}(x) = g(\boldsymbol{\phi}(x)), \qquad g(u) = \begin{cases} u + 1, & u > 0, \\ \exp(u), & u \leq 0, \end{cases} \tag{32}$$

which corresponds to $\mathrm{ELU}(u) + 1$. This transformation ensures $\widetilde{\boldsymbol{\phi}}(x) \geq 0$ element-wise.

Figure 3a shows the empirical denominator distributions for different values of M. When M is small, the denominator distribution of RFFs has non-negligible mass near zero, indicating that the finite



(a) Distribution of the linear-attention denominator of RFFs under different random feature dimensions M.



(b) Fraction of negative or relatively small denominators versus M.

Figure 3: Additional stability analysis of RFFs and PRFs.

random-feature approximation can produce unstable normalization. As M increases, the distribution becomes more concentrated away from zero, suggesting that using more random features reduces the probability of cancellation.

Figure 3b further quantifies this trend by reporting the fraction of negative or relatively small denominators. For direct RFF features, the unstable fraction decreases overall as M increases, but it is not completely eliminated under finite random-feature dimensions. In contrast, PRF keeps this fraction at zero across all tested values of M.

These results motivate the use of PRF in our linear-attention implementation. Although direct RFFs provide an unbiased approximation to the target kernel, their signed feature components can make the denominator numerically unstable. Using a positive feature transformation sacrifices the signed direct feature representation but gives a more stable attention normalization in finite-sample settings. This is particularly important in practice because M cannot be made arbitrarily large due to memory and runtime constraints. Therefore, PRF provides a simple and effective stabilization mechanism for scalable FSKA attention.

B.1.1 Kernel Induced by the Positive Transformation

For a fixed collection of spectral samples Ω, write the implemented signed map as $\phi_{\mathrm{impl},\theta}(x;\Omega)$ and the positive map as

$$z_\theta(x;\Omega) = g(\phi_{\mathrm{impl},\theta}(x;\Omega)), \qquad g(u) = \mathrm{ELU}(u) + 1.$$

The finite-feature kernel actually used in attention is

$$K^{(m)}_{\mathrm{ELU},\theta}(x,y;\Omega) = z_\theta(x;\Omega)^\top z_\theta(y;\Omega), \tag{33}$$

where m counts spectral frequency pairs. This is the same affinity denoted by $\widetilde{k}_\theta$ in the main text. For arbitrary inputs $x_1, \ldots, x_n$ and coefficients $c_1, \ldots, c_n$,

$$\sum_{i,j} c_i c_j K^{(m)}_{\mathrm{ELU},\theta}(x_i, x_j; \Omega) = \left\| \sum_i c_i z_\theta(x_i; \Omega) \right\|_2^2 \geq 0.$$

Thus, it is positive semidefinite for every fixed shared spectral draw. Its population counterpart is

$$\overline{K}^{(m)}_{\mathrm{ELU},\theta}(x, y) = \mathbb{E}_{\Omega \sim p_\theta^{\otimes m}} \left[K^{(m)}_{\mathrm{ELU},\theta}(x, y; \Omega) \right]. \tag{34}$$

The expectation is also positive semidefinite. For a general RealNVP spectral distribution, we do not have a simpler closed-form expression for this expectation. The superscript emphasizes that the feature normalization and therefore this transformed kernel depend on the finite feature count.

Theorem 3.2 proves unbiasedness only for the signed trigonometric feature map in Eq. (7), before any positive transformation and under its stated spectral distribution. It does not imply that $K^{(m)}_{\mathrm{ELU},\theta}$ estimates the same spectral kernel without bias. Nor does it assert unbiasedness of the normalized attention weights. ELU+1 changes the similarity function as well as its numerical behavior; its effect is not equivalent to merely adding a small constant to the denominator.

B.1.2 Similarity Gap After ELU+1

To quantify how the positive transformation changes similarities, we reproduce the diagnostic reported in the author response. It compares the signed Gram matrix $S_{ij} = \phi(x_i)^\top \phi(x_j)$ with $S^+_{ij} = g(\phi(x_i))^\top g(\phi(x_j))$. The associated analysis uses synthetic segmented tokens with varying mean directions and within-segment drift, with 96 tokens, dimension 16, and four segments. This is a feature-map diagnostic without end-to-end training: the reference script samples two Gaussian frequency sets, normalizes tokens and frequency vectors to unit length, and uses the 2π phase convention and the bivariate trigonometric construction before ELU+1. These diagnostic normalizations belong to the synthetic feature-map analysis, which is separate from the trained text-classification experiments.

Both matrices are cosine-normalized:

$$C_{ij} = \frac{S_{ij}}{\sqrt{S_{ii} S_{jj}}}, \qquad C^+_{ij} = \frac{S^+_{ij}}{\sqrt{S^+_{ii} S^+_{jj}}}.$$

The reported metrics are the Pearson correlation of their off-diagonal entries and the mean absolute discrepancy,

$$\rho = \mathrm{Corr}\left((C_{ij})_{i \neq j}, (C^+_{ij})_{i \neq j} \right), \qquad \Delta = \frac{1}{L(L-1)} \sum_{i \neq j} |C_{ij} - C^+_{ij}|.$$

Table 4 retains the six numerical pairs from the author response. The signed matrix is a finite-feature reference, not an exact evaluation of the analytical spectral integral. Consequently, Δ measures disagreement between the two normalized feature similarities, not a direct error against the exact kernel k.

Table 4: ELU+1 similarity gap from the author response. Metrics use off-diagonal token pairs after cosine normalization. The frequency-pair count m is denoted by M in the response and reference analysis.

Frequency pairs m	Correlation ρ	Absolute discrepancy Δ
16	0.921	0.794
32	0.945	0.794
64	0.963	0.800
128	0.978	0.798
256	0.989	0.806
512	0.994	0.802

Correlation increases from 0.921 to 0.994, indicating greater linear alignment between the relative similarity patterns in this diagnostic. However, the absolute discrepancy remains approximately 0.8.

High correlation therefore does not mean equality of similarities, preservation of every pairwise ordering, or convergence to the original spectral kernel. PRF introduces a positive baseline and can compress similarity contrast; the persistent discrepancy is a limitation of interpreting the transformed affinity as the original signed spectral kernel. Its use in FSKA is an explicit modeling and normalization choice, supported here by denominator diagnostics and accompanied by this measured change in similarities.

B.2 Toy Example for Nonstationary Regression

We design a controlled toy benchmark to evaluate whether different attention kernels can model position-dependent nonstationarity under distribution shift. The focus is on simultaneously capturing global trend consistency and local regime transitions.

Data generation. We construct a 1D dataset on a normalized grid

$$t_i = \frac{i-1}{N-1}, \quad i = 1, \dots, N, \quad N = 500, \quad t_i \in [0, 1].$$

The target is

$$y_i = f(t_i) + \varepsilon_i, \qquad \varepsilon_i \sim \mathcal{N}\big(0, \sigma^2(t_i)\big)\,,$$

where $f(\cdot)$ is a smooth position-dependent mixture of regime-specific oscillatory components:

$$f(t) = \sum_{k=1}^{K} w_k(t)\, s_k(t) + m(t), \qquad K = 4.$$

The mixing weights are defined by logistic gates over absolute position:

$$s_1(t) = \sigma(\alpha(t - \tau_1)),\ s_2(t) = \sigma(\alpha(t - \tau_2)),\ s_3(t) = \sigma(\alpha(t - \tau_3)),$$

$$w_0(t) = 1 - s_1(t),\ w_1(t) = s_1(t) - s_2(t),\ w_2(t) = s_2(t) - s_3(t),\ w_3(t) = s_3(t),$$

with $\sigma(u) = 1/(1 + e^{-u})$, $\alpha > 0$, and $\tau_1 < \tau_2 < \tau_3$. Each $s_k(t)$ uses different sinusoidal frequency/phase patterns, $m(t)$ is a position-dependent mean-shift/trend term, and $\sigma^2(t)$ introduces heteroscedastic noise. Inputs include time, sinusoidal/polynomial terms, auxiliary variables, and gate features. We use a 70/30 train-test split with a fixed split.

We compare 9 models: Transformer, IKAN-direct, MIKAN, five text-kernel baselines (KPCAScaled, Performer, RKA, GMM-RKS, MetaLA), and ours FSKA. All models are trained with Adam under the same split and evaluated by test MSE. Figure 4 reports per-model prediction plots and MSE.

Ours FSKA achieves the best test error (MSE $= 0.0334$), slightly outperforming IKAN-direct (0.0338), and substantially improving over Transformer (0.1219), MIKAN (0.1938), and most text-kernel baselines ($\approx 0.18 - 0.23$). Qualitatively, Ours FSKA tracks regime boundaries and high-curvature regions more faithfully while preserving global phase alignment. These observations are consistent with our hypothesis: an expressive NF-parameterized spectral kernel improves adaptation to nonstationary structure, while simplified projection design avoids unnecessary optimization entanglement.

B.3 Hyperparameter Sensitivity

We investigate the sensitivity of our model to three key hyperparameters: the NGSM mixture count Q, the RFF dimension M, and the normalizing flow depth N_f. All experiments are conducted on MNIST, and results are summarized in Table 5.

NGSM mixture count Q. The mixture count Q controls the expressiveness of the NGSM kernel approximation. Performance improves steadily as Q increases from 1 to 10 (0.8483 $\rightarrow$ 0.8743), confirming that a richer mixture provides a more flexible kernel representation. However, increasing Q beyond 10 yields no further gains and even slight degradation (Q=20: 0.8431), suggesting overfitting of the mixture components. Notably, runtime remains nearly constant across all values of Q (75–77s), indicating that the mixture computation is efficiently parallelized and does not constitute a bottleneck.

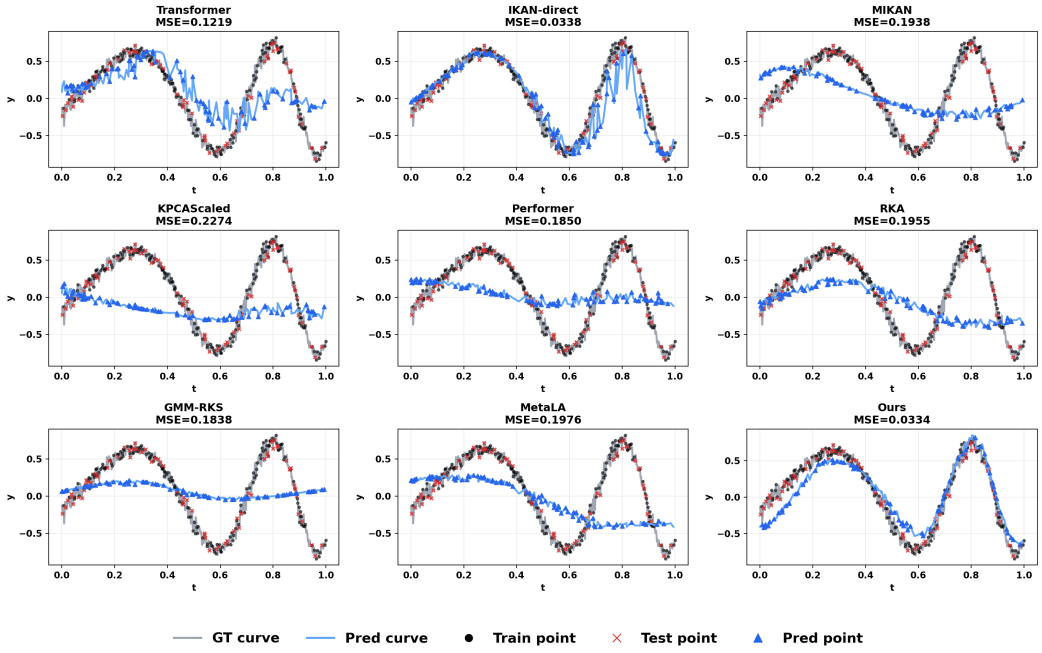


Figure 4: Regression plots on the toy nonstationary example across baselines.

RFF dimension M. The RFF dimension M governs the rank of the random feature approximation to the implicit kernel. Increasing M from 32 to 96 yields a modest accuracy improvement (0.8656 $\rightarrow$ 0.8743) with only a mild runtime increase, demonstrating favorable cost-accuracy scaling. Further increasing to M=128 pushes accuracy to 0.8822 but nearly doubles the runtime (149.20s), as the quadratic cost of the feature map begins to dominate. The default M=96 offers a balanced trade-off between accuracy (0.8743) and efficiency (85.32s).

Normalizing flow depth N_f. The flow depth N_f determines the capacity of the normalizing flow to transform the base distribution into a complex kernel shape. Increasing N_f from 2 to 10 produces a consistent accuracy gain (0.8503 $\rightarrow$ 0.8707), but at a substantial runtime cost that scales approximately linearly ($\sim$13s per additional layer). Beyond N_f=4, the marginal accuracy improvement diminishes while the computational overhead continues to grow. The default N_f=5 strikes an effective balance, achieving near-peak accuracy without excessive overhead.

Overall, The default configuration (Q=10, M=96, N_f=5) consistently achieves strong performance.

B.4 Additional Details for Sentence Classification

B.4.1 Additional Results for Sentence Classification

Table 6 reports additional accuracy results on six sentence classification benchmarks. Overall, FSKA achieves competitive performance across all datasets and obtains the best accuracy on DBPedia, TREC, RT-Polarity, and SST-2, while ranking second on AGNews. Notably, FSKA improves over the standard Transformer from 70.93 to 73.18 on RT-Polarity, and from 75.99 to 77.58 on SST-2. These gains suggest that the proposed flow-based spectral kernel can provide more effective token interactions than standard dot-product attention on sentiment classification tasks.

B.4.2 Baselines and Their Parameter Settings

All models are trained within the same sentence classification benchmark pipeline. Unless otherwise noted, the shared backbone hyperparameters are: embedding dimension 128, hidden dimension 128, 4 attention heads, 2 encoder layers, feed-forward dimension 256, dropout 0.1, maximum sequence length 256, batch size 32, learning rate 10^{-3}, weight decay 10^{-4}, cosine learning-rate scheduling, 5 warm-up epochs, early stopping patience 10, and at most 50 training epochs. Text is lowercased,

Table 5: Parameter sensitivity analysis on MNIST.

Model	**Acc**	**Runtime (s)**
Ours (FSKA: Q=10, M=96, N_f=5)	0.8743	85.32
(a) NGSM Mixture Count Q		
$Q = 1$	0.8483	76.52
$Q = 3$	0.8543	76.30
$Q = 5$	0.8612	76.39
$Q = 20$	0.8431	75.32
$Q = 40$	0.8643	76.82
(b) RFF Dimension M		
$M = 32$	0.8656	69.43
$M = 64$	0.8692	81.70
$M = 128$	0.8822	149.20
(c) Normalizing Flow Depth N_f		
$N_f = 2$	0.8503	64.52
$N_f = 4$	0.8643	90.78
$N_f = 6$	0.8623	119.53
$N_f = 8$	0.8660	146.84
$N_f = 10$	0.8707	173.78

Table 6: Accuracy (%) on text classification benchmark datasets. Mean and standard deviation are computed over five runs. The best result per column is bolded and the second-best is underlined.

MODEL	Venue	DBPedia	AGNews	TREC	RT-Polarity	SST-2	SST-5
METRIC		Accuracy (↑)					
Transformer	NeurIPS 2017	98.13 ± 0.15	90.45 ± 0.07	86.67 ± 0.90	70.93 ± 0.83	75.99 ± 1.06	38.16 ± 0.67
MIKAN	AAAI 2021	97.94 ± 0.01	91.15 ± 0.24	86.60 ± 0.16	71.57 ± 0.38	75.40 ± 0.40	37.33 ± 0.73
Performer	ICLR 2021	98.12 ± 0.09	90.27 ± 0.29	85.00 ± 0.86	70.31 ± 0.42	77.26 ± 0.31	39.44 ± 0.42
RKA	ICLR 2021	98.16 ± 0.01	90.82 ± 0.28	86.80 ± 0.82	72.65 ± 0.82	76.77 ± 0.88	**39.87 ± 0.50**
GMM-RKS	TMLR 2022	96.82 ± 0.18	90.61 ± 0.34	85.74 ± 0.88	71.42 ± 0.96	76.38 ± 0.73	38.72 ± 0.61
KPCA	NeurIPS 2024	97.60 ± 0.10	**91.22 ± 0.39**	85.82 ± 0.72	72.94 ± 1.36	76.72 ± 0.91	39.15 ± 1.12
MetaLA	NeurIPS 2024	97.62 ± 0.11	91.15 ± 0.47	85.31 ± 0.83	72.52 ± 1.56	74.84 ± 0.78	37.31 ± 1.24
Ours	–	**98.23 ± 0.17**	91.20 ± 0.31	**86.87 ± 0.76**	**73.18 ± 1.22**	**77.58 ± 0.95**	39.01 ± 1.21

tokenized with a whitespace tokenizer, and represented with a vocabulary built from the training split only.

Transformer. The Transformer baseline is the standard encoder-style multi-head self-attention model of Vaswani et al. [Vaswani et al., 2017]. In our implementation, each layer contains learned Q, K, V, and output projections, with hidden dimension 128, 4 heads, and per-head dimension 32. We use 2 Transformer encoder layers, feed-forward dimension 256, and dropout 0.1. This model serves as the quadratic-attention reference.

MIKAN. MIKAN is our implicit kernel attention baseline, implemented in the benchmark as a multi-head implicit kernel attention network [Song et al., 2021]. It uses the same backbone dimensions as the Transformer baseline, but replaces softmax attention with the implicit kernel attention module. The attention layer uses $M = 64$ random Fourier features per head.

Performer. Performer is the FAVOR+ linear-attention baseline of Choromanski et al. [Choromanski et al., 2021]. It uses the same encoder backbone as the other models, but replaces quadratic softmax attention with positive orthogonal random feature attention. In our code, the number of Performer random features is not explicitly overridden by the benchmark driver, so the implementation falls back to its internal default $n_{\text{features}} = \lfloor d_{\text{head}} \log d_{\text{head}} \rfloor$. With $d_{\text{head}} = 32$, this yields $n_{\text{features}} = \lfloor 32 \log 32 \rfloor = 110$ random features. The projection matrix is sampled as a Gaussian orthogonal random matrix and is redrawn every 1000 training calls.

RKA. RKA is a random kernel attention baseline with learned random features [Peng et al., 2021]. It shares the same encoder backbone dimensions as the other baselines, but generates Fourier features through a small generator network rather than using fixed random projections. In our implementation, RKA uses 64 random features, i.e., `n_features=64`. Since no explicit `noise_dims` are passed by the benchmark driver, the code uses the default generator dimensions $[d_{\text{head}}, 2d_{\text{head}}, d_{\text{head}}] = [32, 64, 32]$. The generator uses leaky-ReLU hidden activations and a tanh output activation.

GMM-RKS. GMM-RKS is a Gaussian-mixture random kitchen sinks baseline that learns the spectral distribution through per-head Gaussian mixture parameters [Pal Chowdhury et al., 2022]. It uses the same shared encoder backbone hyperparameters as the other baselines. In our implementation, the feature map uses `n_dims=64`, with one learnable mean vector and one learnable full covariance matrix per attention head.

FlashAttention (document classification). FlashAttention is an IO-aware algorithm for exact softmax attention that uses tiling to reduce data transfers between GPU memory levels [Dao et al., 2022]. We include it in the full-document classification comparison in Section 4.3 as an efficient exact-attention baseline.

Mamba (document classification). Mamba is a sequence model based on selective state spaces, whose input-dependent parameters control the propagation of information along the sequence [Gu and Dao, 2023]. It scales linearly with sequence length and provides a non-attention comparison in the full-document classification experiments in Section 4.3.

B.4.3 Datasets, Sources, and Access Links

We evaluate on six standard sentence classification benchmarks spanning sentiment analysis, question classification, news topic classification, ontology classification, and review rating prediction.

RT Polarity. RT Polarity is a binary sentiment classification benchmark built from movie review snippets by Pang and Lee [Pang and Lee, 2005]. The public source is the Cornell Movie Review Data page: https://www.cs.cornell.edu/people/pabo/movie-review-data/. In our pipeline, RT Polarity is split into train/test using an $80/20$ stratified split, and then 10% of the training portion is further held out for validation.

SST-2. SST-2 is the binary sentiment subset of the Stanford Sentiment Treebank, introduced by Socher et al. [Socher et al., 2013]. We load it from Stanford Sentiment Treebank release: https://www-nlp.stanford.edu/sentiment/treebank.html. In our preprocessing code, SST-2 is reconstructed from the original SST sentence-level annotations by mapping sentiment scores to binary labels and discarding neutral examples. We use the official train/dev/test split provided with the SST release.

SST-5. SST-5 is the five-way fine-grained version of the Stanford Sentiment Treebank [Socher et al., 2013]. It is loaded from the same original Stanford Sentiment Treebank source as SST-2: https://www-nlp.stanford.edu/sentiment/treebank.html. Unlike SST-2, it retains all five sentiment levels (very negative, negative, neutral, positive, very positive). We use the official train/dev/test split from the original release.

TREC. TREC is a six-class question classification dataset commonly associated with the setup of Li and Roth [Li and Roth, 2002]. In our benchmark, it is loaded through the HuggingFace `datasets` package https://huggingface.co/docs/datasets using the dataset identifier `trec`.

AG News. AG News is a four-class news topic classification dataset included in the large-scale character-level text classification benchmark of Zhang et al. [Zhang et al., 2015]. In our code, it is loaded through the HuggingFace `datasets` package https://huggingface.co/docs/datasets using the dataset identifier `ag_news`. The benchmark uses the standard train/test split provided by the package and additionally reserves 10% of the training set for validation via a stratified split.

DBpedia. DBpedia ontology classification is a 14-class topic classification benchmark also popularized by Zhang et al. [Zhang et al., 2015]. In our code, it is loaded through the HuggingFace

datasets package https://huggingface.co/docs/datasets using the dataset identifier dbpedia_14. As with AG News, we use the standard train/test split returned by the package and construct a stratified validation split from the training data.

Preprocessing and vocabulary construction. All datasets are lowercased and tokenized with a simple whitespace tokenizer. No punctuation or number stripping is enabled in the default benchmark configuration. A vocabulary is built only from the training split, using <PAD> and <UNK> as special tokens, with maximum vocabulary size $50{,}000$ and minimum token frequency 1. All sequences are truncated or padded to a maximum length of 256 tokens.

B.5 Additional Details of LRA Experiments

LRA-style synthetic benchmark. In addition to the sentence classification datasets above, we also conduct a Long Range Arena (LRA)-style scalability benchmark to evaluate memory usage and throughput under increasing context lengths. We construct synthetic input sequences with varying sequence lengths and measure training/inference memory together with throughput under a controlled architecture configuration. Concretely, for each model, we generate random input tensors $x \in \mathbb{R}^{B\times n\times d}$ with entries sampled from a standard Gaussian distribution, where B is the batch size, n is the sequence length, and d is the model dimension. For training-time memory measurements, we also generate random targets of the same shape and optimize a mean-squared-error objective for several warm-up and measurement steps. We vary the sequence length over $\{256, 512, 1024, 2048, 4096\}$ while keeping the remaining model hyperparameters fixed. This setup is intended to isolate computational scaling behavior rather than downstream task accuracy, and therefore should be interpreted as an LRA-style systems benchmark rather than an evaluation on the official LRA datasets.

B.6 Document Classification Protocol

Tasks and sequence handling. The document classification experiments in Section 4.3 cover legal opinions (SCOTUS), news articles (BBC News), newsgroup posts (20NG), and scientific papers (arXiv). SCOTUS documents contain approximately 5–9k tokens and arXiv papers average approximately 6k tokens. BBC News and 20NG contain shorter documents, averaging approximately 380 and 300 words, respectively; these word counts are distinct from tokenized sequence lengths. Every model receives the full tokenized document, without truncation, chunking, or sliding-window segmentation. The 256-token limit in Appendix B.4 applies only to the sentence classification benchmarks.

Data preparation and splits. SCOTUS is loaded from the scotus subset of LexGLUE, BBC News from SetFit/bbc-news, and arXiv from ccdv/arxiv-classification.[4] 20NG is loaded with sklearn.datasets.fetch_20newsgroups, using the original train/test partitions and removing headers, footers, and quoted text.[5] The preparation pipeline caps the training pool at 4,000 documents and the test set at 2,000 documents; smaller partitions are retained. The loader reserves 10% of the resulting training pool for a stratified validation split, using a fixed partition across model runs. Table 7 summarizes the split sizes obtained from the dataset metadata and this loading procedure.

Table 7: Document classification splits produced by the data preparation pipeline. Training counts exclude the held-out validation subset.

Dataset	Classes	Train	Validation	Test
SCOTUS	13	3,600	400	1,400
BBC News	5	1,102	123	1,000
20NG	20	3,600	400	2,000
arXiv	11	3,600	400	2,000

Tokenization and vocabulary. The shared text preprocessor lowercases text, normalizes whitespace, and tokenizes by whitespace splitting. Punctuation and numbers are retained. The vocabulary

[4] Dataset sources: LexGLUE, BBC News, and arXiv classification.

[5] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_20newsgroups.html

is built on the training split only, with minimum token frequency 2 and a maximum of 50,000 entries including <PAD> and <UNK>. Unseen tokens map to <UNK>; padding is accompanied by an attention mask.

FSKA implementation and training. The supplied FSKA classification configuration uses embedding and hidden dimensions of 128, four attention heads, two encoder layers, a feed-forward dimension of 256, and dropout 0.1. The training implementation uses Adam with learning rate 10^{-3}, weight decay 10^{-4}, batch size 32, gradient-norm clipping at 1.0, and a budget of 50 epochs. Its cosine scheduler uses $T_{\max} = 45$ and is stepped once per completed epoch; the scheduler's configuration offset does not implement a separate learning-rate warm-up phase. Classification is trained with cross-entropy; FSKA additionally uses the spectral penalty weighted by $\lambda_{\text{spec}} = 10^{-3}$, with no regularization warm-up in the default configuration. Validation accuracy is monitored during training, and test accuracy and macro-F1 are evaluated after training.

Models and evaluation. We compare Transformer, FlashAttention, Performer, Mamba, RKA, GMM-RKS, KPCA-Scaled, MIKAN, and FSKA. FSKA uses the positive random features described in Section 3.1, with learned query–key projections removed and fixed orthogonal value projections. Table 3 reports accuracy and macro-averaged F1 in percent, with mean and standard deviation across five independent runs. Accuracy measures the fraction of correctly classified documents; macro-F1 averages the per-class F1 scores with equal weight across classes.

B.7 Additional Ablations on Document Classification

B.7.1 Learning the Bivariate Spectral Density

To separate the contribution of spectral learning from the bivariate feature construction, we replace the learned flow-based spectral density with a fixed bivariate Gaussian. The bivariate nonstationary feature construction, PRF transformation, removal of learned query–key projections, fixed value projection, and remaining experimental settings are retained. Table 8 reports the comparison on BBC News and 20NG. The learned spectrum improves accuracy by 26.56 and 8.44 percentage points, and macro-F1 by 26.81 and 8.46 points, respectively. This comparison supports the contribution of learning the spectrum beyond the fixed bivariate construction evaluated here.

Table 8: Learned versus fixed bivariate spectral density. Accuracy and macro-F1 are reported in percent.

	BBC News		20NG	
Model	Accuracy	Macro-F1	Accuracy	Macro-F1
FSKA	**90.76**	**90.31**	**30.24**	**28.86**
Fixed bivariate Gaussian	64.20	63.50	21.80	20.40

B.7.2 Finite-Capacity Query–Key Ablation

We compare FSKA with and without learnable query–key projections on 20NG while varying the random-feature capacity and flow depth. The default flow depth for these document-classification experiments is $N_f = 3$. The first sweep uses $M \in \{32, 64, 96, 128\}$ with $N_f = 2$; the second uses $N_f \in \{2, 3, 5, 7, 9\}$ with $M = 32$. Table 9 reports the signed accuracy gaps between the two configurations in percentage points. Following the convention that a positive gap is an accuracy cost of removing projections, we write

$$\Delta_{\text{QK}} = \text{Acc}(\text{with Q/K}) - \text{Acc}(\text{without Q/K}).$$

Thus, positive values favor retaining learned projections and negative values favor removing them. Across the tested settings, the gap magnitude is at most 0.90 points. At $N_f = 2$, the gap changes from positive at $M = 32, 64$ to negative at $M = 96, 128$; with $M = 32$, it is negative at each tested depth of at least three layers. The results provide a finite-capacity comparison on a document classification task, separate from the approximation assumptions used in the theoretical analysis.

Table 9: Capacity sweeps on 20NG. Entries are signed Q/K accuracy gaps in percentage points.

Random-feature capacity sweep, fixed $N_f = 2$					
M	32	64	96	128	–
Signed gap	$+0.90$	$+0.81$	-0.50	-0.87	–
Flow-depth sweep, fixed $M = 32$					
N_f	2	3	5	7	9
Signed gap	$+0.90$	-0.51	-0.55	-0.51	-0.47

B.7.3 Spectral Regularization Strength

The spectral regularization weight is $\lambda_{\text{spec}} = 10^{-3}$ by default. We vary this weight over $\{0, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$ on 20NG, as reported in Table 10. Setting the weight to zero removes the spectral regularization penalty while retaining the learned spectral model. Relative to the default, removing the penalty reduces accuracy from 30.24% to 27.42% and macro-F1 from 28.86% to 25.83%. Increasing the weight to 10^{-1} also reduces both metrics. The default gives the highest reported scores among the tested values; the scores at 10^{-4} and 10^{-2} are within 0.66 accuracy points and 0.83 macro-F1 points of the default.

Table 10: Spectral regularization sensitivity on 20NG. Accuracy and macro-F1 are reported in percent.

λ_{spec}	Accuracy	Macro-F1
0	27.42	25.83
10^{-4}	29.58	28.03
10^{-3} (default)	**30.24**	**28.86**
10^{-2}	29.91	28.41
10^{-1}	28.76	27.19

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes].

   Justification: The abstract and introduction summarize the main contributions of FSKA, including flow-based spectral kernel learning, simplified projection design, and empirical evaluation on classification and scalability benchmarks. The claims are supported by the theoretical discussion in Section 3 and the experimental results in Section 4, with additional analyses provided in Appendix B.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: The paper discusses practical limitations of finite random-feature attention in Section 3.1, where signed RFF features may cause unstable normalization when the number of random features is limited. The paper also presents PRF as a stabilization strategy and provides additional empirical analysis in Appendix B.1.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes].

   Justification: The assumptions are stated in the theorem statements. The main theoretical result, Theorem 3.3, is proved in Appendix A.2; other spectral results are stated with references to existing Bochner's theorem.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes].

   Justification: The paper describes the model architecture, baselines, datasets, evaluation metrics, and main training settings. Additional implementation details and dataset preprocessing are provided in Appendix B.4, with further experimental analyses in Appendix B. We also include the implementation code in the supplementary material to facilitate reproducibility.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes].

   Justification: We provide anonymized implementation code in the supplementary material, together with instructions for reproducing the main experiments. The datasets used in our experiments are publicly available, and their sources and preprocessing details are described in Appendix B.4.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes].

   Justification: The paper reports the main experimental settings, including datasets, baselines, and hyperparameter settings. Additional details are provided in Appendix B.4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes].

   Justification: For the main sentence and document classification experiments, we report mean and standard deviation over five runs.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes].

   Justification: We report the hardware environment for the main experiments, and provide memory and throughput comparisons in the scalability study. Additional implementation details are provided in Appendix B.4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes].

   Justification: The research uses standard public benchmark datasets and does not involve human-subject data collection, private data, or deployed decision-making systems. We preserve anonymity in the submission and follow the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A].

    Justification: This work is a foundational study on attention mechanisms and kernel learning, and does not introduce a deployed system, collect new user data, or target a specific societal application.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A].

    Justification: The paper does not release high-risk models, scraped datasets, or systems intended for deployment. The released code is for reproducing the proposed attention mechanism and benchmark experiments, so no additional safeguards are required.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes].

Justification: We use publicly available benchmark datasets and properly cite their original sources. Dataset sources and access information are provided in Appendix B.4, and all assets are used only for academic benchmarking.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes].

Justification: We provide anonymized implementation code for the proposed method in the supplementary material. The code is accompanied by documentation and instructions for reproducing the main experiments.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A].

Justification: The paper does not involve crowdsourcing experiments or research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A].

    Justification: The paper does not involve crowdsourcing experiments or research with human subjects, so IRB approval is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A].

    Justification: The core method development in this work does not involve LLMs as an important, original, or non-standard component.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.