TESLA: Taylor Expansion of Sinusoidal Learnable Activations

Daehwa Ko **Jaehyeon Kim** **Seunghyun Ham** **Jay Hoon Jung**
Korea Aerospace University, Goyang, Republic of Korea

Abstract

The parity problem—deciding whether the number of ones in a binary vector is odd or even—remains challenging for standard neural networks due to linear inseparability and the need for global interactions. We propose TESLA, an activation defined as a learnable combination of sine and cosine terms, enabling explicit control over polynomial degree and selective amplification of high-order components. Theoretically, we show that constraining TESLA's coefficients yields Lipschitz/Rademacher complexity bounds and shapes the training dynamics to emphasize higher-frequency structure. Empirically, on parity with input length $n = 32$, TESLA attains strong generalization with 100K training samples ($\approx 0.002\%$ of the 2^{32} input space) and remains robust under heavy corruption, retaining high accuracy with up to 30% label noise. We also compare against periodic and frequency-based baselines (SIREN, SNAKE, and Fourier feature embeddings) on parity and Forrelation. Beyond synthetic structure, TESLA delivers comparable performance on ImageNet-100, indicating that activation-level degree control transfers to more general vision workloads. Code: `https://github.com/KAU-QuantumAILab/TESLA`

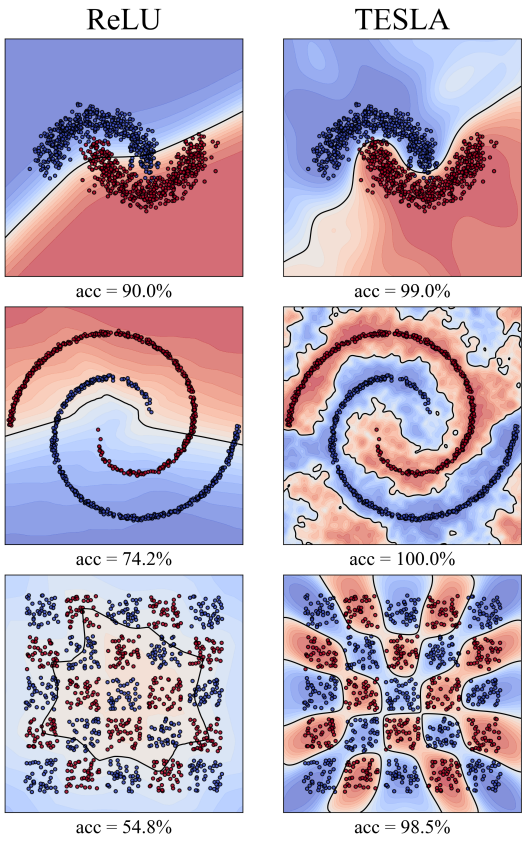


Figure 1: Visualization of the decision boundaries for TESLA($K = 8$) and baseline activations on a 2D feature space. The solid lines separate the predicted classes.

1 INTRODUCTION

Modern neural networks largely build complex functions by stacking local nonlinearities such as ReLU (Nair and Hinton, 2010) and GeLU (Hendrycks and Gimpel, 2016) with linear maps. These designs excel at capturing local structure in images and language, but they can be inefficient for global or long-range interaction tasks that depend on high-order combinations of many input coordinates, global parity/phase, or other global symmetries (Daniely and Malach, 2020). Empirically and theoretically, traditional activation functions and vanilla MLPs exhibit a spectral bias toward low-frequency (low-degree) components, so representing strong high-order or global structure often requires substantially greater depth, width, or very large weight norms (Rahaman et al., 2019; Zhi-Qin John Xu et al., 2020; Tancik et al., 2020;

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

Sitzmann et al., 2020).

To enable direct and learnable control over the polynomial order at the activation level, we propose a simple parametric activation function based on a finite sine and cosine basis. This formulation allows the network to learn Fourier-like coefficients that explicitly control the effective polynomial order of activations. We refer to this activation as Taylor Expansion of Sinusoidal Learnable Activation (TESLA). Our main contributions are as follows:

- We derive tight Lipschitz bounds, establish Rademacher complexity–based generalization guarantees, and characterize learning dynamics, showing that TESLA's inductive bias naturally favors recovery of low- and mid-frequency structure.
- We present constructive approximation results for ridge- and interaction-type functions, together with complementary lower bounds showing that standard piecewise-linear activations require substantially more resources to match TESLA's ability to represent oscillatory structure.
- On synthetic benchmarks, TESLA generalizes a 32-bit parity task with **100K training samples** and remains robust under label noise up to 30%. It also improves performance on Forrelation and LPN tasks and remains usable in ImageNet-100-scale models while incurring negligible throughput overhead ($\approx 1\%$).

2 RELATED WORK

Recent research has explored various ways to enrich neural network representations through activation design. Several works, such as PReLU (He et al., 2015) and APL (Agostinelli et al., 2015), introduce parametric nonlinear activations with a small number of learnable parameters to improve expressiveness and facilitate optimization. While these methods directly modify the pointwise nonlinearity, they do not provide explicit mechanisms for controlling polynomial or spectral degree. In contrast, approaches that learn per-edge or per-unit activations—such as KANs (Liu et al., 2024) and spline-based methods (Apicella et al., 2021; Scardapane et al., 2019)—offer substantially greater expressivity and can closely approximate complex mathematical or scientific functions. However, prior studies and surveys have noted that this increased flexibility can lead to overfitting or instability in low-data regimes unless it is carefully regulated through techniques such as regularization or parameter sharing.

Building on these insights, a complementary line of work employs sinusoidal and Fourier-based parameterizations to address the spectral limitations of standard activations and embeddings. These approaches aim to expand the spectral capacity of neural networks and reshape their training dynamics, motivating the development of periodic activations and Fourier feature mappings. Periodic activations (e.g., SIREN (Sitzmann et al., 2020)) and positional or Fourier feature encodings (Tancik et al., 2020; Rahimi and Recht, 2007) mitigate spectral bias by introducing high-frequency components into the network—either by expanding its effective spectral support or by modifying the spectrum of the initial Neural Tangent Kernel (NTK, (Jacot et al., 2018)). The NTK is a theoretical construct that characterizes the training dynamics of infinitely wide neural networks, showing that such networks evolve like linear models governed by a fixed kernel, which in turn guarantees convergence to a global minimum.

In parallel, a substantial body of theoretical work has emerged to analyze spectral bias (or the frequency principle) and NTK dynamics, aiming to explain why standard networks prioritize low-frequency components and how modifying kernels or embeddings can alter this behavior (Rahaman et al., 2019; Zhi-Qin John Xu et al., 2020; Jacot et al., 2018). Classical approximation-theoretic results—such as Barron-type bounds (Barron, 1993, 1994), depth–width separation theorems (Eldan and Shamir, 2016; Telgarsky, 2016), and lower bounds for piecewise-linear approximations (Yarotsky, 2017; Telgarsky, 2016)—provide a rigorous framework for comparing different parameterizations. These results motivate our constructive ridge-approximation upper bounds and lower-bound sketches, which characterize when piecewise-linear activations incur high resource costs to emulate oscillatory global features (Barron, 1993; Telgarsky, 2016).

3 PROPOSED METHOD

3.1 Activation Function Definition

We define a parametric activation function, shared across all neurons within a layer, built from a finite sine and cosine basis:

$$\phi_K(z) \;=\; \sum_{k=1}^{K} \left(\frac{a_k}{k} \sin(kz) + \frac{b_k}{k} \cos(kz) \right), \qquad (1)$$

where $K \in \mathbb{N}$ controls the maximum frequency, and $\{a_k, b_k\}_{k=1}^{K}$ are learnable scalar coefficients. For a neuron receiving input $z = v^\top x$, the neuron output is $f(x) = \phi_K(v^\top x)$. We also define the coefficient bud-

get:

$$A_K \;:=\; \sum_{k=1}^{K} \big(|a_k| + |b_k|\big), \tag{2}$$

which appears in stability and complexity bounds.

3.2 Taylor (Maclaurin) Expansion and Polynomial Coefficients

Expanding ϕ_K via the Taylor series of sine and cosine gives:

$$\phi_K(z) = \sum_{k=1}^{K} \frac{b_k}{k} + \sum_{m=0}^{\infty} \left(\frac{(-1)^m}{(2m+1)!} \sum_{k=1}^{K} a_k k^{2m} \right) z^{2m+1} + \sum_{m=1}^{\infty} \left(\frac{(-1)^m}{(2m)!} \sum_{k=1}^{K} b_k k^{2m-1} \right) z^{2m}. \tag{3}$$

Eq. (3) shows that each polynomial order is an explicit linear functional of the learned coefficients $\{a_k, b_k\}$, giving direct control over degree components at the activation level.

Interpretation vs. Implementation. TESLA is implemented as the finite trigonometric expansion in Eq. (1); we do not evaluate a polynomial series during the forward pass. We use Eq. (3) as an interpretation that links learned trigonometric coefficients to effective polynomial degree and motivates the coefficient budget in our analysis.

Input Spectralization vs. Activation Spectralization. Input-side spectralization (e.g., Fourier features; Tancik et al., 2020) maps inputs to high-frequency coordinates and leaves mode selection to later layers. TESLA instead spectralizes the activation itself: coefficients $\{a_k, b_k\}$ directly shape Maclaurin-order terms, enabling selective amplification or attenuation of target orders.

In practice, this design reduces the need for handcrafted frequency mappings at the input stage and moves spectral control into a small set of learnable activation coefficients. As a result, practitioners can tune inductive bias with fewer architectural changes while preserving compatibility with standard training pipelines. This makes TESLA particularly attractive when transferring across tasks with different frequency characteristics.

4 THEORETICAL ANALYSIS

In this section, we provide a concise theoretical foundation for TESLA. We first bound derivatives and Lipschitz constants to show how the per-harmonic ℓ_1 budget A_K controls gradient magnitudes. We then present Rademacher-complexity generalization bounds, analyze NTK-style mode-wise learning dynamics, and derive a practical rule for selecting K as a function of the effective order m_{eff}, sample size N, and budget A_K.

4.1 Derivative Bound and Stability

Differentiation of Eq. (1) gives:

$$\phi'_K(z) \;=\; \sum_{k=1}^{K} \big(a_k \cos(kz) - b_k \sin(kz)\big).$$

Since $|\sin(\cdot)| \le 1$ and $|\cos(\cdot)| \le 1$, each term is bounded by its amplitude, $|a_k \cos(kz) - b_k \sin(kz)| \le \sqrt{a_k^2 + b_k^2}$; therefore,

$$\|\phi'_K\|_\infty \;\le\; \sum_{k=1}^{K} \sqrt{a_k^2 + b_k^2} \;\le\; \sum_{k=1}^{K} (|a_k| + |b_k|) \;=\; A_K.$$

Composition Lipschitz Bounds. Let $h(x) = \phi_K(Wx)$, where ϕ_K is applied elementwise. By the chain rule,

$$\mathrm{Lip}(h) \le \|\phi'_K\|_\infty \, \|W\|_{\text{op}} \le A_K \, \|W\|_{\text{op}}.$$

Here, $\|\cdot\|_{\text{op}}$ denotes the spectral norm, and $\|\cdot\|_\infty$ denotes the L_∞ norm.

Network-level Lipschitz and Degree Control. For an L-layer network

$$f(x) = W_L \phi_K(\cdots \phi_K(W_1 x)), \qquad \|\phi'_K\|_\infty \le A_K,$$

the Lipschitz constant satisfies

$$\mathrm{Lip}(f) \le \left(\prod_{\ell=1}^{L} \|W_\ell\|_{\text{op}} \right) A_K^{L-1}.$$

Beyond Lipschitz control, the same composition also governs the growth of polynomial complexity: under the Maclaurin-coefficient view induced by Eq. (3), composing $L-1$ TESLA layers activates terms up to order $O(K^{L-1})$. Moreover, the ℓ_1 norm of the corresponding coefficient vector is bounded as

$$\|\mathrm{coef}(f)\|_1 \le A_K^{L-1} \prod_{\ell=1}^{L} \|W_\ell\|_{1\to 1}.$$

Thus, a single per-layer budget A_K stabilizes gradients, while K controls the accessible frequency range. In particular, for an L_{loss}-Lipschitz loss $\mathcal{L}$,

$$\|\nabla_{W_\ell} \mathcal{L}(f(x), y)\|_F \le L_{\text{loss}} \|x\|_2 \left(\prod_{j=1}^{L} \|W_j\|_{\text{op}} \right) A_K^{L-1}.$$

This highlights the trade-off: increasing K expands representable high frequencies, while controlling A_K keeps optimization stable.

4.2 Rademacher Complexity and Generalization

Theorem 1. *Assume $\|x_i\|_2 \le R$ for all $i = 1, \dots, N$. Define*

$$\mathcal{F}_K = \{\, x \mapsto \phi_K(v^\top x) \mid \|v\|_2 \le W,\ A_K \le A \,\},$$

and

$$B_0(A) := \sup_{\phi_K:\, A_K \le A} |\phi_K(0)|.$$

Then, for any sample of size N, the empirical Rademacher complexity satisfies

$$\hat{\mathfrak{R}}_N(\mathcal{F}_K) \le \frac{AWR}{\sqrt{N}} + \frac{B_0(A)}{\sqrt{N}} = \frac{AWR + B_0(A)}{\sqrt{N}}.$$

Consequently, for any L_{loss}-Lipschitz loss $\mathcal{L}$, the generalization gap scales as $O\Big(L_{\text{loss}}(AWR + B_0(A))/\sqrt{N}\Big)$.

The bound recovers the familiar linear-class rate $\Theta(WR/\sqrt{N})$ up to the multiplicative factor A due to the activation Lipschitz constant. In deep architectures, the same effect accumulates multiplicatively via layer-wise operator norms and activation budgets.

Proof. Let

$$\mathcal{F}_{\text{lin}} := \{x \mapsto v^\top x : \|v\|_2 \le W\}.$$

By a standard argument,

$$\hat{\mathfrak{R}}_N(\mathcal{F}_{\text{lin}}) = \frac{1}{N}\,\mathbb{E}\Big[\sup_{\|v\|_2 \le W} \sum_{i=1}^{N} \sigma_i v^\top x_i\Big] \le \frac{WR}{\sqrt{N}}.$$

For any ϕ_K with $A_K \le A$, write $\phi_K(t) = \tilde{\phi}_K(t) + \phi_K(0)$ with $\tilde{\phi}_K(0) = 0$ and $\mathrm{Lip}(\tilde{\phi}_K) \le A$. Also, since $\phi_K(0) = \sum_{k=1}^{K} \frac{b_k}{k}$ and $A_K \le A$, we have $B_0(A) \le A$. By the Ledoux–Talagrand contraction lemma (Ledoux and Talagrand, 2013),

$$\hat{\mathfrak{R}}_N\big(\{\tilde{\phi}_K(v^\top x)\}\big) \le A\,\hat{\mathfrak{R}}_N(\mathcal{F}_{\text{lin}}) \le \frac{AWR}{\sqrt{N}}.$$

For the constants,

$$\begin{aligned}\sup_{|c| \le B_0(A)} \frac{1}{N}\,\mathbb{E}\Big|\sum_{i=1}^{N} \sigma_i c\Big| &= \frac{B_0(A)}{N}\,\mathbb{E}\Big|\sum_{i=1}^{N} \sigma_i\Big| \\ &\le \frac{B_0(A)}{N}\sqrt{\mathbb{E}\Big(\sum_{i=1}^{N} \sigma_i\Big)^2} \\ &= \frac{B_0(A)}{\sqrt{N}}.\end{aligned}$$

Combine the two parts to get the stated bound. □

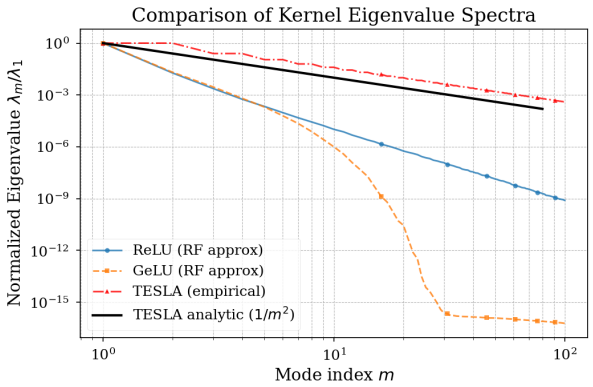


Figure 2: Mode-wise empirical spectra: normalized eigenvalues $\tilde{\lambda}_m = \lambda_m/\lambda_1$ vs. mode index m. Lines: TESLA analytic ($\frac{1}{2m^2}$), TESLA empirical, RF-ReLU, RF-GeLU. RF denotes finite random-feature approximations of the infinite-width ReLU/GeLU kernels.

4.3 Mode-Wise Learning Dynamics

We analyze learning dynamics on the circle $\mathbb{T} = [0, 2\pi)$ in the Fourier basis, where a *mode* refers to a trigonometric component (e.g., $\sin(mt)$ or $\cos(mt)$) with frequency index m. We equip $\mathbb{T}$ with the inner product

$$\langle f, g\rangle = \frac{1}{2\pi}\int_0^{2\pi} f(t)g(t)\,dt.$$

Under squared loss, the linearized gradient flow around initialization is

$$\partial_\tau(f_\tau - f^\star) = -\eta K_\phi(f_\tau - f^\star),$$

where $\eta > 0$ is the learning-rate scale and K_ϕ is the kernel operator

$$\begin{aligned}K_\phi(t,t') &= \nabla_\theta f_\theta(t)^\top \nabla_\theta f_\theta(t'), \\ (K_\phi g)(t) &= \int_0^{2\pi} K_\phi(t,t')g(t')\,\frac{dt'}{2\pi}.\end{aligned}$$

With TESLA coefficients $\{a_k, b_k\}_{k=1}^{K}$, the Jacobian features satisfy

$$\frac{\partial f_\theta}{\partial a_k}(t) = \frac{1}{k}\sin(kt), \qquad \frac{\partial f_\theta}{\partial b_k}(t) = \frac{1}{k}\cos(kt),$$

so K_ϕ is translation-invariant with kernel

$$K_\phi(t,t') = \sum_{k=1}^{K} \frac{1}{k^2}\cos\big(k(t-t')\big).$$

The orthonormal Fourier basis $\psi_{m,s}(t) = \sqrt{2}\sin(mt)$ and $\psi_{m,c}(t) = \sqrt{2}\cos(mt)$ diagonalizes K_ϕ:

$$\begin{aligned}K_\phi[\psi_{m,a}] &= \lambda_m^{(\phi)}\psi_{m,a}, \\ \lambda_m^{(\phi)} &= \tfrac{1}{2m^2}, \quad a \in \{s,c\},\ 1 \le m \le K.\end{aligned}$$

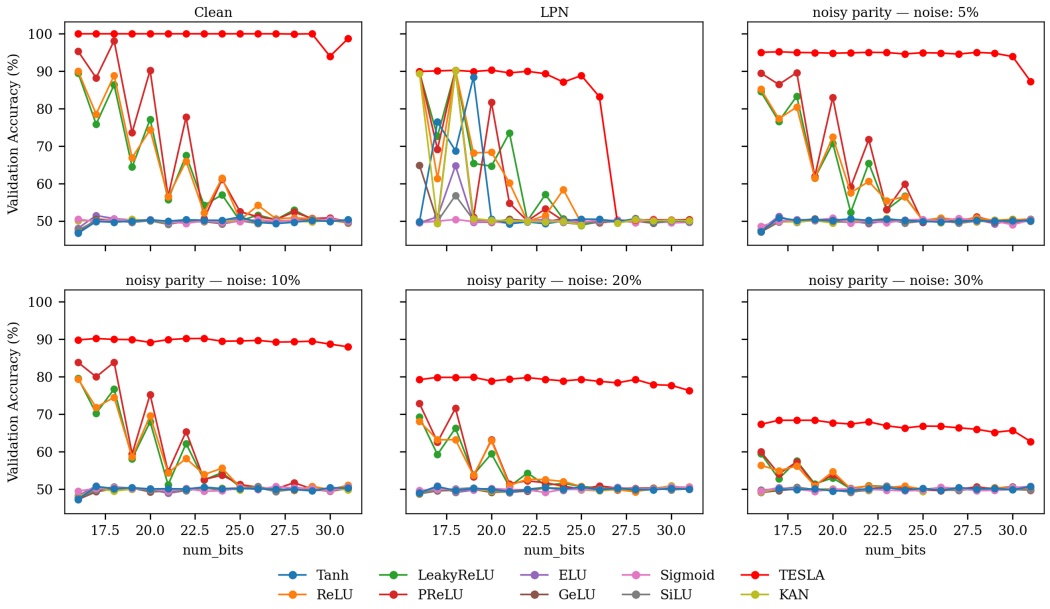


Figure 3: Validation accuracy (%) for the parity task. Each panel shows a different noise condition: Clean, LPN (noise level = 10%) and noisy-parity with 5%, 10%, 20%, 30% label noise. While most baseline activations remain near chance (≈ 50%), TESLA maintains substantially higher accuracy across bit-lengths and noise levels.

and $\lambda_m^{(\phi)} = 0$ for $m > K$. Writing the modal coefficients

$$e_{m,a}(\tau) = \langle f_\tau - f^\star, \psi_{m,a} \rangle,$$

the dynamics decouple as

$$e_{m,a}(\tau) = e_{m,a}(0) \exp\{-\eta \lambda_m^{(\phi)} \tau\},$$
$$\|f_\tau - f^\star\|_{L^2}^2 = \sum_{m,a} e_{m,a}(\tau)^2.$$

Thus, larger $\lambda_m^{(\phi)}$ yields faster decay of the corresponding mode. Analytically, TESLA gives $\lambda_m^{(\phi)} \propto m^{-2}$; empirically (Fig. 2), its spectrum is flatter than RF proxies, allocating relatively larger eigenvalues to medium–high modes and accelerating recovery of higher-order structure.

4.4 Theory-Guided Choice of the Harmonic Count K

We give a compact, operational rule for how the harmonic count K should scale with task complexity and sample size by balancing approximation and estimation. Throughout this subsection, N denotes the training sample size and $f^\star$ the target. For approximation statements, we consider Boolean targets of the form $f^\star : \{0,1\}^d \to \{\pm 1\}$, while using a continuous periodic proxy $f^\star : \mathbb{T} \to \mathbb{R}$ to interpret mode-wise NTK behavior. Constants $c, C, C_0, \ldots$ are absolute and may change value between displays.

Task Complexity via Effective Order. Write the Walsh expansion (Walsh, 1923)

$$f^\star(x) = \sum_{T \subseteq [d]} \hat{f}_T \, \chi_T(x).$$

For $\varepsilon \in (0, \frac{1}{2})$, define the effective interaction order

$$m_{\text{eff}} := \inf \Big\{ m \in \{0, \ldots, d\} \ : \ \sum_{|T| \le m} \hat{f}_T^2 \ge 1 - \varepsilon \Big\}.$$

m_{eff} is the interaction order needed to capture most of the target energy (e.g., parity of order m has $m_{\text{eff}} = m$).

Approximation Behavior of K-harmonic Activations. Using Eq. (1), let the atoms $\{\sigma_k\}$ be normalized by $\|\sigma_k\|_\infty \le 1$ with $1/k$ scaling. Assume the best-in-class L^2 approximation error over $\mathcal{F}_K$ satisfies

$$\inf_{f \in \mathcal{F}_K} \mathbb{E}\big[(f(x) - f^\star(x))^2\big] \ \le \ B\Big(\frac{K}{m_{\text{eff}}}\Big),$$
$$B(z) \in \{c_1 e^{-\gamma z}, \ c_1 z^{-p}\}, \qquad (4)$$

for some $c_1, \gamma, p > 0$; i.e., once $K = \Theta(m_{\text{eff}})$, the approximation error decays monotonically.

Estimation Term for ℓ_1-Budgeted Mixtures. Let $\hat{\mathfrak{R}}_N$ denote empirical Rademacher complexity on N samples. By the ℓ_1-budgeted mixture bound (budget A_K),

$$\hat{\mathfrak{R}}_N(\mathcal{F}_K) \ \le \ C_0 \, \frac{A_K}{\sqrt{N}} \sqrt{\log(2K)}. \qquad (5)$$

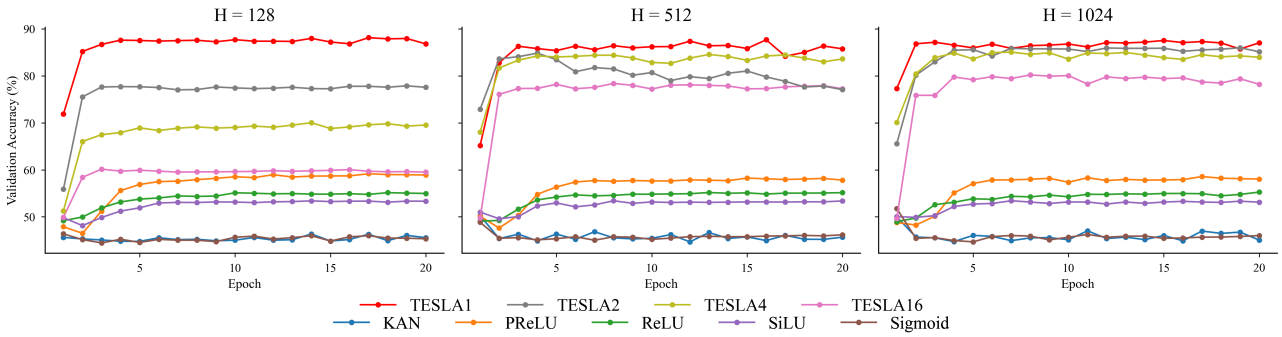


Figure 4: Validation accuracy (%) for the Forrelation task with $d = 12$ using a two-layer MLP. Each panel shows a different hidden size $H \in \{128, 512, 1024\}$. Curves compare TESLA with $K \in \{1, 2, 4, 16\}$ against standard activations. TESLA converges faster and achieves higher accuracy on this global-correlation task.

Consequently, for any 1-Lipschitz surrogate loss $\mathcal{L}$, with probability at least $1 - \delta$,

$$\sup_{f \in \mathcal{F}_K} \left| \mathbb{E}[\mathcal{L}(f(x), y)] - \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(f(x_i), y_i) \right| \leq C_1 \frac{A_K}{\sqrt{N}} \sqrt{\log(2K)} + C_2 \sqrt{\frac{\log(1/\delta)}{N}}. \quad (6)$$

Excess-risk Bound and Optimal Scale. Let $\hat{f}_K \in \mathcal{F}_K$ minimize empirical surrogate risk. Combining Eq. (4)–(6) yields

$$\mathcal{E}(\hat{f}_K) \lesssim \underbrace{B\left(\frac{K}{m_{\text{eff}}}\right)}_{\text{approximation}} + \underbrace{C \frac{A_K}{\sqrt{N}} \sqrt{\log(2K)}}_{\text{estimation}} + C \sqrt{\frac{\log(1/\delta)}{N}}. \quad (7)$$

Balancing the first two terms gives

$$K^\star = \Theta\big(m_{\text{eff}}\big) \cdot \kappa(N, A_K),$$

where κ varies slowly with N and A_K: for $B(z) = c_1 e^{-\gamma z}$, $\kappa = \Theta(\log N)$; for $B(z) = c_1 z^{-p}$, $\kappa = \Theta\big(N^{1/(2p)} (\log N)^{1/(2p)}\big)$ up to constants. Intuitively, taking K much larger than m_{eff} dilutes the per-harmonic budget (average amplitude $\sim A_K/K$) and increases estimation error.

In d-bit Parity, $m_{\text{eff}} = d$, hence the theory predicts $K^\star = \Theta(d)$ up to slowly varying factors. Empirically (Sec. 5), we find a stable range $\kappa \in [0.2, 0.4]$ at $N = 10^5$, placing the optimum near $K \approx d/4$. This aligns with the NTK view: the analytic TESLA kernel on $\mathbb{T}$ has eigenvalues $\lambda_m^{(\phi)} = \frac{1}{2m^2}$, but finite-feature and finite-sample effects flatten the empirical spectrum and emphasize medium–high modes. Thus, once $K \gtrsim m_{\text{eff}}$, marginal approximation gains are offset by the estimation term in Eq. (7).

5 EXPERIMENTS

5.1 Comparisons with Periodic and Frequency-based Baselines

To provide a balanced comparison with periodic and Fourier-style activations, we include SIREN, SNAKE, and a Fourier-feature embedding baseline in both parity and Forrelation experiments, using matched architectures and training protocols unless otherwise noted. For the Fourier-feature embedding baseline, we use a 64-dimensional input mapping with $\sin(2^k \pi x)$ and $\cos(2^k \pi x)$ for $k = 0{:}31$, followed by the same MLP architecture with ReLU.

5.2 Continuous-domain Evaluation

Sinusoidal activations are often effective in continuous-domain learning problems. We therefore evaluate TESLA on three representative settings: (i) physics-informed neural networks (PINNs) for PDE solving, (ii) implicit neural representations (INRs) for coordinate-based image reconstruction, and (iii) mixed-frequency regression for explicit high-frequency signal modeling.

Physics-informed neural networks (PINNs). We train a PINN for the 1D viscous Burgers' equation. The total loss is

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{PDE}} + 100\,\mathcal{L}_{\text{IC}} + 100\,\mathcal{L}_{\text{BC}}, \quad (8)$$

where $\mathcal{L}_{\text{PDE}}$, $\mathcal{L}_{\text{IC}}$, and $\mathcal{L}_{\text{BC}}$ are mean-squared errors of the PDE residual, initial condition, and boundary conditions, respectively.

As shown in Table 1, TESLA achieves the lowest final loss on the Burgers' equation PINN among the compared activations.

Table 1: 1D Burgers' equation PINN: final losses after 20,000 epochs (MSE).

Activation	$\mathcal{L}_{\text{total}}$	$\mathcal{L}_{\text{PDE}}$	$\mathcal{L}_{\text{IC}}$	$\mathcal{L}_{\text{BC}}$
ReLU	7.13×10^{-1}	5.52×10^{-1}	1.31×10^{-3}	3.03×10^{-4}
Tanh	1.19×10^{-2}	3.81×10^{-3}	2.93×10^{-6}	7.77×10^{-5}
SIREN	1.04×10^{-2}	6.00×10^{-3}	1.06×10^{-5}	3.38×10^{-5}
TESLA	$\mathbf{1.21 \times 10^{-3}}$	$\mathbf{5.01 \times 10^{-4}}$	$\mathbf{1.11 \times 10^{-6}}$	$\mathbf{6.00 \times 10^{-6}}$

Implicit neural representations (INRs). We evaluate INRs on Kodak24 and DIV2K, where a 6-layer MLP (512 hidden units) is trained per image for 3000 epochs under an identical training budget.

Table 2: Implicit neural representations on Kodak24 and DIV2K. We report reconstruction quality (PSNR/SSIM) and the number of epochs needed to reach 25 dB (Epochs@25dB).

Dataset	Activation	PSNR ↑	SSIM ↑	Epochs@25dB ↓
Kodak24	TESLA	**27.72**	**0.9671**	**1595.8**
	SIREN	25.33	0.9404	1887.5
	ReLU	21.22	0.8389	3000.0
	Tanh	20.17	0.7818	3000.0
DIV2K	TESLA	**24.73**	**0.9522**	**2485.0**
	SIREN	22.77	0.9301	2695.0
	ReLU	18.21	0.8030	3000.0
	Tanh	17.94	0.7819	3000.0

Mixed-frequency regression. We regress a target signal composed of mixed sine/cosine components from 3.29 Hz to 79.90 Hz. Table 3 reports the dominant frequencies in the target (left) and train/test MSE of different activations (right).

Table 3: Left: top 5 dominant frequencies of the target synthetic function. Right: train/test MSE comparison across activation functions for mixed-frequency signal modeling.

Rank	Freq (Hz)	Amp	Phase
1	79.19	0.98	1.79
2	79.23	0.96	4.41
3	9.78	0.95	5.97
4	18.02	0.92	1.42
5	3.29	0.89	4.72

Model	Train ↓	Test ↓
ReLU	0.0682	0.0681
GeLU	0.0752	0.0756
SiLU	0.0753	0.0756
SIREN	0.0236	0.0248
TESLA	**0.0097**	**0.0100**

In this mixed-frequency setting, TESLA is the only method with test MSE below 0.011.

5.3 Parity with Label Noise: Task, Metrics, and Interpretation

Task. We evaluate on Parity and global-statistics problems where the target depends on a high-order global interaction of the input bits. Let $x \in \{0,1\}^d$ and $S \subseteq [d]$ with $|S| = m$. The clean label is

$$y_{\text{clean}} = \bigoplus_{i \in S} x_i \in \{0,1\},$$

i.e., the parity of the selected bits.[1] This benchmark is intentionally adversarial to local, piecewise-linear activations, as the Bayes-optimal classifier is a global rule with a single nonzero Fourier coefficient at frequency S.

Data Generation, Splits, and Noise Settings. For each run, we sample i.i.d. inputs uniformly from $\{0,1\}^d$ and assign splits deterministically via a fixed hash partition $h(x) \bmod 3$, ensuring disjoint and reproducible train/validation/test sets (i.e., no leakage). We then sample without replacement within each split, using 100,000 training samples and 20,000 validation samples for all d. We evaluate three settings: (i) clean parity, (ii) noisy parity with independent label flips $p \in \{0.05, 0.10, 0.20, 0.30\}$, and (iii) LPN; in the noisy setting, train and test use the same flip rate p.

LPN. We also evaluate the classical LPN (Learning Parity with Noise) variant, where the relevant subset is unknown and must be inferred from data. Fix a hidden secret vector $s \in \{0,1\}^d$ and draw query vectors $a \sim \text{Unif}(\{0,1\}^d)$ i.i.d. The observed label is

$$y \;=\; \langle a, s \rangle \bmod 2 \;\oplus\; e, \quad e \sim \text{Bernoulli}(\eta).$$

Compared to the noisy parity setting in Sec. 5.3—where the subset S is fixed per run and implicitly known to the data generator—LPN additionally hides S (equivalently s), making both identification and optimization more challenging at larger d.

Metrics and Bayes Limit. With symmetric label noise (flip rate $p < \frac{1}{2}$) on the test set, perfect recovery of the underlying rule cannot yield 100% measured accuracy. The Bayes-optimal expected test accuracy under symmetric, input-independent label flips at rate p is $A_{\max}(p) = 1 - p$.

5.4 Forrelation

Forrelation (Aaronson and Ambainis, 2015) is a decision problem on pairs of Boolean functions $f, g : \{0,1\}^n \to \{\pm 1\}$. Let $M = 2^n$. The goal is to decide whether the correlation

$$\Phi_{f,g} = \frac{1}{\sqrt{M}} \langle Hf, g \rangle$$

is high or low under a standard promise. Here H is the Hadamard matrix and $\langle \cdot, \cdot \rangle$ denotes the inner product.

[1] We use a fixed subset S per run; d (number of bits) controls task difficulty.

We treat $\Phi_{f,g} \geq 0.6$ as positive and $|\Phi_{f,g}| \leq 0.01$ as negative. The property of being forrelated is global. A classifier must integrate information spread across the entire pair of truth tables. We use a two-layer MLP that takes as input the concatenation of the truth tables of f and g with length 2^{n+1} and outputs a single logit for binary prediction. We train and evaluate on a synthetic dataset with $n = 12$, consisting of 10,000 training examples and 10,000 test examples, where labels are assigned as $y = 1$ if $\Phi_{f,g} \geq 0.6$ and $y = 0$ if $|\Phi_{f,g}| \leq 0.01$, excluding samples with intermediate values.

5.5 ImageNet-100 Classification

To empirically validate the performance and efficiency of our proposed activation function, TESLA, we conducted experiments on the ImageNet-100 dataset (Russakovsky et al., 2015), a standard benchmark for image classification. To demonstrate TESLA's general applicability, our evaluation includes both Transformer-based models (ViT (Dosovitskiy et al., 2021), MLP-Mixer (Tolstikhin et al., 2021)) and CNN-based models (ResNet (He et al., 2016), MobileNetV3 (Howard et al., 2019)). We compare TESLA's performance against conventional activation functions, namely ReLU (Nair and Hinton, 2010), GeLU (Hendrycks and Gimpel, 2016), and SiLU (Elfwing et al., 2018). TESLA uses $K = 2$ for ViT-T/16, ResNet-18, and MobileNetV3, and $K = 6$ for MLP-Mixer. The evaluation is based on a comprehensive analysis of not only the classification performance (Top-1 Accuracy) but also key computational efficiency metrics, including FLOPs, throughput (imgs/s), and the number of parameters. Table 4 shows these results. The top-1 accuracy is reported as the mean and standard deviation over three seeds and both the measurement of FLOPs and throughput measurement are conducted on a single GPU.

5.6 Result Analysis

Parity. Figure 3 shows the performance of TESLA and baseline models. For the parity experiments we use a fully-connected MLP with 2 hidden layers and hidden dimension $H = 128$. Across the plotted settings, TESLA matches the Bayes limit across noise levels while remaining near 100% on the clean task. For $p \in \{0.05, 0.10, 0.20, 0.30\}$, TESLA's final accuracy concentrates near $\{95\%, 90\%, 80\%, 70\%\}$. In contrast, common activations always collapse toward 50% as the number of bits grows, which is no better than random guessing of parity. In the LPN setting, TESLA sustains high accuracy up to 27-bits before degrading in the hardest regimes.

Table 4: ImageNet-100 results across architectures comparing TESLA with common activations. Top-1 accuracy is mean ± std (3 seeds). Bold indicates best per architecture. FLOPs reported in GigaFLOPs, throughput (imgs/s) measured on a single GPU, and parameters (Params) in millions.

Model	Act	Top-1 (%)	FLOPs	imgs/s	Params
ViT-T/16	ReLU	62.49 ± 0.73	1.08	7,512	5.7
	GeLU	$\mathbf{63.67 \pm 0.76}$	1.08	7,491	5.7
	SiLU	63.60 ± 0.26	1.08	7,482	5.7
	TESLA	62.53 ± 0.71	1.09	7,509	5.7
MLP-Mixer-b16	ReLU	57.81 ± 0.13	12.62	1,151	59.9
	GeLU	57.42 ± 0.01	12.62	1,144	59.9
	SiLU	58.36 ± 0.91	12.62	1,146	59.9
	TESLA	$\mathbf{58.79 \pm 0.02}$	12.75	1,139	59.9
ResNet-18	ReLU	73.14 ± 0.31	1.82	8,447	11.7
	GeLU	73.93 ± 0.47	1.82	8,307	11.7
	SiLU	73.99 ± 0.66	1.82	8,326	11.7
	TESLA	$\mathbf{74.01 \pm 0.33}$	1.82	8,433	11.7
MobileNetV3-S	ReLU	68.20 ± 0.16	0.05	35,607	2.0
	GeLU	$\mathbf{68.23 \pm 0.94}$	0.05	35,082	2.0
	SiLU	67.92 ± 0.15	0.05	35,056	2.0
	TESLA	67.82 ± 0.17	0.05	35,483	2.0

Periodic baselines on parity. Table 5 compares TESLA against periodic and frequency-based baselines. For easier settings (16–20 bits), TESLA and SIREN both reach 100% accuracy under no noise. However, as the bit-length and noise increase, SIREN collapses to random guessing ($\approx 50\%$) while TESLA remains stable (e.g., at 24 bits with 30% noise, TESLA achieves 65.53% vs. 49.47% for SIREN; at 32 bits with no noise, TESLA achieves 95.87% vs. 50.10% for SIREN). SNAKE remains near chance across all settings, and Fourier-Emb similarly fails to capture the global interaction signal in these regimes.

Forrelation. We evaluate a two-layer MLP on the $n = 12$ Forrelation dataset. Figure 4 reports validation accuracy over epochs for hidden sizes $H = \{128, 512, 1024\}$. TESLA consistently learns faster and reaches higher final accuracy than standard activations across all K settings. Larger hidden sizes improve performance for every method. The gap in favor of TESLA remains, which indicates that TESLA captures the global correlation signal more effectively.

ImageNet-100. Table 4 reports results on ViT-T/16, MLP-Mixer, ResNet-18, and MobileNetV3. To ensure architecture-aware comparison, we keep activations inside convolutional blocks unchanged for CNNs and replace only non-convolutional activations (e.g., MLP heads, projection/FFN layers) with TESLA; for ViT and MLP-Mixer, we replace all pointwise activations. TESLA is trained with an explicit ℓ_1 penalty enforcing a layer-wise coefficient budget A_K for stability and generalization control. Across models, TESLA

Bits	Activation	Noise probability			
		0.0	0.1	0.2	0.3
16	TESLA($K = 8$)	**100.00**	89.80	79.22	68.00
	SIREN($w_0 = 30.0$)	**100.00**	**89.83**	**79.61**	**69.53**
	SNAKE($\alpha = 1.0$)	49.95	49.97	49.58	49.89
	Fourier-Emb	73.21	50.19	50.09	50.34
20	TESLA($K = 8$)	**100.00**	89.52	78.78	67.14
	SIREN($w_0 = 30.0$)	**100.00**	**89.74**	**79.42**	**69.67**
	SNAKE($\alpha = 1.0$)	50.01	50.02	50.06	50.07
	Fourier-Emb	52.13	49.68	50.12	50.33
24	TESLA($K = 8$)	**100.00**	89.75	78.50	**65.53**
	SIREN($w_0 = 30.0$)	**100.00**	**90.08**	**80.06**	49.47
	SNAKE($\alpha = 1.0$)	50.24	49.90	50.52	49.97
	Fourier-Emb	50.55	49.20	50.20	50.00
28	TESLA($K = 8$)	**96.97**	**89.61**	**78.10**	**68.14**
	SIREN($w_0 = 30.0$)	50.48	50.52	49.58	49.78
	SNAKE($\alpha = 1.0$)	49.95	49.36	50.02	50.19
	Fourier-Emb	49.74	50.45	50.27	49.81
32	TESLA($K = 8$)	**95.87**	**85.78**	**78.05**	**68.78**
	SIREN($w_0 = 30.0$)	50.10	49.89	50.02	49.95
	SNAKE($\alpha = 1.0$)	50.24	50.37	50.34	50.10
	Fourier-Emb	50.06	49.42	50.12	50.48

Table 5: Parity with long bit-length and label noise: validation accuracy(%) (20 epochs) under matched architectures and training settings.

remains competitive in Top-1 accuracy while keeping FLOPs and throughput within about 1% of standard activations. These results indicate that TESLA extends beyond synthetic parity-style benchmarks as a practical drop-in replacement for standard activations in realistic vision workloads.

Stability vs. sinusoidal baselines. To assess whether TESLA remains numerically stable in practical architectures, we also compare against SIREN under the same ImageNet-100 training settings. In these runs, SIREN exhibits severe optimization instability and much lower accuracy, whereas TESLA trains stably and reaches standard accuracy levels (see Table 6).

Model	Act	Top-1 (%)
ViT-T/16	SIREN	4.11 ± 0.67
	TESLA	$\mathbf{62.53 \pm 0.71}$
MLP-Mixer-b16	SIREN	32.41 ± 10.45
	TESLA	$\mathbf{58.79 \pm 0.02}$
ResNet-18	SIREN	37.41 ± 1.34
	TESLA	$\mathbf{74.01 \pm 0.33}$
MobileNetV3-S	SIREN	39.53 ± 0.85
	TESLA	$\mathbf{67.82 \pm 0.17}$

Table 6: ImageNet-100 comparison between TESLA and SIREN under identical training settings. Top-1 accuracy is mean ± std.

6 FUTURE WORK

TESLA opens several promising avenues for future research. Theoretically, we plan to extend our continuous Fourier and NTK analysis to discrete Boolean domains to establish rigorous approximation bounds and explain the empirical gains observed on parity-type tasks. On the systems side, we will investigate sparse factorizations, low-bit quantization, and hardware-aware implementations to preserve TESLA's spectral benefits while minimizing inference latency. Finally, we aim to integrate TESLA into large-scale Transformers and LLMs—such as augmenting FFN activations or positional encodings—to evaluate its impact on long-range reasoning, compositional generalization, and sample efficiency.

7 CONCLUSION

In this work, we introduced TESLA, a learnable sinusoidal activation that enables explicit degree/frequency control while preserving stable optimization through coefficient budgeting. We provided Lipschitz and Rademacher-complexity bounds and analyzed mode-wise learning dynamics to connect the parameterization to frequency-selective behavior.

Empirically, TESLA consistently improves tasks that require global, high-order interactions. On parity with label noise, it remains well above chance at larger bit-lengths and heavy corruption, and is more stable than periodic/frequency-based baselines (SIREN, SNAKE, and Fourier feature embeddings) in the hardest regimes. On Forrelation, TESLA achieves the highest accuracy across widths, while periodic and Fourier-style baselines remain near chance. On continuous-domain tasks (PINNs, INRs, and mixed-frequency regression), TESLA is competitive with or better than both standard and sinusoidal activations. On ImageNet-100, TESLA remains competitive across architectures with modest overhead and is substantially more stable than SIREN. Overall, TESLA suggests that activation-level degree control is a robust, practical alternative to input-only spectralization.

References

Aaronson, S. and Ambainis, A. (2015). Forrelation: A problem that optimally separates quantum from classical computing. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC 2015)*, page 307–316.

Agostinelli, F., Hoffman, M. D., Sadowski, P. J., and Baldi, P. (2015). Learning activation functions to improve deep neural networks. In *3rd International Conference on Learning Representations (ICLR 2015), Workshop Track Proceedings*.

Apicella, A., Donnarumma, F., Isgrò, F., and Prevete, R. (2021). A survey on modern trainable activation functions. *Neural Networks*, 138:14–32.

Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*.

Barron, A. R. (1994). Approximation and estimation bounds for artificial neural networks. *Machine learning*.

Daniely, A. and Malach, E. (2020). Learning parities with neural networks. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, pages 20356–20365.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations (ICLR 2021)*.

Eldan, R. and Shamir, O. (2016). The power of depth for feedforward neural networks. In *29th Annual Conference on Learning Theory (COLT 2016)*, pages 907–940.

Elfwing, S., Uchibe, E., and Doya, K. (2018). Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV 2015)*, pages 1026–1034.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016)*, pages 770–778.

Hendrycks, D. and Gimpel, K. (2016). Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q. V., and Adam, H. (2019). Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV 2019)*, pages 1314–1324.

Jacot, A., Gabriel, F., and Hongler, C. (2018). Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*, pages 8571–8580.

Ledoux, M. and Talagrand, M. (2013). *Probability in Banach Spaces: isoperimetry and processes*. Springer Science & Business Media.

Liu, Z., Wang, Y., Vaidya, S., et al. (2024). Kan: Kolmogorov–arnold networks. *arXiv preprint arXiv:2404.19756*.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML 2010)*, pages 807–814.

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., and Courville, A. (2019). On the spectral bias of neural networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML 2019)*, pages 5301–5310.

Rahimi, A. and Recht, B. (2007). Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems 20 (NIPS 2007)*, pages 1177–1184.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*.

Scardapane, S., Scarpiniti, M., Comminiello, D., and Uncini, A. (2019). *Learning Activation Functions from Data Using Cubic Spline Interpolation*. Springer International Publishing.

Sitzmann, V., Martel, J., Bergman, A., Lindell, D., and Wetzstein, G. (2020). Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, pages 7462–7473.

Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., and Ng, R. (2020). Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, pages 7537–7547.

Telgarsky, M. (2016). benefits of depth in neural networks. In *29th Annual Conference on Learning Theory (COLT 2016)*, pages 1517–1539.

Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., Lucic, M., and Dosovitskiy, A. (2021). Mlp-mixer: An all-mlp architecture for vision. In *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, pages 24261–24272.

Walsh, J. L. (1923). A closed set of normal orthogonal functions. *American Journal of Mathematics*.

Yarotsky, D. (2017). Error bounds for approximations with deep relu networks. *Neural Networks*, 94:103–114.

Zhi-Qin John Xu, Z.-Q. J. X., Yaoyu Zhang, Y. Z., Tao Luo, T. L., Yanyang Xiao, Y. X., and Zheng Ma, Z. M. (2020). Frequency principle: Fourier analysis sheds light on deep neural networks. *Communications in Computational Physics*.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Yes/No/Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

TESLA: Taylor Expansion of Sinusoidal Learnable Activations Supplementary Materials

1 Experiment environment summary

This appendix provides details, such as hyperparameter, computer hardware and software environment information, necessary to reproduce the experiments reported in the main text. K denotes the number of sinusoidal harmonics in each TESLA layer, and A_K is the l_1-style coefficient budget for controlling their amplitude.

Table 1: Hyperparameter summary for main experiments. Each reported experiment in the paper uses the settings below unless noted. All task were conducted on backbone with MLP, 2 layers, and optimizer with Adam and $A_k = 0$.

Task	Width	K	LR	Batch Size	Epochs
Parity ($d = 16 - 32$, $n = 100k$)	128	d/4	1e-3	1024	30
Forrelation ($d = 12$)	[128, 512, 1024]	[1, 2, 4, 16]	1e-3	128	20
LPN (noise = 0.1)	128	d/4	5e-4	1024	30

Table 2: Hyperparameter summary for ImageNet-100. All models are trained with AdamW for 50 epochs using cosine LR decay. Warmup uses a linear schedule; Warmup ratio is the LR multiplier at the first warmup step. All backbones fix the batch size to 128, set the learning rate to 1e-3, perform warm-up for 5 epochs, and set the warm-up ratio to 0.85.

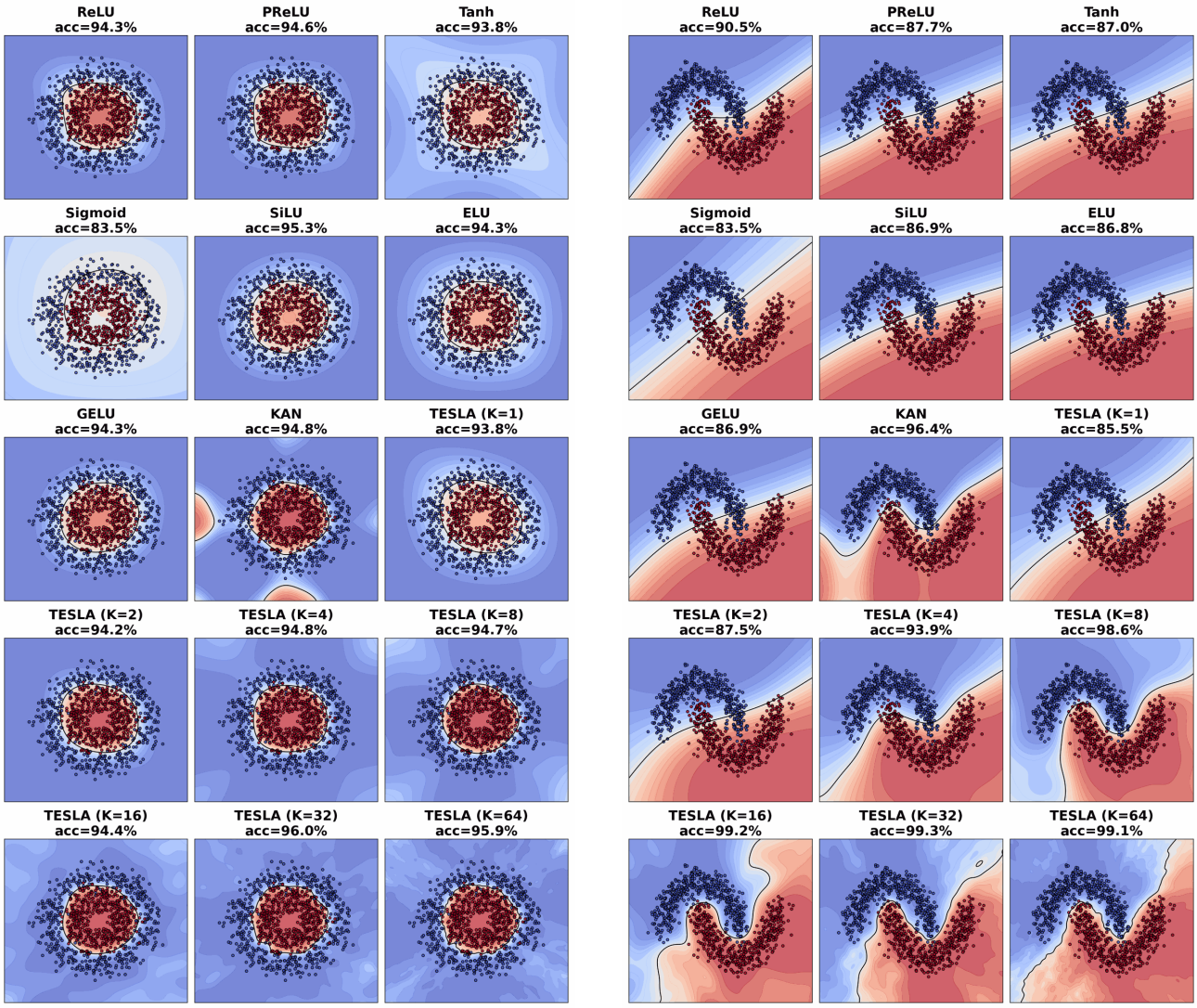
Backbone	K	A_K
ViT-T/16	2	1e-3
MLP-Mixer-b16	6	1e-1
ResNet-18	2	1e-3
MobileNetV3-S	2	1e-3

Table 3: Compute and software environment used for all experiments unless otherwise noted.

Hardware	NVIDIA A100 80GB (1x), AMD EPYC 7543 32-Core Processor, 256GB RAM
OS	Ubuntu 22.04 LTS
Python / Framework	Python 3.10, PyTorch 2.0+
CUDA / cuDNN / Driver	CUDA 11.8, cuDNN 8.x, NVIDIA Driver 535+
Key Python deps	numpy 1.26, scipy 1.14, triton 3.1.0, tqdm 4.x
Reproducibility	Fixed seeds $\{0, 1, 2\}$; all results reported as mean $\pm$ std over seeds.

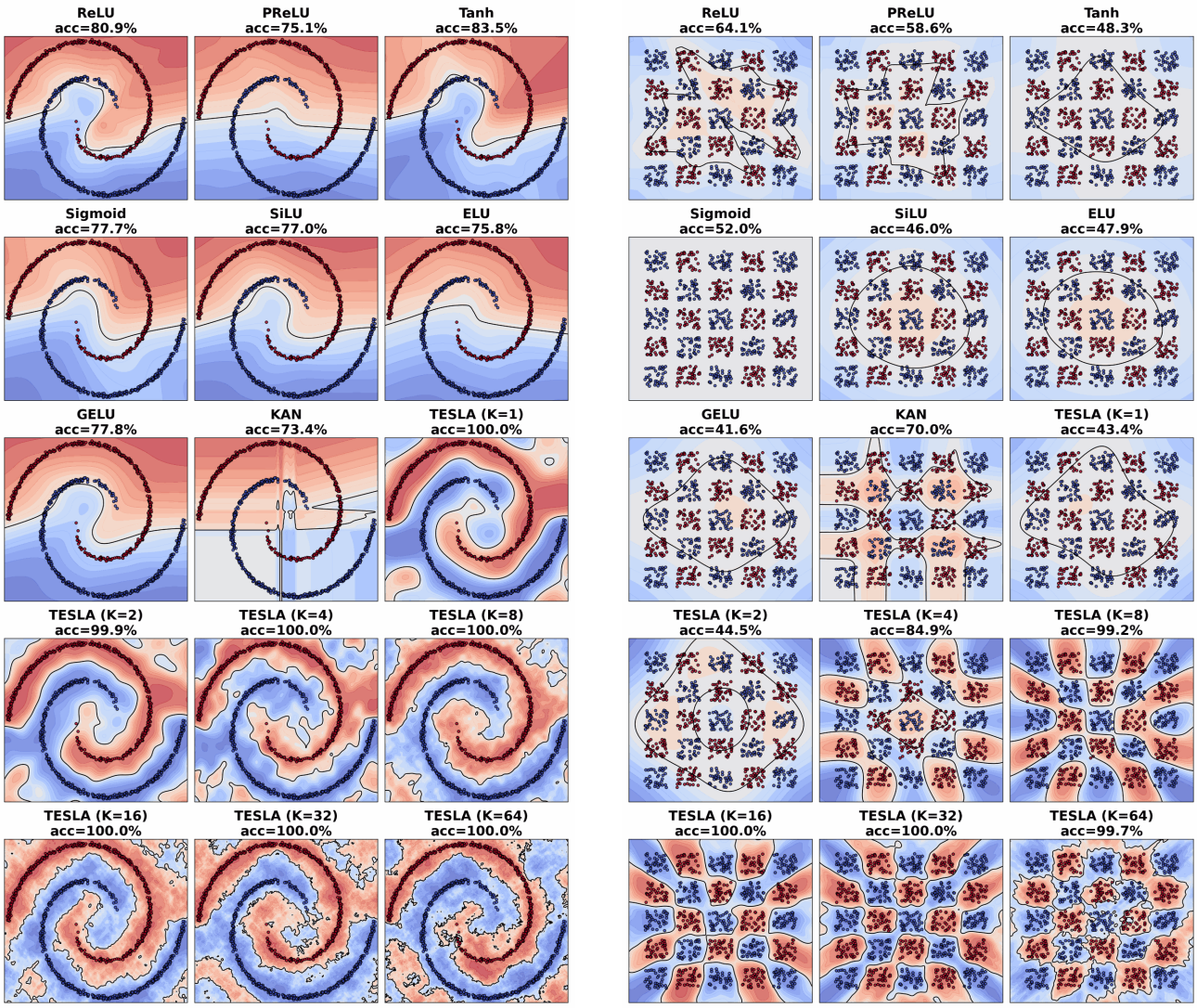
2 Representations and Generalization on Toy Tasks: Synthetic Data & Parity

This section complements the theory by visualizing how TESLA differs from standard activations on synthetic 2D tasks and on the 10-bit parity mapping.



(a) **Concentric Circles**. (b) **Moons**.

Figure 1: Decision-boundary comparison on two synthetic datasets. A two-layer MLP with 128 hidden units is trained for 200 epochs on 1,000 samples. TESLA is evaluated with multiple K and compared against KAN and standard activations. Subpanel titles report training accuracy in percent. All subpanels share the same color and legend scale.



(a) **Spiral**. (b) **Checkerboard**.

Figure 2: Decision-boundary comparison on two additional datasets. Training setup matches Figure 1. TESLA consistently recovers global or oscillatory structure, whereas baselines often miss these patterns. Subpanel titles report training accuracy in percent. Color and legend scales are shared across subpanels.

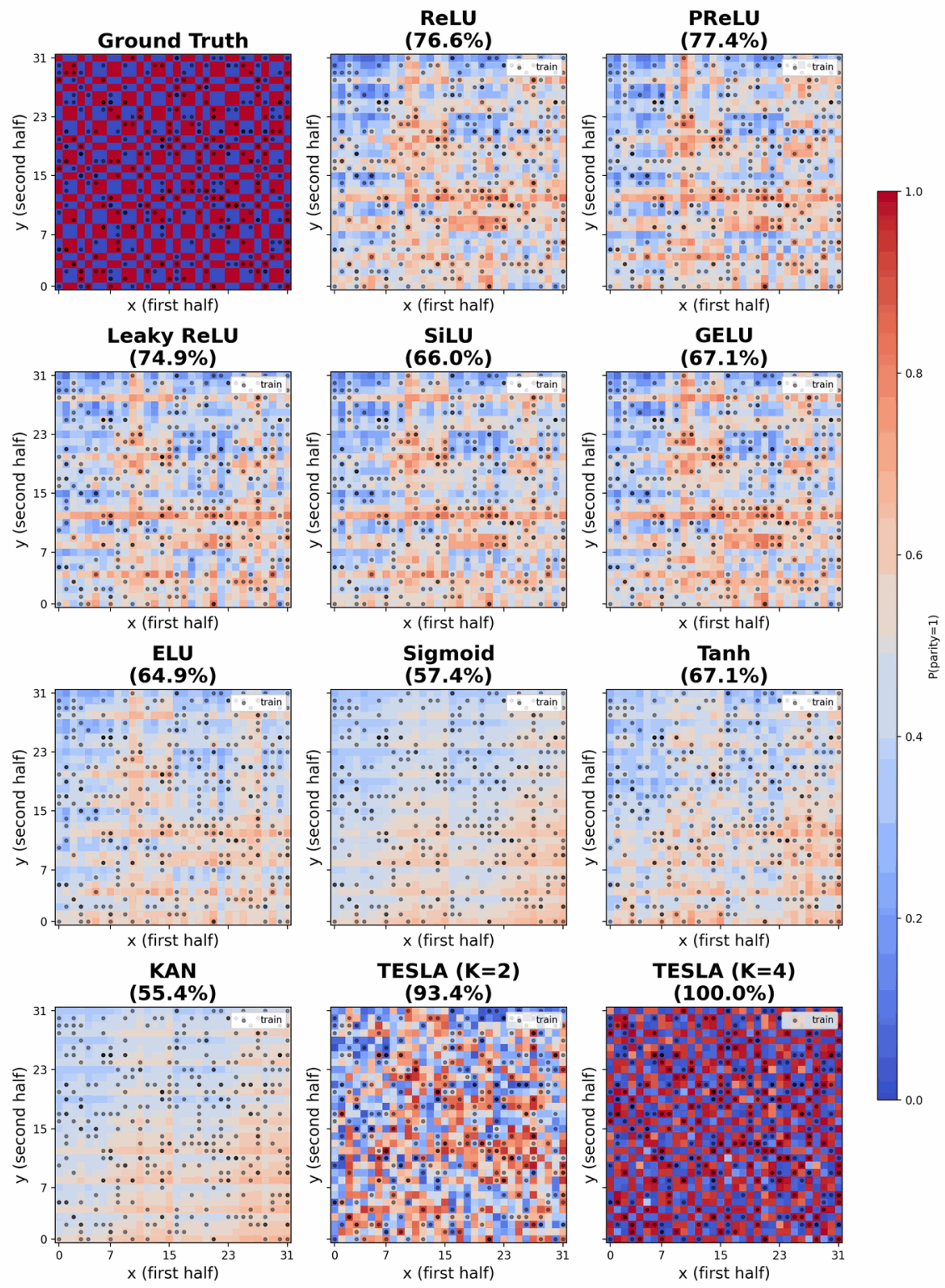


Figure 3: **Parity decision surfaces on 10-bit inputs.** A two-layer MLP (hidden = 128, lr = 1e-3, 50 epochs) is trained on 350 samples with different activations. Each 10-bit string is split into two 5-bit halves mapped to the x- and y-axes as binary integers (0–31), yielding a 32×32 grid. Color shows predicted $P(y = 1)$ (blue = 0, red = 1). Black dots mark training points. The top-left panel is ground truth. Panel titles report training accuracy (%).