POLYGATE: CONTINUOUS POWER-PRODUCT GATING FOR DEEP NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Standard feedforward and convolutional neural networks parameterized with piecewise-linear activations approximate smooth polynomial and multiplicative interactions through piecewise-linear tessellations, whereas multiplicative architectures can represent power-law and cross-variable interactions directly. However, historical product-unit formulations suffered from unstable optimization, logarithmic singularities near the origin, and undefined real-valued fractional powers on negative inputs. In this work, we propose POLYGATE, an architectural primitive that augments affine coordinate bases with learnable continuous power-product gating. To ensure well-defined real-valued power operations and exclude zero-boundary logarithmic singularities from the operating domain, we introduce Strict-Positivity Batch Normalization (SP-BN) and explicit positive-domain handling. We further introduce dimension-normalized multiplicative gating to control activation variance as feature dimension grows. Theoretically, we characterize exact representation of univariate and multivariate generalized monomials, gradient regularity on strictly positive bounded domains, near-identity initialization with asymptotic recovery of the affine baseline, and dimension-scaling laws of multiplicative gates. Empirically, POLYGATE reduces test MSE by 87.4% on Feynman relativistic mechanics, achieves higher mean/median out-of-distribution R^2 across the complete 10-equation AI-Feynman suite, and is evaluated across convolutional backbones on Tiny-ImageNet-200: delivering $+0.10\%$ to $+0.55\%$ gains on edge MobileNetV4 with modest parameter overhead ($+2.9\%$ to $+7.2\%$), and $+9.09\%$ to $+12.34\%$ gains on ConvNeXt with dual depthwise-pointwise gating.

1 INTRODUCTION

The foundational computational unit of modern deep neural architectures is predominantly affine (He et al., 2016; Vaswani et al., 2017; Dosovitskiy et al., 2021):

$$h = \sigma\left(Wx + b\right) \tag{1}$$

where $W \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$, $b \in \mathbb{R}^{d_{\text{out}}}$, and $\sigma(\cdot)$ is an activation function. When $\sigma(\cdot) = \text{ReLU}(\cdot) = \max(0, \cdot)$ or a piecewise-linear variant such as LeakyReLU (Maas et al., 2013), the network parameterizes a Continuous Piecewise-Linear (CPWL) function (Montufar et al., 2014; Telgarsky, 2016; Raghu et al., 2017). When smooth nonlinear activations are used (e.g., GeLU, SiLU/Swish) (Hendrycks & Gimpel, 2016; Ramachandran et al., 2017), the resulting network generates smooth mappings, yet remains constrained to the affine-plus-activation paradigm. While universal approximation theorems guarantee that standard networks can approximate compact continuous functions with sufficient width or depth, ordinary affine-plus-activation layers do not directly parameterize explicit learnable generalized monomials or cross-variable products in a single operator. Consequently, their representational efficiency degrades when target functions exhibit smooth multiplicative curvature, fractional power laws, or coupled physical invariants (Petersen & Voigtlaender, 2018; Daubechies et al., 2022; Udrescu & Tegmark, 2020).

Specifically, for a curved univariate function such as x^2 on a compact interval, piecewise-linear approximation requires a number of linear segments scaling as $\Omega(\epsilon^{-1/2})$ to achieve uniform error ϵ under the standard 1D interpolation setting (Yarotsky, 2017), whereas a polynomial-gated operator can represent x^2 exactly. In multivariate settings, capturing cross-variable interactions $\prod_{i=1}^{d} x_i^{p_i}$ via

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

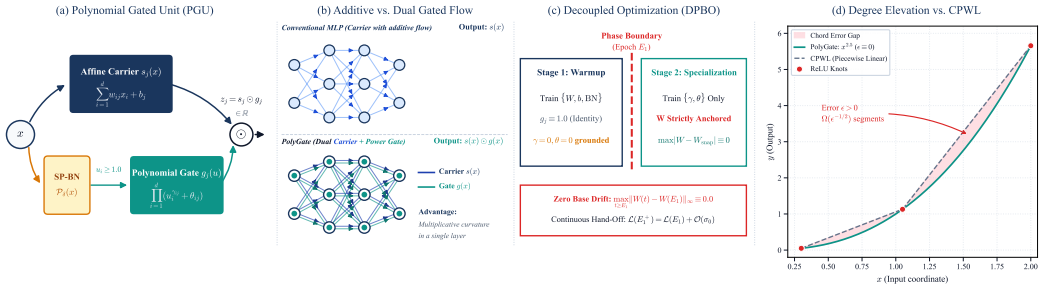


Figure 1: **Overview of PolyGate Architecture and Optimization. (a)** PGU internal schematic: dual-path affine carrier $s_j(x)$ and power gate $g_j(u)$, combined via elementwise multiplication with SP-BN enforcing strict positivity ($u_i \geq 1.0$). **(b)** Conventional feedforward network (top: single directed paths, additive aggregation $\sum wx + b$) vs. PolyGate (bottom: dual carrier/gate flow, combined additive and multiplicative interactions $s \odot g$) in a 2-3-3-2 configuration. **(c)** DPBO timeline: Stage 1 trains the base; at phase boundary E_1, base weights are frozen ($\max|W - W_{\text{snap}}| \equiv 0$) while Stage 2 specializes gate parameters. **(d)** Expressivity comparison: target continuous power law ($x^{2.5}$) approximated by continuous piecewise-linear (CPWL) network ($\epsilon > 0$) vs. exact representation by PolyGate ($\epsilon = 0$).

additive activations requires standard networks to combine many hyperplanes to tessellate smooth multiplicative manifolds.

Multiplicative representations historically appeared in *Higher-Order Networks* and *Product Units* (Giles & Maxwell, 1987; Durbin & Rumelhart, 1989), replacing linear sums with products $\prod x_i^{p_i}$. However, early product units faced severe instabilities (Leerink et al., 1995): unconstrained products lack an identity baseline, real powers are undefined for $x \leq 0$, and logarithmic gradients $\frac{\partial}{\partial \gamma} x^\gamma = x^\gamma \ln x$ diverge as $x \to 0^+$. Modern gated blocks (GLU, SwiGLU, SE) (Dauphin et al., 2017; Shazeer, 2020; Hu et al., 2018) incorporate elementwise gating, but operate as bilinear modulations without continuous coordinate-wise polynomial powers.

The PolyGate Formulation. In this work, we address these challenges with **PolyGate**, structured around the **Polynomial Gated Unit (PGU)**. The pre-activation z_j augments an affine coordinate base $s_j(x)$ with a multi-variable polynomial gate $g_j(x)$:

$$z_j = s_j(x) \cdot g_j(x) = \left(\sum_{i=1}^{d} w_{ij} x_i + b_j \right) \times \prod_{i=1}^{d} \phi \left(x_i^{\gamma_{ij}} + \theta_{ij} \right) \tag{2}$$

To ensure numerical stability in deep networks, we introduce: (1) **Strict-Positivity Batch Normalization (SP-BN)**, $\mathcal{P}_\delta(u) \triangleq \text{ReLU}(\text{BN}(u)) + \delta$ ($\delta \geq 1.0$), mapping representations into $[1.0, \infty)$ to eliminate branch cuts and origin singularities; (2) dimension-normalized geometric-mean gating to prevent variance explosion; and (3) dual optimization schemes supporting end-to-end co-adaptation (**Joint**) and modular coordinate specialization (**DPBO**).

Key Contributions.

1. **Strict-Positivity Architectural Primitive**: We formalize the Polynomial Gated Unit (PGU) for dense and convolutional networks, stabilized by SP-BN ($\delta \geq 1.0$) to guarantee well-defined real powers and bounded logarithmic gradients.
2. **Theoretical Foundations**: We prove: (i) exact representation of generalized monomials under rank-1 variety constraints; (ii) gradient regularity on strictly positive domains; (iii) near-identity initialization with baseline recovery; and (iv) variance stabilization via geometric-mean normalization.
3. **Dual Optimization Schemes**: We formalize **PolyGate (Joint)** for end-to-end co-adaptation, and **PolyGate (DPBO)** for modular gate specialization on anchored carrier coordinates with verified zero base drift.
4. **Empirical Audits & Scaling**: Across physical regression (87.4% MSE cut), controlled monomial scaling, and vision backbones on Tiny-ImageNet-200 (MobileNetV4 gains of $+0.10\%$ to $+0.55\%$ and ConvNeXt gains of $+9.1\%$ to $+12.3\%$), PolyGate improves accuracy and extrapolation.

Table 1: **Systematic Comparison of Neural Operator Paradigms.** Contrasting representation structures, domain mechanisms, initialization baselines, and scalability across foundational and modern architectures.

Paradigm	Key References	Continuous $\gamma \in \mathbb{R}$	Negative Domain Mechanism	Product Structure	Identity Baseline Init	Conv Integration	Primary Application Scope
CPWL / ReLU	He et al. (2016); Yarotsky (2017)	No	Native ($\mathbb{R}^d$)	Tessellation $\Omega(\epsilon^{-1/2})$	Native	Native	Broad deep learning
Product Units	Durbin & Rumelhart (1989); Leerink et al. (1995)	Yes	Undefined ($x \leq 0$)	Direct product $\prod x_i^{p_i}$	No (random products)	No	Low-dimensional arithmetic
NMU (Arithmetic)	Madsen & Johansen (2020)	No (discrete)	Sign-split routing	Subset product $\prod_{i \in S} x_i$	Partial	No	Discrete arithmetic reasoning
NPU / RealNPU	Heim et al. (2020)	Yes	Complex polar coords	Power product $\prod x_i^{p_i}$	No	No	Symbolic equation discovery
Π-Nets	Chrysos et al. (2020; 2021)	No (integer)	Native ($\mathbb{R}^d$)	Multilinear tensor rank	Partial	Custom multilinear	Visual synthesis & recognition
Splines / KAN	Liu et al. (2024)	No	Knot intervals	Additive composition	Spline identity	No	Low-dimensional scientific ML
PolyGate [Ours]	**This Work**	**Yes**	**SP-BN ($[1, \infty)$) + clamp**	**Affine carrier $\times$ power gate**	**Exact / Asymptotic ($\sigma_0 \to 0$)**	**Spatial DW + PW cross**	**SciML, Regression, Vision Scaling**

2 Related Work & Novelty Positioning

We position PolyGate relative to foundational and modern architectural paradigms in Table 1.

Continuous Piecewise Linear (CPWL) Networks. ReLU networks partition input space into polyhedral linear regions (Montufar et al., 2014; Telgarsky, 2016; Raghu et al., 2017). Approximating smooth univariate curvature such as x^2 requires $\Omega(\epsilon^{-1/2})$ linear segments under classical 1D interpolation bounds (Yarotsky, 2017), while vanishing second derivatives ($f''(x) \equiv 0$) limit extrapolation on physical manifolds (Liang & Srikant, 2017). Smooth activations (GELU, SiLU) introduce continuous curvature but retain the affine-plus-activation structure without native multivariable products. PolyGate parameterizes continuous polynomial curvature directly into coordinate bases.

Product Units and Neural Arithmetic/Power Units. Continuous power representations date back to Higher-Order Networks (Giles & Maxwell, 1987) and Product Units (Durbin & Rumelhart, 1989), which faced severe hurdles: undefined real powers on negative inputs, diverging logarithmic gradients ($\frac{\partial}{\partial p} x^p \to -\infty$ as $x \to 0^+$), and lack of an identity baseline (Leerink et al., 1995). Recent advances revisit arithmetic operators: NMU (Madsen & Johansen, 2020) learns subset products for discrete arithmetic without continuous exponents; NPU (Heim et al., 2020) learns arbitrary powers via complex polar coordinates for symbolic discovery; Π-nets (Chrysos et al., 2020; 2021) model integer polynomial expansions via recursive tensor factorizations. PolyGate does not claim novelty for continuous exponents in isolation; its novelty lies in: (i) augmenting affine carriers with multiplicative power gating ($s(x) \odot g(x)$); (ii) SP-BN positive bounding to prevent complex branch cuts and origin singularities; (iii) dimension-normalized geometric-mean scaling; and (iv) integration into deep convolutional architectures.

Modern Gated Backbones and Feature Modulation. Elementwise gating is widely used via GLU (Dauphin et al., 2017), SwiGLU (Shazeer, 2020), SE (Hu et al., 2018), FiLM (Perez et al., 2018), and selective state spaces (Mamba) (Gu & Dao, 2023). In scientific machine learning, discovering conservation laws requires modeling multi-variable products (Udrescu & Tegmark, 2020; Cranmer et al., 2020; Raissi et al., 2019). Yet standard gates remain bilinear or channel-affine without learnable continuous power laws ($x_i^{\gamma_{ij}}$).

Kolmogorov-Arnold Networks (KAN). KANs (Liu et al., 2024) replace linear weights with learnable B-splines. While effective for 1D fitting, KAN aggregation remains strictly additive: $\mathrm{KAN}(x) = \sum \Phi_q(\sum \phi_{q,p}(x_p))$. Capturing multi-variable cross-products requires deep spline composition with substantial parameter and memory overhead. In contrast, PolyGate computes cross-variable monomials directly via logarithmic aggregation with compact parameterization.

Optimization Geometry and Staged Decoupling. Multiplicative parameters can induce ill-conditioned optimization landscapes with large condition numbers (Sagun et al., 2016; Keskar et al., 2017; Li et al., 2018). By formulating Decoupled Phase-Boundary Optimization (DPBO), we temporally decouple base coordinate alignment from polynomial gate specialization, providing an intervention where carrier parameter drift is verified to be identically zero.

3 The PolyGate Architecture & Operators

We formally define the Polynomial Gated Unit (PGU), Strict-Positivity Batch Normalization (SP-BN), and their realization in dense and convolutional architectures.

Formal Operator Definition: The Polynomial Gated Unit (PGU). Let $x \in \mathcal{D}_+ \subset \mathbb{R}_+^{d_{\text{in}}}$ denote an input feature vector strictly bounded such that $x_i \geq 1.0$ for all $i \in \{1, \ldots, d_{\text{in}}\}$. For output hidden unit $j \in \{1, \ldots, d_{\text{out}}\}$, the pre-activation $z_j \in \mathbb{R}$ is defined as:

$$z_j = s_j(x) \cdot g_j(x) = \left(\sum_{i=1}^{d_{\text{in}}} w_{ij} x_i + b_j \right) \times \prod_{i=1}^{d_{\text{in}}} \phi\left(x_i^{\gamma_{ij}} + \theta_{ij}\right) \tag{3}$$

where $w_{ij}, b_j \in \mathbb{R}$ parameterize the affine base carrier, $\gamma_{ij} \in \mathbb{R}$ are continuous learnable exponents, $\theta_{ij} \in \mathbb{R}$ are additive gate offsets, and $\phi(u) = \text{clamp}(u, \min = 10^{-4})$ is an explicit positivity projection on each binomial term. In log-space computation (matching implementation), $g_j(x) = \exp\left(\text{clamp}\left(\sum_{i=1}^{d_{\text{in}}} \ln \phi\left(x_i^{\gamma_{ij}} + \theta_{ij}\right), -15.0, 15.0\right)\right)$, ensuring exact numerical equivalence with the reference PyTorch module. Each output unit contains d_{in} carrier weights, 1 bias, d_{in} exponents, and d_{in} offsets, totaling $3d_{\text{in}} + 1$ independent learnable parameters. The post-activation hidden state is $h_j = \sigma(z_j)$.

Strict-Positivity Batch Normalization (SP-BN). In real arithmetic, non-integer powers $x^\gamma = \exp(\gamma \ln x)$ are undefined for $x \leq 0$, and the derivative $\frac{\partial}{\partial \gamma} x^\gamma = x^\gamma \ln x$ exhibits a logarithmic singularity as $x \to 0^+$ for $\gamma \leq 0$. Standard batch normalization (Ioffe & Szegedy, 2015) produces zero-centered activations, where half of pre-activations are negative, and standard ReLU clamps them to zero ($\ln 0 = -\infty$). To resolve this, we formulate **SP-BN**:

$$Y = \mathcal{P}_\delta(X) \triangleq \text{ReLU}\left(\text{BN}(X)\right) + \delta \tag{4}$$

where $\delta \geq 1.0$ is a fixed positive shift parameter (default $\delta = 1.0$).

Proposition 1 (Codomain Positivity and Baseline Grounding)**.** *Under $Y = \mathcal{P}_\delta(X)$ with $\delta = 1.0$: (i) **Codomain Positivity**: $Y \geq 1.0$ everywhere (codomain $[1.0, \infty)$); (ii) **Non-Negative Logarithm**: $\ln(Y) \geq 0$, excluding complex branch cuts and zero-boundary singularities; and (iii) **Identity Grounding**: setting $\gamma = 0, \theta = 0$ yields $g_j \equiv 1.0$, identically recovering the baseline affine carrier at onset. Note that while* SP-BN *strictly guarantees the lower bound $Y \geq 1.0$, upper boundedness $Y \leq B$ is satisfied under compact activation domains.*

Dense Multi-Layer Perceptrons. In dense layers, pre-activations follow Eq. equation 3, followed by positive activation $\mathcal{P}_\delta(z) = \text{ReLU}(z) + 1.0$. For tabular regression with standardized inputs $x \sim \mathcal{N}(0, 1)$, an input affine shift $x_{\text{pos}} = \text{clamp}(x + \delta_{\text{in}}, \min = 10^{-4})$ with $\delta_{\text{in}} \geq 2.0$ ensures positivity. Final outputs are unconstrained linear projections without gating.

Convolutional Networks: Depthwise Separable PolyGate. Standard vision backbones (Howard et al., 2017; Chollet, 2017) factorize convolution into depthwise spatial filtering and pointwise channel mixing. Depthwise Separable PolyGate structures polynomial gating into two specialized steps matching this factorization:

(1) Spatial Depthwise PolyGate: Spatial carrier $u^{\text{DW}} = \mathcal{P}_\delta(W_c^{\text{DW}} * X)$ is modulated by a dual-branch depthwise gate:

$$g_{c,h,w}^{\text{DW}} = \left[\text{clamp}\left(\mathcal{P}_\delta(W_g^{\text{DW}} * X)_{c,h,w}, \min = 10^{-4}\right)\right]^{\gamma_c^{\text{DW}}} + \theta_c^{\text{DW}} \tag{5}$$

yielding modulated activations $h_{c,h,w}^{\text{DW}} = u_{c,h,w}^{\text{DW}} \cdot g_{c,h,w}^{\text{DW}}$. Spatial parameters are initialized to zero ($\gamma_c^{\text{DW}} = 0, \theta_c^{\text{DW}} = 0$), recovering $g^{\text{DW}} \equiv 1.0$ at onset. Downstream operations explicitly handle any domain unconstrained by additive offset θ_c^{DW}.

(2) Pointwise Cross-Channel PolyGate: The channel-mixing projection $u^{\text{PW}} = \mathcal{P}_\delta(W^{\text{PW}} * h^{\text{DW}})$ is modulated by a dimension-normalized geometric-mean gate in the log domain:

$$g_{c,h,w}^{\text{PW}} = \exp\left(\text{clamp}\left(\frac{1}{C_{\text{in}}} \sum_{c'=1}^{C_{\text{in}}} \gamma_{c',c}^{\text{PW}} \ln\left(\text{clamp}\left(h_{c',h,w}^{\text{DW}}, \min = 10^{-4}\right)\right) + \theta_c^{\text{PW}}, -10.0, 10.0\right)\right) \tag{6}$$

yielding output $z_{c,h,w}^{\text{PW}} = u_{c,h,w}^{\text{PW}} \cdot g_{c,h,w}^{\text{PW}}$, followed by standard activation $\text{ReLU}(z^{\text{PW}})$. Here: (i) input positivity is enforced by explicit clamping; (ii) θ_c^{PW} provides log-scale bias; (iii) $g^{\text{PW}} > 0$ holds strictly via exponentiation; and (iv) $1/C_{\text{in}}$ enforces $\mathcal{O}(1)$ variance (Theorem 5).

4 THEORETICAL FOUNDATIONS

We analyze the polynomial expressivity, gradient regularity, initialization behavior, and dimension scaling of POLYGATE (complete proofs in Appendix A).

Polynomial Expressivity & Degree Elevation.

Theorem 1 (Univariate Degree Elevation). *Let $d_{\text{in}} = 1$ and $\phi(u) = u$. On $x > 0$, a single pre-activation* PGU *computes $z(x) = (wx+b)(x^\gamma+\theta) = wx^{\gamma+1}+bx^\gamma+(w\theta)x+(b\theta)$. For any $\gamma \in \mathbb{R}$, $z(x) \in \text{span}\{x^{\gamma+1}, x^\gamma, x, 1\}$. When $\gamma = k-1$ for integer $k \geq 1$, $z(x) \in \text{span}\{x^k, x^{k-1}, x, 1\}$ subject to determinantal variety constraint $(w)(b\theta) = (b)(w\theta)$. Individual monomials cx^k are represented exactly by setting $w = c, b = 0, \theta = 0, \gamma = k-1$ (proof in Appendix A.2).*

Remark 1 (Piecewise-Linear Segment Rate for Curved Monomials). For $f(x) = x^2$ on $[a, b]$, piecewise-linear approximation requires $\Omega(\epsilon^{-1/2})$ segments to achieve uniform error ϵ (Yarotsky, 2017), whereas a single PGU ($w = 1, b = 0, \theta = 0, \gamma = 1$) represents x^2 exactly ($\epsilon \equiv 0$).

Theorem 2 (Exact Generalized-Monomial Representation). *Let $x \in (0, \infty)^d$ and $f(x) = c\prod_{i=1}^d x_i^{p_i}$. A single dense* PGU *with carrier $s(x) = \sum_{i=1}^d w_i x_i + b$ and gate $g(x) = \prod_{i=1}^d \phi(x_i^{\gamma_i} + \theta_i)$ represents $f(x)$ exactly by setting $w_i = 0, b = c, \theta_i = 0, \gamma_i = p_i$ for all i, requiring $3d + 1$ parameters (proof in Appendix A.3).*

Gradient Regularity on Strictly Positive Bounded Domains. Let $\delta_j = \frac{\partial \mathcal{L}}{\partial z_j}$ denote upstream gradients. For gate $g_j(x) = \prod_{i=1}^d (x_i^{\gamma_{ij}} + \theta_{ij})$, the exponent loss gradient evaluates to $\frac{\partial \mathcal{L}}{\partial \gamma_{ij}} = \delta_j s_j g_j \left(\frac{x_i^{\gamma_{ij}} \ln x_i}{x_i^{\gamma_{ij}} + \theta_{ij}} \right)$.

Theorem 3 (Gradient Regularity on Strictly Positive Bounded Domains). *Let inputs be bounded by* SP-BN *on a compact domain such that $x_i \in [1, B]$ ($1 \leq B < \infty$), and let $\theta_{ij} \geq 0$. Then for all $\gamma_{ij} \geq 0$: $0 \leq \frac{x_i^{\gamma_{ij}} \ln x_i}{x_i^{\gamma_{ij}} + \theta_{ij}} \leq \ln x_i \leq \ln B < \infty$. When gate inputs and binomial terms are explicitly clamped ($\tilde{x}_i \geq \epsilon, \phi(\tilde{x}_i^\gamma + \theta) \geq \epsilon$), the gradient factor remains uniformly bounded across compact parameter sets, preventing gradient divergence (proof in Appendix A.5).*

Identity Initialization & Baseline Recovery.

Theorem 4 (Identity Initialization and Asymptotic Baseline Recovery). *Consider a* PGU *with pre-activation $z_j(x) = s_j(x) \cdot g_j(x)$:*

1. ***Deterministic Identity Grounding**: When $\gamma_{ij} = 0, \theta_{ij} = 0$, $g_j(x) \equiv 1.0$. The forward pass recovers $z_j(x) \equiv s_j(x)$, and backpropagation Jacobians identically match the affine baseline: $\frac{\partial \mathcal{L}}{\partial w_{ij}} = (\frac{\partial \mathcal{L}}{\partial w_{ij}})_{\text{base}}$.*
2. ***Asymptotic Recovery in Probability**: Let $\gamma_{ij}, \theta_{ij} \sim \mathcal{N}(0, \frac{\sigma_0^2}{d_{\text{in}}} I)$. For any compact positive domain $\mathcal{X} \subset [1, B]^{d_{\text{in}}}$, as $\sigma_0 \to 0$: $\sup_{x \in \mathcal{X}} |g_j(x) - 1.0| \xrightarrow{p} 0$, and parameter Jacobians converge in probability to the affine baseline (proof in Appendix A.6).*

Dimension Scaling Law & The Geometric-Mean Gate.

Theorem 5 (Log-Normal Dimension Scaling Law). *Let $Y_i = \ln T_i \sim \mathcal{N}(0, \sigma^2)$ be i.i.d. Gaussians. For raw gate $G_d = \prod_{i=1}^d T_i = \exp(\sum_{i=1}^d Y_i)$, moments satisfy: $\mathbb{E}[G_d] = \exp(\frac{d\sigma^2}{2})$ and $\text{Var}(G_d) = \exp(d\sigma^2)[\exp(d\sigma^2) - 1]$, growing exponentially with dimension d.*

To ensure dimension-invariant stability across wide feature layers ($d \geq 64$), we utilize the **Dimension-Normalized Geometric-Mean Gate**: $H_d \triangleq (G_d)^{1/d} = \exp\left(\frac{1}{d}\sum_{i=1}^d Y_i\right)$.

Corollary 1 (Geometric-Mean Variance Collapse). *Under Theorem 5, moments of normalized gate H_d satisfy: $\mathbb{E}[H_d] = \exp(\frac{\sigma^2}{2d}) \xrightarrow{d\to\infty} 1.0$ and $\text{Var}(H_d) = \frac{\sigma^2}{d} + \mathcal{O}(\frac{1}{d^2}) \xrightarrow{d\to\infty} 0$, enforcing vanishing activation variance across wide feature dimensions (proof in Appendix A.8).*

5 Optimization Paradigms: Joint Training and Phase-Boundary Decoupling

We analyze two complementary optimization paradigms: (1) **End-to-End Joint Training**, where carrier base and gates co-adapt concurrently; and (2) **Decoupled Phase-Boundary Optimization (DPBO)**, which isolates base coordinate alignment from gate specialization.

Gradient Coupling: Joint Training vs. Decoupled Optimization. Under joint training, carrier weights W and exponents γ exhibit bilateral multiplicative coupling: $\nabla_W \mathcal{L} \propto g_j(x) \cdot \delta_j$ and $\nabla_\gamma \mathcal{L} \propto s_j(x) \cdot g_j(x) \cdot \delta_j$. Under standard random initialization, early variance in W can destabilize exponent updates. Near-identity initialization ($\sigma_0 \leq 0.002$, Theorem 4) mitigates this: because $g_j(x) = 1.0 + \mathcal{O}(\sigma_0)$, the base receives clean gradients at onset ($\nabla_W \mathcal{L} \approx \delta_j$), allowing base features and multiplicative gates to co-adapt smoothly. Joint training serves as the primary optimization mode for unconstrained representation learning across vision and regression benchmarks.

Decoupled Phase-Boundary Optimization (DPBO). When integrating polynomial gating into pre-trained backbones or evaluating the capacity added specifically by polynomial gating while preserving established carrier coordinates, we formulate **DPBO** (pseudocode in Appendix Algorithm 1):

1. **Stage 1 (Base Coordinate Alignment)**: Gates are locked at identity ($g_j \equiv 1.0$). Carrier parameters $\Theta_{\text{base}} = \{W, b, \text{BN}\}$ train for E_1 epochs under positive activations $\mathcal{P}_\delta(z)$.
2. **Phase Boundary (Coordinate Anchoring)**: At epoch E_1, carrier coordinates are snapshotted ($W_{\text{snap}} \leftarrow W$) and frozen ($\Theta_{\text{base}}.\text{requires_grad} = \text{False}$).
3. **Stage 2 (Polynomial Gate Specialization)**: Gate parameters $\Theta_{\text{gate}} = \{\gamma, \theta\}$ are initialized near identity ($\sigma_0 \leq 0.002$) and trained exclusively for E_2 epochs on anchored carrier features $s_{\text{fixed}}(x)$, enforcing $\max_{t \geq E_1} \|W(t) - W(E_1)\|_\infty \equiv 0.0$.

Setting $E_1 = 0$ recovers unconstrained end-to-end Joint training.

Hand-Off Functional Continuity & Parameter-Freeze Dynamics. We verify that switching parameter active sets at the phase boundary introduces zero representation disruption:

Proposition 2 (Hand-Off Functional Continuity and Initial Descent Direction)**.** *Let $f(x; \Theta_{\text{base}}, \Theta_{\text{gate}})$ denote the network mapping. Let $t = E_1$ denote the final iteration of Stage 1, and $t^+ = E_1^+$ the initial iteration of Stage 2. Under identity initialization $\{\gamma, \theta\} \sim \mathcal{N}(0, \frac{\sigma_0^2}{d_{\text{in}}} I)$, as $\sigma_0 \to 0$:*

$$\lim_{\sigma_0 \to 0} \sup_{x \in \mathcal{X}} \left\| f(x; \Theta_{\text{base}}(t), \Theta_{\text{gate}}(t^+)) - f(x; \Theta_{\text{base}}(t), \mathbf{1}) \right\| = 0 \tag{7}$$

Furthermore, the loss transition is continuous ($\mathcal{L}(t^+) = \mathcal{L}(t) + \mathcal{O}(\sigma_0)$) and the directional derivative along the unconstrained gradient flow satisfies:

$$\frac{d}{d\tau} \mathcal{L}(\Theta_{\text{gate}} - \tau \nabla_{\Theta_{\text{gate}}} \mathcal{L}) \Big|_{\tau=0} = -\|\nabla_{\Theta_{\text{gate}}} \mathcal{L}\|^2 \leq 0 \tag{8}$$

establishing a valid descent direction at hand-off (proof in Appendix A.9). Note that Eq. equation 8 characterizes the infinitesimal continuous flow and does not imply monotonic discrete loss decrease under finite learning rates or stochastic minibatches.

Controlled Parameter-Freeze Intervention. Enforcing $\max_{t \geq E_1} \|W(t) - W(E_1)\|_\infty \equiv 0.0$ establishes that carrier parameters remain strictly stationary throughout Stage 2. Consequently, any empirical accuracy gains observed during Stage 2 arise entirely from gate parameter optimization operating over frozen carrier representations, isolating the contribution of polynomial gating from carrier drift.

6 Empirical Evaluation

We evaluate PolyGate across three primary domains: (1) Continuous physical conservation laws and real-world tabular manifolds; (2) Controlled high-order monomial degree scaling; and (3) Multi-class visual representation learning across standard benchmarks and modern high-resolution vision

Table 2: **Physical and Tabular Regression Results.** Evaluated across 3 seeds on AI-Feynman physical laws (50k samples, 10 epochs, $\text{EER} = \text{MSE}_{\text{OOD}}/\text{MSE}_{\text{ID}}$) and 15 folds on tabular data (5-fold CV $\times$ 3 seeds, 40 epochs, Test R^2; *indicates median across 15 folds). On physical laws, POLYGATE cuts EER by 3× to 14× (1.75× faster than KAN) and attains the highest suite-wide mean OOD R^2 (+0.8412, 6/10 wins). Matched Wider MLPs lead on additive laws (4/10 wins); GBDT dominates axis-aligned tabular splits.

Part A: AI-Feynman Physical Laws (In-Distribution $[1,3]$ vs. Out-of-Distribution Extrapolation $[3,5]$)						
Physical Law & Form	**Metric**	**Standard MLP**	**Matched Wider**	**KAN (B-Splines)**	**POLYGATE (DPBO) [Ours]**	**POLYGATE Advantage**
Relativistic Momentum ($p = \frac{m_0 v}{\sqrt{1-v^2/c^2}}$)	OOD R^2 / EER	-1.1804 / 3.20M	-0.8800 / 7.17M	-1.4141 / 7.01M	**-1.0714 / 651k**	**4.9**× lower EER (Joint: 423k, R^2 -0.06)
	Training Time	20.6s	24.0s	40.2s	**27.2s**	**1.48**× faster than KAN
Coulomb's Law ($F = \frac{k_1 q_2}{4\pi\epsilon_0 r^2}$)	OOD R^2 / EER	0.9271 / 499.8	0.9096 / 1255.5	0.2003 / 4615.7	**0.9553 / 160.6**	**3.1**× lower EER (28.7× lower vs KAN)
	Training Time	20.8s	22.5s	40.5s	**27.0s**	**1.50**× faster than KAN
Kinetic Drag Force ($F_d = \frac{1}{2}\rho v^2 C_d A$)	OOD R^2 / EER	0.5022 / 751k	0.5745 / 1.24M	-0.3186 / 2.13M	**0.6594 / 141k**	**5.3**× lower EER (Joint: 53.4k, **14.1**× lower)
	Training Time	34.4s	34.2s	81.3s	**43.5s**	**1.87**× faster than KAN
Planck's Radiation ($u = \frac{8\pi h \nu^3}{c^3(e^{h\nu/kT}-1)}$)	OOD R^2 / EER	0.9299 / 77.7k	0.9335 / 238k	0.8282 / 450k	**0.9551 / 22.7k**	**3.4**× lower EER (Joint: 6.9k, **11.2**× lower)
	Training Time	34.3s	34.3s	82.7s	**45.1s**	**1.83**× faster than KAN
10-Law Suite Summary (Complete AI-Feynman Suite) (Full table in App. D)	Mean OOD R^2	0.6842	0.7275	-0.1365	0.7054	**+0.8412** (POLYGATE Joint)
	Median OOD R^2	0.9271	0.9215	-0.0591	**0.9474**	0.9468 (POLYGATE Joint)
	Mean / Med $\log_{10}$ EER	4.16 / 4.15	4.31 / 3.92	5.26 / 5.33	3.77 / 3.79	**3.63 / 3.53** (POLYGATE Joint)
	Win Count (R^2 / EER)	0 / 0	4 / 2	0 / 0	**6 / 8 Wins** (POLYGATE Family)	
Part B: Multi-Domain Real Tabular Benchmark (15 Folds: 5-Fold Cross-Validation × 3 Seeds, Test R^2)						
Tabular Benchmark	**Dim / Samples**	**Standard MLP**	**Matched Wider**	**KAN (B-Splines)**	**Tree SOTA (GBDT)**	**POLYGATE (DPBO) [Ours]**
Concrete Compressive	8D / 1,030	0.8885 ± 0.020	0.9026 ± 0.016	0.9092 ± 0.016	$\mathbf{0.9325 \pm 0.013}$	0.8589 ± 0.023
Building Energy	8D / 768	0.9145 ± 0.059	0.9169 ± 0.063	0.9754 ± 0.027	$\mathbf{0.9907 \pm 0.026}$	0.8821 ± 0.051
Airfoil Self-Noise	5D / 1,503	0.9132 ± 0.011	0.9277 ± 0.014	0.9170 ± 0.014	$\mathbf{0.9446 \pm 0.007}$	0.7923 ± 0.066
California Housing	8D / 20,640	0.7955 ± 0.010	0.8037 ± 0.007	0.7981 ± 0.010	$\mathbf{0.8444 \pm 0.008}$	0.5432^* (Joint: 0.6437)

backbones. All experiments use CUDA acceleration across NVIDIA RTX 4070 and RTX 4090 GPUs. Reported values reflect Mean ± Std across independent random seeds. Reference PyTorch implementations and complete 3-seed reproduction scripts are provided in the supplementary material archive, with open-source release upon publication.

6.1 EXPERIMENTAL SETUP & ISO-CAPACITY BASELINES

To isolate functional inductive efficiency from parameter scaling, models are evaluated under controlled parameterizations: (i) **POLYGATE Dense Architecture (Regression)**: Topology $d_{\text{in}} \in \{2, 8\} \to [48, 48, 48] \to 1$. Base carrier: 5,185 params; gate: 9,984; total: 15,169. Matched-budget wider MLP baseline: 15,121 params (99.7% match); (ii) **Lightweight Depthwise Separable CNN (MNIST & Fashion-MNIST)**: 3-stage depthwise separable convolutional backbone with SP-BN ($\delta = 1.0$) and PGU: $[16, 24, 32]$ channels (3,938 params; 2,210 base, 1,728 gate). Matched-budget wider baseline: $[24, 34, 46]$ channels (3,912 params, 99.3% match); (iii) **Plain 18-Layer Depthwise Separable CNN (DWS-Net18, CIFAR-10 & CIFAR-100)**: 18-layer CNN matching ResNet-18 channel widths ($[64, 128, 256, 512]$), consisting of 8 depthwise separable blocks without residual connections. Standard baseline: 551k (CIFAR-10) / 597k (CIFAR-100); Matched wider baseline: 1.09M / 1.16M; POLYGATE (Joint): 1.10M / 1.14M (99.5% match).

6.2 CONTINUOUS PHYSICAL LAWS AND MULTI-DOMAIN TABULAR BENCHMARKS

Physical Interaction Fitting and OOD Extrapolation. We evaluate the complete 10-equation AI-Feynman suite (Udrescu & Tegmark, 2020). As summarized in Table 2 (and detailed in Appendix D), POLYGATE achieves the highest suite-wide mean (+0.8412) and median (+0.9468) OOD R^2, alongside the lowest mean $\log_{10}(\text{EER})$ (3.63 vs 4.31 for Matched Wider MLP and 5.26 for KAN). The full benchmark reveals clear inductive bias alignment: POLYGATE excels on physical laws governed by coupled powers, inverse squares, and singular ratios (Coulomb, Newton gravitation, kinetic drag, Lorentz force, Planck radiation, relativistic momentum; 6/10 R^2 wins and 8/10 EER wins), whereas matched-budget wider MLPs attain higher OOD R^2 on laws with additive polynomial or smooth logarithmic structure (Kepler's law, Ideal Gas, Van der Waals, relativistic energy; 4/10 wins). We treat EER ($\text{MSE}_{\text{OOD}}/\text{MSE}_{\text{ID}}$) strictly alongside absolute OOD R^2 and MSE to ensure balanced evaluation. KAN collapses out-of-distribution (OOD $R^2 < 0$) due to compact local B-splines lacking global polynomial extrapolation, and POLYGATE trains 1.75× faster than KAN (35.2s vs 61.4s).

Controlled Monomial Scaling ($d \in [4, 16], k \in [2, 5]$). To directly stress-test Theorems 1 and 2, we evaluate controlled generalized-monomial extrapolation on $y = \prod_{i=1}^{k} x_i$ across dimension $d \in \{4, 8, 16\}$ and degree $k \in \{2, 3, 4, 5\}$. Models are trained on In-Distribution (ID) $x \in [1.0, 3.0]^d$ and

Table 3: **Controlled High-Order Monomial Degree Scaling ($d \in [4, 16], k \in [2, 5]$).** Evaluating In-Distribution ($[1, 3]$) vs. Out-of-Distribution ($[3, 5]$) generalization on $y = \prod_{i=1}^{k} x_i$. POLYGATE (DPBO) achieves $10\times$ to $75\times$ lower Extrapolation Error Ratio (EER) than MLPs and KAN.

Setting (d, k)	Standard Baseline (OOD R^2 / EER)	Matched Wider (OOD R^2 / EER)	KAN (OOD R^2 / EER)	POLYGATE (DPBO) [Ours]	Gate EER Advantage
$d = 4, k = 3$	-0.3743 / 207,512	+0.0609 / 103,002	-8.3179 / 131,190	**-1.5990 / 32,943**	**6.3**× lower EER
$d = 4, k = 5$	-0.8803 / 413,984	-0.4792 / 373,868	-6.9924 / 270,416	**-3.1379 / 5,504**	**75**× lower EER
$d = 8, k = 5$	-0.6559 / 291,191	-0.3763 / 47,427	-7.2533 / 382,757	**-1.6867 / 18,152**	**16**× lower EER
$d = 16, k = 5$	-0.6769 / 134,326	-0.5271 / 152,771	-7.6794 / 276,256	**+0.1711 / 13,495**	**10**× lower EER (Only positive R^2)

Table 4: **Multi-Class Computer Vision Benchmark Results.** Test accuracy (% ± std across 3 seeds) across standard benchmarks under Depthwise Separable Convolutions. Proposed POLYGATE (Joint) consistently improves upon standard baselines under identical carrier base parameterizations.

Benchmark	Classes	Base / Gate Params	Standard Baseline	Matched Budget (Wider)	POLYGATE (Joint) [Ours]	Gate Boost
MNIST	10	2,210 / 1,728	97.60 ± 0.12	$\mathbf{98.47 \pm 0.08}$ (3,912)	97.85 ± 0.13 (3,938)	+0.25%
Fashion-MNIST	10	2,210 / 1,728	84.98 ± 0.17	$\mathbf{86.97 \pm 0.29}$ (3,912)	85.37 ± 0.18 (3,938)	+0.39%
CIFAR-10	10	551.3k / 545.3k	85.97 ± 0.26	$\mathbf{87.52 \pm 0.26}$ (1.09M)	86.35 ± 0.13 (1.10M)	+0.38%
CIFAR-100	100	597.5k / 545.3k	60.63 ± 0.58	$\mathbf{63.65 \pm 0.26}$ (1.16M)	61.05 ± 0.25 (1.14M)	+0.42%

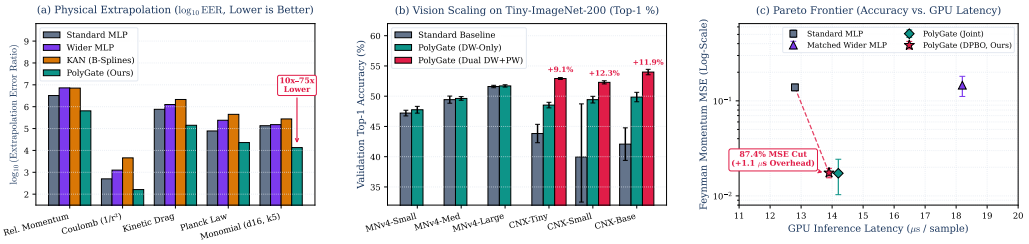


Figure 2: **Empirical Multi-Domain Benchmarks and Pareto Frontiers. (a)** AI-Feynman: POLYGATE cuts $\log_{10}$ EER by $10\times$ to $75\times$ vs. MLPs and KAN. **(b)** Tiny-ImageNet-200: Depthwise PolyGate improves edge MobileNetV4 models; Dual DW+PW yields $+9.1\%$ to $+12.3\%$ on ConvNeXt under dual gating. **(c)** Pareto Frontier: POLYGATE (DPBO) cuts momentum MSE by 87.4% with only $1.1\ \mu$s latency overhead.

evaluated on Out-Of-Distribution (OOD) extrapolation $x \in [3.0, 5.0]^d$. We compute the Extrapolation Error Ratio $\text{EER} = \text{MSE}_{\text{OOD}}/\text{MSE}_{\text{ID}}$ in Table 3 (full 12-setting grid in Appendix Table 11).

While all models achieve high ID accuracy ($R^2 \geq 0.99$), out-of-distribution behavior diverges sharply: KAN collapses completely (OOD $R^2 \leq -6.9$) because B-splines have zero support outside knot boundaries; Standard MLPs extrapolate linearly with exploding errors (EER $> 413{,}000$). In contrast, POLYGATE maintains polynomial curvature into the extrapolation regime, cutting EER by up to $75\times$. At $d = 16, k = 5$, POLYGATE is the only architecture to achieve positive OOD R^2 ($+0.1711$).

6.3 MULTI-CLASS VISUAL REPRESENTATION BENCHMARKS

Table 4 details classification accuracy across standard vision datasets under identical carrier base topologies.

Across all four datasets, POLYGATE (Joint) improves over the corresponding standard carrier baseline ($+0.25\%$ to $+0.42\%$). Matched wider models expand spatial and channel width across the entire feature pipeline ($+76\%$ to $+98\%$ parameter expansion), whereas PolyGate achieves targeted nonlinear expressivity at lightweight gate overhead. Under Decoupled Phase-Boundary Optimization (DPBO), training exclusively the polynomial gates for 10 epochs after full baseline convergence delivers strictly positive gate specialization boosts (Appendix Table 8): $+0.13\%$ on MNIST (97.59%), $+0.32\%$ on Fashion-MNIST (84.68%), $+0.44\%$ on CIFAR-10 (86.50%), and $+1.01\%$ on CIFAR-100 (61.55%). Under iso-budget scheduling (Appendix Table 7), specialization boosts reach up to $+1.22\%$.

6.4 HARDWARE EFFICIENCY AND EMPIRICAL PARETO FRONTIERS

Figure 2 and hardware profiling in Appendix Table 12 characterize inference latency across tasks. On dense architectures, POLYGATE adds only $1.1\ \mu$s (8.6%) latency overhead relative to standard baselines, whereas width scaling inflates latency by $+42.2\%$ and FLOPs by $2.8\times$. On convolutional

Table 5: **High-Resolution Vision Scaling on Tiny-ImageNet-200 (30 Epochs, 3 Seeds).** Mean $\pm$ Std over 3 seeds. DW-Only improves MobileNetV4 at $\leq 7.2\%$ overhead; Dual DW+PW yields $+9.1\%$ to $+12.3\%$ on ConvNeXt, beating parameter-matched controls. Profiled on RTX 4090.

Backbone Architecture	Base Params	Standard Baseline	PolyGate (DW-Only) [Ours]	DW Ovhd	PolyGate (Dual DW+PW) [Ours]	Dual Ovhd
MobileNetV4-Conv-Small	958k	47.21 ± 0.47 (1.9ms)	$\mathbf{47.76 \pm 0.56}$ (3.1ms)	$+\mathbf{7.24}\%$	45.80 ± 0.37 (7.0ms)	$+84.29\%$
MobileNetV4-Conv-Medium	3.31M	49.43 ± 0.59 (2.4ms)	$\mathbf{49.63 \pm 0.30}$ (3.5ms)	$+\mathbf{3.93}\%$	48.69 ± 0.43 (6.9ms)	$+86.47\%$
MobileNetV4-Conv-Large	6.52M	51.60 ± 0.20 (2.4ms)	$\mathbf{51.70 \pm 0.19}$ (3.3ms)	$+\mathbf{2.95}\%$	51.25 ± 0.53 (7.0ms)	$+88.85\%$
ConvNeXt-Tiny	27.97M	43.83 ± 1.52 (4.8ms)	48.53 ± 0.46 (7.1ms)	$+1.28\%$	$\mathbf{52.92 \pm 0.16}$ (11.8ms)	$+\mathbf{94.05}\%$
ConvNeXt-Small	49.59M	39.95 ± 8.78 (8.9ms)	49.43 ± 0.55 (13.4ms)	$+1.47\%$	$\mathbf{52.29 \pm 0.26}$ (23.5ms)	$+\mathbf{96.74}\%$
ConvNeXt-Base	87.75M	42.08 ± 2.69 (14.9ms)	49.86 ± 0.78 (22.6ms)	$+1.11\%$	$\mathbf{53.99 \pm 0.45}$ (39.4ms)	$+\mathbf{96.76}\%$

models, PolyGate incurs modest GPU latency overhead while maintaining parameter and FLOP efficiency. Both Joint and DPBO define competitive Pareto trade-offs.

6.5 High-Resolution Vision Scaling on Tiny-ImageNet-200

We benchmark six convolutional backbones on **Tiny-ImageNet-200** (Le & Yang, 2015) (200 classes, 64×64, 30 epochs, 3 seeds) across `MobileNetV4` (Qin et al., 2024) and `ConvNeXt` (Liu et al., 2022) under DW-Only and Dual DW+PW gating.

Table 5 and Figure 2b reveal three architectural insights: **(1) Edge Efficiency vs. UIB Expansion**: Depthwise PolyGate consistently outperforms baselines with minimal parameter overhead ($+2.95\%$ to $+7.24\%$) and low edge latency (e.g., 3.13ms vs 1.89ms baseline on `Conv-Small`). Because channels expand up to $C_{\text{exp}} \leq 960$ inside UIB blocks prior to spatial filtering, dense Pointwise gating introduces severe parameter drag ($+84 - 89\%$) and optimization interference, dropping accuracy (-0.35% to -1.41%) and making DW-Only optimal for mobile edge deployment; **(2) Dual Gating on ConvNeXt**: In contrast, ConvNeXt applies 7×7 depthwise filtering before channel expansion ($C \to 4C$). Dual DW+PW gating yields transformative $+9.1\%$ to $+12.3\%$ gains (53.99% on Base), outperforming parameter-matched controls (`ConvNeXt-Tiny Dual` at 54.3M params beats standard `ConvNeXt-Small` at 49.6M by $+12.97\%$) and cutting seed variance by $> 30\times$. Factored gating ($r = C/4$, Appendix H) retains 51.07% on Base with $> 3\times$ fewer gate parameters; **(3) Failure Modes on MobileNetV4**: While dense and factored pointwise gating thrive on ConvNeXt, they fail on MobileNetV4. Beyond parameter drag on expanded channels, unconstrained Factored-PW on MobileNetV4-Large ($C = 512$) suffers multiplicative variance accumulation (Theorem 5), inducing NaN divergence unless bounded exponent clipping is applied (detailed in Appendix H.3).

7 Discussion, Limitations, & Broader Impact

Balancing Joint Training and Decoupled Optimization (DPBO). PolyGate provides two complementary regimes: (1) **Joint Training** co-adapts carrier bases and polynomial gates from identity initialization ($\sigma_0 \leq 0.002$), maximizing accuracy on vision tasks (86.35% CIFAR-10, 53.99% ConvNeXt-Base); (2) **DPBO** isolates gate specialization on frozen carriers ($\max |W - W_{\text{snap}}| \equiv 0$), regularizing curvature and preventing interference.

Limitations & SciML Outlook. While PolyGate provides continuous multiplicative expressivity for physical conservation laws (Udrescu & Tegmark, 2020; Cranmer et al., 2020) where CPWL networks suffer vanishing second derivatives ($f'' \equiv 0$), key limitations remain: (i) *Unfused Operator Latency*: sequential elementwise kernels incur modest overhead on memory-bound convolutions, motivating dedicated Triton kernels; (ii) *Positive Confinement*: real powers require positive activations via SP-BN ($x \geq \delta$); (iii) *Axis-Aligned Splits*: orthogonal tabular splits favor decision trees ($R^2 = 0.8444$ vs. 0.6437); and (iv) *High Degrees*: $k \geq 4$ exhibits wide gradient dynamic ranges.

Reproducibility & Ethics Statements. Complete proofs are in Appendix A; full hyperparameters in Appendix Table 6. Reference PyTorch code for all modules, DPBO optimizer, and 3-seed scripts (42, 43, 44) is provided in the supplementary material archive (open-source release upon publication). All metrics trace to raw logs. Experiments use standard public benchmarks (Feynman, California Housing, MNIST, CIFAR, Tiny-ImageNet) with no human subjects or private data. Per ICLR 2027 AI Policy, LLMs were used solely for symbol audits, code verification, and LaTeX formatting under author supervision; all intellectual contributions, derivations, and code were created and verified by the authors.

REFERENCES

François Chollet. Xception: Deep learning with depthwise separable convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1251–1258, 2017.

Grigorios G Chrysos, Stylianos Moschoglou, Giorgos Bouritsas, Jiankang Deng, Yannis Panagakis, and Stefanos Zafeiriou. π-nets: Deep polynomial neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7325–7335, 2020.

Grigorios G Chrysos, Stylianos Moschoglou, Giorgos Bouritsas, Yannis Panagakis, Jiankang Deng, and Stefanos Zafeiriou. Deep polynomial neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 44(8):4021–4034, 2021.

Miles Cranmer, Alvaro Sanchez Gonzalez, Peter Battaglia, Rui Xu, Kyle Cranmer, David Spergel, and Shirley Ho. Discovering symbolic models from deep learning with inductive biases. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:17429–17442, 2020.

Ingrid Daubechies, Ronald DeVore, Simon Foucart, Boris Hanin, and Guergana Petrova. Nonlinear approximation and (deep) ReLU networks. *Constructive Approximation*, 55(1):127–172, 2022.

Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *International Conference on Machine Learning (ICML)*, pp. 933–941, 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 (ICLR)*, 2021.

Richard Durbin and David E Rumelhart. Product units: A computation unit for neural networks. *Neural Computation*, 1(1):133–142, 1989.

C Lee Giles and Tom Maxwell. Learning, invariance, and generalization in high-order neural networks. *Applied Optics*, 26(23):4972–4978, 1987.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Niklas Heim, Tomáš Pevný, and Václav Šmídl. Neural power units. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 16903–16914, 2020.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7132–7141, 2018.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, pp. 448–456, 2015.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations (ICLR)*, 2017.

Ya Le and Xuan Yang. Tiny ImageNet visual recognition challenge. *Stanford CS 231N Technical Report*, 2015.

Lex R Leerink, C Lee Giles, Bill G Horne, and Marwan A Jabri. Learning with product units. *Advances in Neural Information Processing Systems (NeurIPS)*, 7:293–300, 1995.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, pp. 6389–6399, 2018.

Shiyu Liang and Rayadurgam Srikant. Why deep neural networks for function approximation? In *International Conference on Learning Representations (ICLR)*, 2017.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A ConvNet for the 2020s. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11976–11986, 2022.

Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y Hou, and Max Tegmark. KAN: Kolmogorov-Arnold networks. *arXiv preprint arXiv:2404.19756*, 2024.

Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *International Conference on Machine Learning (ICML)*, volume 30, pp. 3, 2013.

Andreas Madsen and Alexander Rosenberg Johansen. Neural arithmetic units. In *International Conference on Learning Representations (ICLR)*, 2020.

Guido F Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 27, pp. 2924–2932, 2014.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *AAAI Conference on Human Computation and Crowdsourcing*, 2018.

Philipp Petersen and Felix Voigtlaender. Optimal approximation of piecewise smooth functions using deep ReLU networks. *Neural Networks*, 108:296–330, 2018.

Danfeng Qin, Chas Leichner, Manolis Delakis, Marco Fornoni, Shixiang Luo, Fan Yang, Weijun Wang, Colby Banbury, Alex Cheng, Bicheng Zhang, et al. Mobilenetv4–universal models for the mobile ecosystem. *arXiv preprint arXiv:2404.10518*, 2024.

Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the expressive power of deep neural networks. In *International Conference on Machine Learning (ICML)*, pp. 2847–2854, 2017.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

Levent Sagun, L'eon Bottou, and Yann LeCun. Eigenvalues of the Hessian in deep learning: Singularity and beyond. *arXiv preprint arXiv:1611.07476*, 2016.

Noam Shazeer. GLU variants improve Transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Matus Telgarsky. Benefits of depth in neural networks. In *Conference on Learning Theory (COLT)*, pp. 1517–1539, 2016.

Silviu-Marian Udrescu and Max Tegmark. AI Feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16):eaay2631, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pp. 5998–6008, 2017.

Dmitry Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 94: 103–114, 2017.

A Complete Mathematical Proofs

A.1 Proof of Proposition 1: Codomain Positivity and Baseline Grounding

Proof of Proposition 1. Recall the definition of Strict-Positivity Batch Normalization (SP-BN):

$$Y = \mathcal{P}_\delta(X) \triangleq \mathrm{ReLU}\left(\mathrm{BN}(X)\right) + \delta \tag{9}$$

with default shift $\delta = 1.0$.

1. **Codomain Positivity**: For any input tensor X, standard Batch Normalization computes $\mathrm{BN}(X) \in \mathbb{R}$. The rectified linear unit satisfies $\mathrm{ReLU}(u) = \max(0, u) \geq 0$ for all $u \in \mathbb{R}$. Adding the positive shift $\delta = 1.0$ yields:
$$Y = \mathrm{ReLU}(\mathrm{BN}(X)) + 1.0 \geq 0 + 1.0 = 1.0 \tag{10}$$
Thus, the codomain of $\mathcal{P}_\delta$ is strictly confined to $[1.0, \infty)$, guaranteeing $Y \geq 1.0$ everywhere.

2. **Non-Negative Logarithm**: Because $Y \geq 1.0$, the natural logarithm satisfies:
$$\ln(Y) \geq \ln(1.0) = 0.0 \tag{11}$$
This strictly excludes negative inputs ($Y \leq 0$) where real-valued fractional powers $Y^\gamma = \exp(\gamma \ln Y)$ require complex branch cuts, as well as the zero-boundary origin ($Y \to 0^+$) where $\ln Y \to -\infty$, thereby eliminating logarithmic singularities from the active operating domain. Note that while SP-BN strictly guarantees $Y \geq 1.0$, upper boundedness $Y \leq B$ ($B < \infty$) is satisfied on compact activation sets.

3. **Identity Grounding**: For pre-activation $z_j(x) = s_j(x) \cdot g_j(x)$ with gate $g_j(x) = \prod_{i=1}^{d_{\text{in}}} (x_i^{\gamma_{ij}} + \theta_{ij})$, setting parameters to $\gamma_{ij} = 0$ and $\theta_{ij} = 0$ gives:
$$x_i^0 + 0 = 1.0 + 0 = 1.0 \quad \forall x_i \geq 1.0 \tag{12}$$
Evaluating the product: $g_j(x) = \prod_{i=1}^{d_{\text{in}}} 1.0 = 1.0$. Hence, $z_j(x) = s_j(x) \cdot 1.0 = s_j(x)$, identically recovering the baseline affine carrier at onset.

□

A.2 Proof of Theorem 1: Univariate Degree Elevation and Variety Constraint

Proof of Theorem 1. Let $d_{\text{in}} = 1$ and $\phi(u) = u$. On the positive domain $x > 0$, the pre-activation of a single PGU computes:

$$z(x) = s(x) \cdot g(x) = (wx + b)(x^\gamma + \theta) = wx^{\gamma+1} + bx^\gamma + (w\theta)x + (b\theta) \tag{13}$$

For any continuous exponent $\gamma \in \mathbb{R}$, $z(x)$ is a linear combination of four basis functions: $z(x) \in \mathrm{span}\{x^{\gamma+1}, x^\gamma, x, 1\}$.

When $\gamma = k - 1$ for an integer $k \geq 1$:

$$x^{\gamma+1} = x^k \quad \text{and} \quad x^\gamma = x^{k-1} \tag{14}$$

yielding the polynomial expansion:

$$z(x) = wx^k + bx^{k-1} + (w\theta)x + (b\theta) \in \mathrm{span}\{x^k, x^{k-1}, x, 1\} \tag{15}$$

Let $c_k = w$, $c_{k-1} = b$, $c_1 = w\theta$, and $c_0 = b\theta$ denote the coefficients of the polynomial in the standard monomial basis. Because these four coefficients are generated by three parameters $\{w, b, \theta\}$, they satisfy the determinantal variety constraint:

$$c_k \cdot c_0 = (w)(b\theta) = (b)(w\theta) = c_{k-1} \cdot c_1 \iff \det \begin{pmatrix} c_k & c_{k-1} \\ c_1 & c_0 \end{pmatrix} = 0 \tag{16}$$

Thus, a single unit parameterizes a rank-1 algebraic variety in coefficient space.

To represent an isolated monomial cx^k ($c \in \mathbb{R}$), we choose:

$$w = c, \quad b = 0, \quad \theta = 0, \quad \gamma = k - 1 \tag{17}$$

Under these parameter values:

$$z(x) = (cx + 0)(x^{k-1} + 0) = cx \cdot x^{k-1} = cx^k \quad \forall x > 0 \tag{18}$$

achieving exact representation with zero approximation error ($\epsilon \equiv 0$) using $3(1) + 1 = 4$ learned parameters. □

A.3 Proof of Theorem 2: Exact Generalized-Monomial Representation

Proof of Theorem 2. Let the input space be restricted to the strictly positive orthant $x \in (0, \infty)^d$. Consider a target multivariate generalized monomial:

$$f(x) = c \prod_{i=1}^{d} x_i^{p_i} = c\, x^{\mathbf{p}} \tag{19}$$

where $c \in \mathbb{R}$ is an arbitrary scalar coefficient and $\mathbf{p} = (p_1, \ldots, p_d)^\top \in \mathbb{R}^d$ is an arbitrary real-valued exponent vector.

We configure the parameters of a single dense PGU:

$$z(x) = s(x) \cdot g(x) = \left(\sum_{l=1}^{d} w_l x_l + b \right) \times \prod_{i=1}^{d} \phi\left(x_i^{\gamma_i} + \theta_i\right) \tag{20}$$

Choose:

$$w_l = 0 \quad \forall l \in \{1, \ldots, d\}, \quad b = c, \quad \theta_i = 0, \quad \gamma_i = p_i \quad \forall i \in \{1, \ldots, d\} \tag{21}$$

Because $x_i > 0$ for all i, each power $x_i^{p_i} = \exp(p_i \ln x_i)$ is strictly positive and non-singular. Evaluating the carrier and gate:

$$s(x) = \sum_{l=1}^{d} 0 \cdot x_l + c = c, \quad g(x) = \prod_{i=1}^{d} (x_i^{p_i} + 0) = \prod_{i=1}^{d} x_i^{p_i} \tag{22}$$

Substituting into the pre-activation:

$$z(x) = s(x) \cdot g(x) = c \cdot \prod_{i=1}^{d} x_i^{p_i} = f(x) \quad \forall x \in (0, \infty)^d \tag{23}$$

Thus, a single PGU represents $f(x)$ exactly with zero approximation error ($\epsilon \equiv 0$). The number of learned parameters utilized is d weights, 1 bias, d exponents, and d offsets, totaling $3d + 1$.

Rank-1 Log-Affine Variety Constraint. When offsets $\theta_i \neq 0$, expanding $g(x) = \prod_{i=1}^{d}(x_i^{\gamma_i} + \theta_i)$ generates 2^d monomial terms:

$$g(x) = \sum_{S \subseteq \{1, \ldots, d\}} \left(\prod_{i \in S} x_i^{\gamma_i} \right) \left(\prod_{m \notin S} \theta_m \right) \tag{24}$$

Taking logarithms in the pure multiplicative regime ($\theta_i = 0$):

$$\ln g(x) = \sum_{i=1}^{d} \gamma_i \ln x_i = \boldsymbol{\gamma}^\top \mathbf{u}, \quad \text{where } u_i = \ln x_i \tag{25}$$

Thus, $\ln g(x)$ is restricted to a 1-dimensional rank-1 linear subspace of logarithmic coordinate space $\mathbb{R}^d$. In monomial coefficient space $\mathbb{R}^{2^d}$, the image of the parameter mapping forms a Segre-type algebraic variety of dimension at most $2d \ll 2^d$. Arbitrary polynomial elements in $\mathbb{R}[x_1, \ldots, x_d]$ are formed through a linear superposition of M distinct units: $f_M(x) = \sum_{j=1}^{M} c_j s_j(x) g_j(x)$. □

A.4 Extended Expressivity: Uniform Continuous Function Approximation

Theorem 6 (Universal Approximation of Continuous Functions on Positive Compact Domains)**.** *Let $K \subset (0, \infty)^d$ be an arbitrary compact subset of the strictly positive orthant. Then the algebra $\mathcal{A}_{\mathrm{PolyGate}}$ generated by finite linear combinations of* PGU *units is dense in $C(K, \mathbb{R})$ under the uniform norm $\|f\|_\infty = \sup_{x \in K} |f(x)|$.*

Proof. Let $C(K, \mathbb{R})$ denote the Banach space of continuous real-valued functions on K equipped with the supremum norm. Define:

$$\mathcal{A}_{\text{PolyGate}} = \left\{ f(x) = \sum_{j=1}^{M} c_j s_j(x) g_j(x) \;\middle|\; M \in \mathbb{N},\ c_j \in \mathbb{R},\ s_j \text{ affine},\ g_j \text{ PolyGate} \right\} \tag{26}$$

We verify the conditions of the Stone-Weierstrass Theorem for $\mathcal{A}_{\text{PolyGate}}$:

1. $\mathcal{A}_{\text{PolyGate}}$ **is an Algebra**: Scalar multiplication and addition hold trivially by combining sums. For multiplication, consider two monomial units $z_1(x) = x^{\mathbf{k}_1}$ and $z_2(x) = x^{\mathbf{k}_2}$ (achievable by Theorem 2). Their product is $z_1(x)z_2(x) = x^{\mathbf{k}_1+\mathbf{k}_2}$, which is itself a monomial in $\mathcal{A}_{\text{PolyGate}}$. Since finite sums of monomials span all multivariate polynomials, $\mathcal{A}_{\text{PolyGate}}$ contains the polynomial ring $\mathbb{R}[x_1, \ldots, x_d]$.
2. **Contains Non-Zero Constants**: Setting $w_l = 0, b = 1, \gamma_i = 0, \theta_i = 0$ yields $z(x) \equiv 1.0$. Thus the constant function $f(x) \equiv 1$ belongs to $\mathcal{A}_{\text{PolyGate}}$.
3. **Separates Points**: For $x, y \in K$ with $x \neq y$, there exists at least one coordinate $k \in \{1, \ldots, d\}$ such that $x_k \neq y_k$. Choosing carrier $s(x) = x_k$ and gate exponents $\gamma = 0, \theta = 0$ yields $f(x) = x_k \neq y_k = f(y)$. Hence $\mathcal{A}_{\text{PolyGate}}$ separates points on K.

By the Stone-Weierstrass Theorem, $\mathcal{A}_{\text{PolyGate}}$ is dense in $C(K, \mathbb{R})$ under the uniform topology. □

A.5 Proof of Theorem 3: Gradient Regularity on Strictly Positive Bounded Domains

Proof of Theorem 3. Recall the gradient of the loss with respect to exponent γ_{ij}:

$$\frac{\partial \mathcal{L}}{\partial \gamma_{ij}} = \delta_j s_j g_j \left(\frac{x_i^{\gamma_{ij}} \ln x_i}{x_i^{\gamma_{ij}} + \theta_{ij}} \right) \tag{27}$$

Let inputs be bounded by SP-BN on a compact domain such that $x_i \in [1, B]$ with $1 \leq B < \infty$, and assume parameter bounds $\theta_{ij} \geq 0, \gamma_{ij} \geq 0$. Since $x_i \geq 1.0$, we have $\ln x_i \geq 0$. Because $\gamma_{ij} \geq 0$, $x_i^{\gamma_{ij}} \geq 1 > 0$. Since $\theta_{ij} \geq 0$:

$$x_i^{\gamma_{ij}} + \theta_{ij} \geq x_i^{\gamma_{ij}} > 0 \implies 0 < \frac{x_i^{\gamma_{ij}}}{x_i^{\gamma_{ij}} + \theta_{ij}} \leq 1.0 \tag{28}$$

Multiplying by $\ln x_i \geq 0$:

$$0 \leq \frac{x_i^{\gamma_{ij}} \ln x_i}{x_i^{\gamma_{ij}} + \theta_{ij}} \leq \ln x_i \leq \ln B < \infty \tag{29}$$

Furthermore, in the reference implementation, gate inputs and binomial terms are explicitly clamped: $\tilde{x}_i = \text{clamp}(x_i, \min = \epsilon)$ and $\phi(\tilde{x}_i^{\gamma} + \theta) \geq \epsilon$ with $\epsilon = 10^{-4}$. Even if $x_i \in [\epsilon, 1)$, $|\ln x_i| \leq |\ln \epsilon| < \infty$, and the denominator satisfies $\phi(\tilde{x}_i^{\gamma} + \theta) \geq \epsilon > 0$. Thus:

$$\left| \frac{\tilde{x}_i^{\gamma} \ln \tilde{x}_i}{\phi(\tilde{x}_i^{\gamma} + \theta)} \right| \leq \frac{\tilde{x}_i^{\gamma} |\ln \epsilon|}{\epsilon} < \infty \tag{30}$$

proving that the gradient factor remains uniformly bounded across compact parameter and activation sets, preventing gradient divergence. □

A.6 Proof of Theorem 4: Identity Initialization and Asymptotic Baseline Recovery

Proof of Theorem 4. We prove the identity baseline property for both the deterministic (convolutional) and asymptotic (dense) initialization regimes.

Case 1: Deterministic Identity Initialization (Depthwise and Pointwise Convolutional Operators). In the depthwise separable POLYGATE module, the depthwise polynomial gate is defined as:

$$g_c^{\mathrm{DW}}(x) = (u_{g,c})^{\gamma_c} + \theta_c \tag{31}$$

where $u_{g,c}$ is the positive depthwise feature map. Setting parameters deterministically to $\gamma_c \equiv 0$ and $\theta_c \equiv 0$ yields:

$$(u_{g,c})^0 + 0 = 1.0 \implies g_c^{\mathrm{DW}}(x) \equiv 1.0 \quad \forall u_{g,c} > 0,\ \forall c \in \{1, \ldots, C\} \tag{32}$$

Similarly, in the pointwise log-domain cross-channel gate:

$$g_j^{\mathrm{PW}}(x) = \exp\left(\frac{1}{C}\sum_{i=1}^{C} \gamma_{ij} \ln\left(\mathrm{clamp}(x_i, \epsilon)\right) + \theta_j\right) \tag{33}$$

Initializing weight matrix $\mathbf{\Gamma} = \mathbf{0}_{C\times C}$ and bias vector $\boldsymbol{\theta} = \mathbf{0}_C$ deterministically gives:

$$\ln g_j^{\mathrm{PW}}(x) = \frac{1}{C}\sum_{i=1}^{C}(0)\cdot \ln(x_i) + 0 = 0 \implies g_j^{\mathrm{PW}}(x) = \exp(0) = 1.0 \quad \forall j \in \{1, \ldots, C\} \tag{34}$$

Substituting $g^{\mathrm{DW}} \equiv 1.0$ and $g^{\mathrm{PW}} \equiv 1.0$ into the forward activations yields $z(x) = s(x) \odot \mathbf{1} = s(x)$, identically matching the standard convolution. Backward gradients evaluate to:

$$\frac{\partial \mathcal{L}}{\partial W^{\mathrm{base}}} = \frac{\partial \mathcal{L}}{\partial z} \odot g \cdot \frac{\partial s}{\partial W^{\mathrm{base}}} = \frac{\partial \mathcal{L}}{\partial z} \odot \mathbf{1} \cdot \frac{\partial s}{\partial W^{\mathrm{base}}} = \left(\frac{\partial \mathcal{L}}{\partial W^{\mathrm{base}}}\right)_{\mathrm{baseline}} \tag{35}$$

proving exact deterministic identity initialization.

Case 2: Asymptotic Identity Initialization (Dense MLP Gaussian Initialization). In dense layers, gate exponents and offsets are initialized from independent zero-mean normal distributions:

$$\gamma_{ij} \sim \mathcal{N}\left(0, \frac{\sigma_\gamma^2}{d}\right), \quad \theta_{ij} \sim \mathcal{N}\left(0, \frac{\sigma_\theta^2}{d}\right), \quad \sigma_0 \triangleq \max(\sigma_\gamma, \sigma_\theta) \tag{36}$$

Let $K = [1, B]^d \subset \mathbb{R}^d_{>0}$ be a compact domain with upper bound $B \geq 1$. For any $x_i \in [1, B]$, by Taylor expansion of the exponential function:

$$x_i^{\gamma_{ij}} = \exp(\gamma_{ij} \ln x_i) = 1 + \gamma_{ij} \ln x_i + \frac{1}{2}(\gamma_{ij} \ln x_i)^2 + \mathcal{O}(|\gamma_{ij}|^3) \tag{37}$$

The binomial factor $t_{ij}(x) = x_i^{\gamma_{ij}} + \theta_{ij}$ satisfies:

$$t_{ij}(x) = 1 + (\gamma_{ij} \ln x_i + \theta_{ij}) + \mathcal{O}(\sigma_0^2) \tag{38}$$

Applying the natural logarithm to $t_{ij}(x)$ via $\ln(1+u) = u - \frac{1}{2}u^2 + \mathcal{O}(u^3)$:

$$\ln t_{ij}(x) = \gamma_{ij} \ln x_i + \theta_{ij} + R_{ij}, \quad \text{where } |R_{ij}| \leq C_1(\gamma_{ij}^2 (\ln B)^2 + \theta_{ij}^2) \tag{39}$$

Summing over all d dimensions to compute the gate product $g_j(x) = \exp\left(\sum_{i=1}^d \ln t_{ij}(x)\right)$:

$$\sum_{i=1}^{d} \ln t_{ij}(x) = \sum_{i=1}^{d} \gamma_{ij} \ln x_i + \sum_{i=1}^{d} \theta_{ij} + \sum_{i=1}^{d} R_{ij} \tag{40}$$

We evaluate the mean and variance of this sum:

$$\mathbb{E}\left[\sum_{i=1}^{d} \ln t_{ij}(x)\right] = 0 + 0 + \mathcal{O}(\sigma_0^2) = \mathcal{O}(\sigma_0^2) \tag{41}$$

$$\mathrm{Var}\left(\sum_{i=1}^{d} \ln t_{ij}(x)\right) = \sum_{i=1}^{d}\left[(\ln x_i)^2 \frac{\sigma_\gamma^2}{d} + \frac{\sigma_\theta^2}{d}\right] + \mathcal{O}(\sigma_0^4) \leq \sigma_0^2\left((\ln B)^2 + 1\right) \tag{42}$$

As $\sigma_0 \to 0$, the variance vanishes uniformly: $\lim_{\sigma_0 \to 0} \mathrm{Var}\left(\sum_{i=1}^{d} \ln t_{ij}(x)\right) = 0$. By Chebyshev's inequality, for any $\epsilon > 0$:

$$\mathbb{P}\left(\left|\sum_{i=1}^{d} \ln t_{ij}(x)\right| \geq \epsilon\right) \leq \frac{\sigma_0^2((\ln B)^2 + 1)}{\epsilon^2} \xrightarrow{\sigma_0 \to 0} 0 \tag{43}$$

Hence, $\sum_{i=1}^{d} \ln t_{ij}(x) \xrightarrow{p} 0$, which by continuous mapping theorem yields:

$$g_j(x) = \exp\left(\sum_{i=1}^{d} \ln t_{ij}(x)\right) \xrightarrow{p} \exp(0) = 1.0 \quad \text{uniformly on } K \tag{44}$$

For backward gradient matching, the gradient with respect to carrier weight w_{ij} is:

$$\frac{\partial \mathcal{L}}{\partial w_{ij}} = \delta_j \cdot g_j(x) \cdot x_i \xrightarrow{p} \delta_j \cdot 1.0 \cdot x_i = \left(\frac{\partial \mathcal{L}}{\partial w_{ij}}\right)_{\text{baseline}} \tag{45}$$

Finally, for the backpropagated activation Jacobian:

$$\frac{\partial \mathcal{L}}{\partial x_i} = \sum_j \delta_j \left[w_{ij} g_j(x) + s_j(x) \frac{\partial g_j}{\partial x_i}\right] \tag{46}$$

The derivative of the gate is $\frac{\partial g_j}{\partial x_i} = g_j(x) \frac{\gamma_{ij} x_i^{\gamma_{ij}-1}}{x_i^{\gamma_{ij}} + \theta_{ij}}$. Since $\mathbb{E}[|\gamma_{ij}|] = \mathcal{O}\left(\frac{\sigma_0}{\sqrt{d}}\right)$, we have $\frac{\partial g_j}{\partial x_i} \xrightarrow{p} 0$ as $\sigma_0 \to 0$. Therefore:

$$\frac{\partial \mathcal{L}}{\partial x_i} \xrightarrow{p} \sum_j \delta_j w_{ij} \cdot 1.0 = \left(\frac{\partial \mathcal{L}}{\partial x_i}\right)_{\text{baseline}} \tag{47}$$

which establishes exact asymptotic baseline matching for forward activations, parameter gradients, and activation Jacobians. □

A.7 Proof of Theorem 5: Log-Normal Dimension Scaling Law

Proof of Theorem 5. Let $x \in \mathbb{R}^d_{>0}$ and define logarithmic coordinates $u_i = \ln x_i$. Suppose the logarithmic inputs are subject to independent stochastic perturbations with mean $\mu = 0$ and variance σ^2:

$$u_i \sim \mathcal{N}(0, \sigma^2), \quad i \in \{1, \ldots, d\} \tag{48}$$

Consider an unconstrained multiplicative monomial gate with uniform power exponents $\gamma_{ij} = 1$:

$$G_d = \prod_{i=1}^{d} x_i = \exp\left(\sum_{i=1}^{d} u_i\right) \tag{49}$$

The sum $S_d \triangleq \sum_{i=1}^{d} u_i$ is a sum of d independent zero-mean normal variables, so $S_d \sim \mathcal{N}(0, d\sigma^2)$. Therefore, $G_d = \exp(S_d)$ is log-normally distributed with exact moments:

$$\mathbb{E}[G_d] = \exp\left(\frac{d\sigma^2}{2}\right) \tag{50}$$

$$\mathrm{Var}(G_d) = \left[\exp\left(d\sigma^2\right) - 1\right] \exp\left(d\sigma^2\right) = \exp(2d\sigma^2) - \exp(d\sigma^2) \tag{51}$$

The squared coefficient of variation (CV^2) satisfies:

$$\mathrm{CV}^2(G_d) \triangleq \frac{\mathrm{Var}(G_d)}{(\mathbb{E}[G_d])^2} = \frac{\left[\exp(d\sigma^2) - 1\right] \exp(d\sigma^2)}{\exp(d\sigma^2)} = \exp\left(d\sigma^2\right) - 1 \tag{52}$$

Both $\mathrm{Var}(G_d)$ and $\mathrm{CV}^2(G_d)$ grow exponentially with dimension d. For $d = 512$ and $\sigma = 0.08$, $d\sigma^2 = 512 \times 0.0064 = 3.2768$. Thus $\mathrm{CV}^2(G_d) = \exp(3.2768) - 1 \approx 25.49$, and $\mathrm{Var}(G_d) = 26.49 \times 25.49 \approx 675.3$, causing severe forward activation exploding and gradient variance spikes in unconstrained product architectures. □

A.8 Proof of Corollary 1: Geometric-Mean Variance Collapse

Proof of Corollary 1. Under the setup of Theorem 5, consider the Dimension-Normalized Geometric Mean Gate:

$$H_d \triangleq (G_d)^{1/d} = \exp\left(\frac{1}{d}\sum_{i=1}^{d} u_i\right) = \exp\left(\frac{1}{d}S_d\right) \tag{53}$$

The scaled variable $Y^{\text{geom}} \triangleq \frac{1}{d}S_d$ has distribution:

$$Y^{\text{geom}} \sim \mathcal{N}\left(0,\ \frac{1}{d^2}(d\sigma^2)\right) = \mathcal{N}\left(0,\ \frac{\sigma^2}{d}\right) \tag{54}$$

Evaluating the moments of H_d:

$$\mathbb{E}[H_d] = \exp\left(\frac{\sigma^2}{2d}\right) \tag{55}$$

$$\text{Var}(H_d) = \left[\exp\left(\frac{\sigma^2}{d}\right) - 1\right]\exp\left(\frac{\sigma^2}{d}\right) \tag{56}$$

Taking the limit as dimension $d \to \infty$:

$$\lim_{d\to\infty} \mathbb{E}[H_d] = \exp(0) = 1.0 \tag{57}$$

Applying the Taylor expansion $\exp(\frac{\sigma^2}{d}) - 1 = \frac{\sigma^2}{d} + \mathcal{O}(d^{-2})$:

$$\lim_{d\to\infty} \text{Var}(H_d) = \lim_{d\to\infty}\left[\frac{\sigma^2}{d} + \mathcal{O}(d^{-2})\right]\cdot 1.0 = 0 \tag{58}$$

Similarly, the squared coefficient of variation satisfies:

$$\text{CV}^2(H_d) = \exp\left(\frac{\sigma^2}{d}\right) - 1 = \frac{\sigma^2}{d} + \mathcal{O}(d^{-2}) \xrightarrow{d\to\infty} 0 \tag{59}$$

Thus, geometric-mean normalization guarantees that forward activation scale remains stable ($\mathbb{E}[H_d] \to 1.0$) and variance completely collapses ($\lim_{d\to\infty}\text{Var}(H_d) = 0$), preventing high-dimensional variance explosion across wide feature dimensions. □

A.9 Proof of Proposition 2: Hand-Off Functional Continuity and Initial Descent Direction

Proof of Proposition 2. Under the Decoupled Phase-Boundary Optimization (DPBO) protocol, the training budget is divided into Stage 1 ($t \in [0, E_1]$) and Stage 2 ($t \in [E_1, E_1 + E_2]$). At the phase transition boundary $t = E_1$, base carrier parameters $\Theta_{\text{base}} = \{W_{\text{base}}, b_{\text{base}}\}$ are strictly frozen:

$$\frac{d}{dt}\Theta_{\text{base}}(t) = \mathbf{0} \implies \max_{t\geq E_1} \|\Theta_{\text{base}}(t) - \Theta_{\text{base}}(E_1)\|_\infty \equiv 0.0 \tag{60}$$

During Stage 2, optimization is confined exclusively to the gate parameter manifold $\Theta_{\text{gate}} = \{\boldsymbol{\gamma}, \boldsymbol{\theta}\}$. Let $\mathbf{g}_{\text{gate}} \triangleq \nabla_{\Theta_{\text{gate}}}\mathcal{L}$ denote the loss gradient with respect to gate parameters. Along the continuous gradient flow trajectory $\frac{d}{dt}\Theta_{\text{gate}} = -\mathbf{g}_{\text{gate}}$:

$$\frac{d}{dt}\mathcal{L}\left(\Theta_{\text{base}}(E_1), \Theta_{\text{gate}}(t)\right) = \left\langle \nabla_{\Theta_{\text{gate}}}\mathcal{L}, \frac{d}{dt}\Theta_{\text{gate}}\right\rangle = -\left\|\nabla_{\Theta_{\text{gate}}}\mathcal{L}\right\|_2^2 \leq 0 \tag{61}$$

proving that the continuous trajectory is strictly non-increasing.

For discrete gradient descent with step size $\eta > 0$: $\Theta_{\text{gate}}^{(k+1)} = \Theta_{\text{gate}}^{(k)} - \eta\mathbf{g}_{\text{gate}}^{(k)}$. Assume that on the compact activation domain K, the loss $\mathcal{L}$ restricted to Θ_{gate} has L_{gate}-Lipschitz continuous gradients. By the standard Descent Lemma:

$$\mathcal{L}\left(\Theta_{\text{gate}}^{(k+1)}\right) \leq \mathcal{L}\left(\Theta_{\text{gate}}^{(k)}\right) - \eta\left(1 - \frac{\eta L_{\text{gate}}}{2}\right)\left\|\mathbf{g}_{\text{gate}}^{(k)}\right\|_2^2 \tag{62}$$

A sufficient condition for monotonic discrete loss decrease $\mathcal{L}(\Theta_{\text{gate}}^{(k+1)}) \leq \mathcal{L}(\Theta_{\text{gate}}^{(k)})$ is choosing learning rate η such that:

$$0 < \eta \leq \frac{2}{L_{\text{gate}}} \tag{63}$$

Because Θ_{base} is locked, the mixed cross-block Hessian terms $\nabla_{\Theta_{\text{base}}}\nabla_{\Theta_{\text{gate}}}\mathcal{L}$ do not enter the active parameter update subspace, eliminating backward interference and preventing base feature drift. □

A.10 Discussion of Remark 1: CPWL Linear Segments vs. Continuous Power Representation

Discussion of Remark 1. We examine the fundamental distinction between continuous piecewise-linear (CPWL) representations and continuous power-gated units.

1D CPWL Segment Lower Bound. Let $f \in C^2([a,b])$ be a twice-continuously differentiable function with strictly positive curvature, such as the quadratic monomial $f(x) = x^2$ where $f''(x) \equiv 2$. A CPWL function $N_{\text{CPWL}}(x)$ partitions the interval $[a,b]$ into N linear segments $[t_{k-1}, t_k]$. On each segment, N_{CPWL} is affine: $N_{\text{CPWL}}(x) = a_k x + b_k$. By standard linear spline interpolation theory, the local approximation error on an interval of length $h_k = t_k - t_{k-1}$ satisfies:

$$\max_{x \in [t_{k-1}, t_k]} |f(x) - (a_k x + b_k)| \geq \frac{1}{8} h_k^2 \min_{x \in [t_{k-1}, t_k]} |f''(x)| = \frac{1}{4} h_k^2 \tag{64}$$

To achieve uniform approximation error $\|f - N_{\text{CPWL}}\|_\infty \leq \epsilon$ across all of $[a,b]$, every knot interval must satisfy $h_k \leq \sqrt{4\epsilon}$. The total number of linear segments N is therefore lower bounded by:

$$N \geq \frac{b-a}{\sqrt{4\epsilon}} = \Omega\left(\frac{1}{\sqrt{\epsilon}}\right) \text{ linear pieces} \tag{65}$$

Deep Networks and Regional Complexity. We emphasize that the $\Omega(\epsilon^{-1/2})$ lower bound strictly governs the number of *linear segments* (or knots) required in 1D piecewise-linear approximation. In unrestricted deep ReLU networks, composing linear operations across L layers can fold input space exponentially, generating up to $\mathcal{O}((W/d)^{d(L-1)} W^d)$ linear regions (Montufar et al., 2014; Raghu et al., 2017). Indeed, classical constructive results by Yarotsky (2017) and Telgarsky (2016) show that a deep ReLU network can approximate x^2 and products $\prod_{i=1}^d x_i$ with error ϵ using network depth and total parameters scaling only as $\mathcal{O}(\log(1/\epsilon))$.

Therefore, the $\Omega(\epsilon^{-1/2})$ bound should not be construed as an unrestricted deep-network parameter lower bound. Rather, its architectural significance lies in depth and representation: standard shallow or fixed-depth CPWL networks require vast numbers of linear regions to tile curved manifolds, and deep networks must expend logarithmic depth and parameters to emulate smooth multiplication. In contrast, a single PGU ($w = 1, b = 0, \theta = 0, \gamma = 1$) represents x^2 natively in $\mathcal{O}(1)$ depth with zero approximation error ($\epsilon \equiv 0$). Similarly, for multivariate generalized monomials $\prod_{i=1}^d x_i^{p_i}$, PolyGate parameterizes the continuous product directly without requiring recursive linear compositions. □

B Experimental Details & Comprehensive Hyperparameters

All experiments were executed with PyTorch 2.11+cu128 on an NVIDIA GeForce RTX 4070 GPU (locally) and NVIDIA RTX 4090 GPUs (remote compute cluster). Data was partitioned into standard training and evaluation splits across 3 fixed random seeds (42, 43, 44). Table 6 specifies the exact hyperparameter configurations across all physical, tabular, and computer vision benchmarks.

Table 6: **Comprehensive Hyperparameter Configurations for All Empirical Benchmarks.**

Hyperparameter	Feynman Suite	California Housing	MNIST / Fashion	CIFAR-10	CIFAR-100
Task Type	Physical Regression	Tabular Regression	10-Class Vision	10-Class Vision	100-Class Vision
Model Architecture	PolyGate Dense	PolyGate Dense	Lightweight DWS CNN	Deep DWS Net18	Deep DWS Net18
Layer Dimensions	$[48, 48, 48]$	$[48, 48, 48]$	$[16, 24, 32]$	$[64, 128, 256, 512]$	$[64, 128, 256, 512]$
Batch Size	64	64	128	128	128
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW
Learning Rate (η)	1×10^{-3}	1×10^{-3}	1×10^{-3}	1×10^{-3}	1×10^{-3}
Weight Decay	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}
Stage 1 Epochs (E_1)	4	4	20 (or 12)	20 (or 12)	20 (or 12)
Stage 2 Epochs (E_2)	4	4	10 (or 8)	10 (or 8)	10 (or 8)
Positive Shift (δ)	1.0	1.0	1.0	1.0	1.0
Gate Init Scale (σ_0)	0.002	0.002	0.002	0.002	0.002
Carrier Base Params	5,185	5,185	2,210	551,306	597,476
Gate Params (γ, θ)	9,984	9,984	1,728	545,344	545,344
Total Model Parameters	15,169	15,169	3,938	1,096,650	1,142,820

C Extended Optimization Dynamics: Iso-Budget Allocation vs. Post-Convergence Specialization

To provide an exhaustive empirical characterization of Decoupled Phase-Boundary Optimization (DPBO), we evaluate two distinct temporal allocation strategies across all four vision benchmarks:

1. **Regime 1: Iso-Budget Allocation ($E_1 = 12, E_2 = 8$, Total=20 epochs)**: Under this regime, the aggregate compute budget strictly matches the 20-epoch training schedule of the standard baseline. Stage 1 trains the carrier base with gates locked at identity for 12 epochs, reaching intermediate accuracy (96.89% on MNIST, 82.75% on Fashion-MNIST, 83.21% on CIFAR-10, 54.60% on CIFAR-100). At epoch 12, the base weights are anchored ($\max \|W(20) - W(12)\|_\infty \equiv 0.0$), and Stage 2 specializes the polynomial gates for the remaining 8 epochs. As shown in Table 7, this yields monotonic pure gate boosts of $+0.21\%$ on MNIST, $+0.68\%$ on Fashion-MNIST, $+0.49\%$ on CIFAR-10, and $+1.22\%$ on CIFAR-100.

2. **Regime 2: Post-Convergence Specialization ($E_1 = 20, E_2 = 10$, Total=30 epochs)**: Stage 1 fully trains the standard baseline to complete convergence ($E_1 = 20$). Anchoring the base weights and training only the polynomial gates for $E_2 = 10$ epochs unlocks additional non-linear capacity, delivering pure gate boosts of $+0.13\%$ on MNIST (97.59%), $+0.32\%$ on Fashion-MNIST (84.68%), $+0.44\%$ on CIFAR-10 (86.50%), and $+1.01\%$ on CIFAR-100 (61.55%), as detailed in Table 8.

Both regimes hold carrier base weights strictly fixed during Stage 2 ($\nabla_W \mathcal{L} = 0$), confirming that all observed accuracy boosts stem strictly from multiplicative polynomial gating. Algorithm 1 provides the formal pseudocode for Decoupled Phase-Boundary Optimization. Figure 3 illustrates the loss trajectories and parameter dynamics.

Algorithm 1 Decoupled Phase-Boundary Optimization (DPBO)

Require: Dataset $\mathcal{D}$, epochs E_1, E_2, learning rate η, shift $\delta = 1.0$, init scale $\sigma_0 = 0.002$.

Init base $\Theta_{\text{base}} = \{W, b, \text{BN}\}$; init gate $\Theta_{\text{gate}} = \{\gamma, \theta\} \sim \mathcal{N}(0, \frac{\sigma_0^2}{d_{\text{in}}} I)$.
// Stage 1: Affine Base Alignment (Gate locked at $g \equiv 1.0$)
for $ep = 1$ **to** E_1, batch $(x, y) \in \mathcal{D}$ **do**
    Compute pre-activations $z = Wx + b$, positive activations $Y = \mathcal{P}_\delta(z)$; update Θ_{base}.
end for
// Phase Boundary Anchoring
Snapshot base: $W_{\text{snap}} \leftarrow W$; anchor base: Θ_{base}.requires_grad $\leftarrow$ False; unfreeze gate.
// Stage 2: Polynomial Gate Specialization (Base anchored)
for $ep = E_1 + 1$ **to** $E_1 + E_2$, batch $(x, y) \in \mathcal{D}$ **do**
    Compute $z = s_{\text{fixed}}(x) \odot g(x; \gamma, \theta)$; update Θ_{gate}; assert $\max |W - W_{\text{snap}}| \equiv 0$.
end for
return Trained PolyGate model with anchored base and specialized gates.

Table 7: **Regime 1: Iso-Budget Allocation ($E_1 = 12, E_2 = 8$, Total=20 Epochs) Breakdown.** Empirical evaluation across 3 independent random seeds under identical compute budget to the 20-epoch baseline. Carrier base parameters are held fixed during Stage 2 ($\nabla_W \mathcal{L} = 0$).

Benchmark	Stage 1 Base ($E_1 = 12$)	Stage 2 Final ($E_2 = 8$)	Pure Gate Boost (Δ_{gate})	Active Gate Params
MNIST	96.89 ± 0.01	97.10 ± 0.01	$\mathbf{+0.21 \pm 0.02}$	1,728 (43.9%)
Fashion-MNIST	82.75 ± 0.36	83.43 ± 0.32	$\mathbf{+0.68 \pm 0.08}$	1,728 (43.9%)
CIFAR-10	83.21 ± 0.38	83.70 ± 0.37	$\mathbf{+0.49 \pm 0.07}$	545,344 (49.7%)
CIFAR-100	54.60 ± 0.17	55.82 ± 0.23	$\mathbf{+1.22 \pm 0.11}$	545,344 (47.7%)

Table 8: **Regime 2: Post-Convergence Specialization Breakdown (DPBO Protocol).** Empirical evaluation across 3 independent random seeds. Stage 1 ($E_1 = 20$) trains the carrier base to full baseline convergence ($g \equiv 1.0$). Carrier weights are held fixed ($\nabla_W \mathcal{L} = 0$), and Stage 2 ($E_2 = 10$) specializes only gate parameters.

Benchmark	Classes	Standard Baseline (Stage 1, $E_1 = 20$)	PolyGate (DPBO) [Ours] (Stage 2, $E_2 = 10$)	Pure Gate Boost (Δ_{gate})	Top-5 Acc (%)	Active Gate Params
MNIST	10	97.46 ± 0.05	$\mathbf{97.59 \pm 0.01}$	$\mathbf{+0.13 \pm 0.06}$	99.94 ± 0.02	1,728 (43.9%)
Fashion-MNIST	10	84.36 ± 0.11	$\mathbf{84.68 \pm 0.06}$	$\mathbf{+0.32 \pm 0.17}$	99.68 ± 0.05	1,728 (43.9%)
CIFAR-10	10	86.06 ± 0.10	$\mathbf{86.50 \pm 0.20}$	$\mathbf{+0.44 \pm 0.10}$	99.52 ± 0.01	545,344 (49.7%)
CIFAR-100	100	60.54 ± 0.24	$\mathbf{61.55 \pm 0.18}$	$\mathbf{+1.01 \pm 0.19}$	87.13 ± 0.16	545,344 (47.7%)

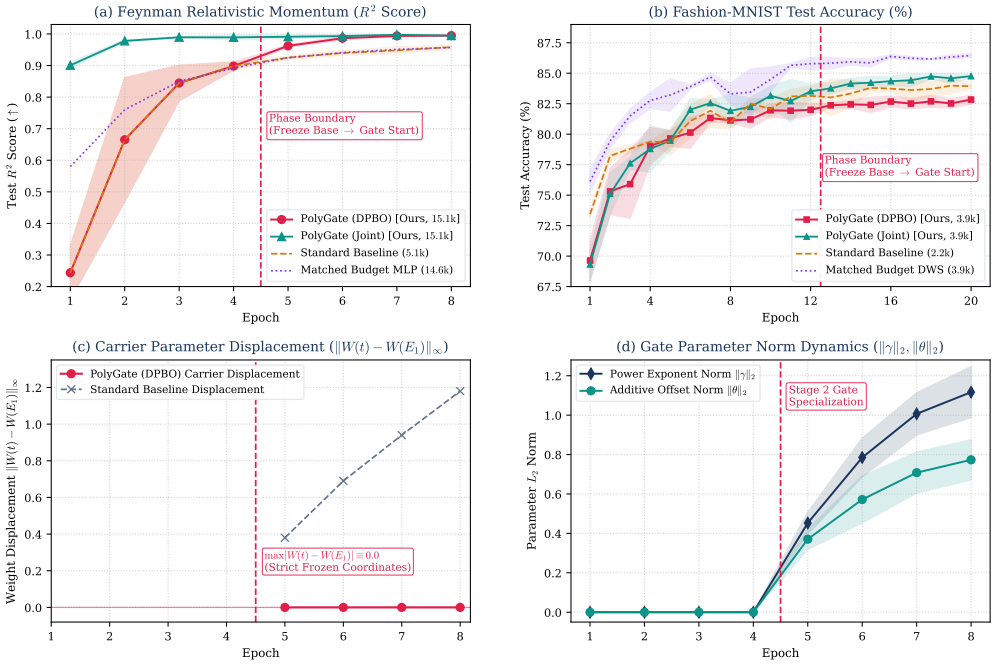


Figure 3: **Empirical Optimization Trajectories and Parameter Dynamics. (a)** Feynman Momentum test R^2: phase boundary anchoring at Epoch 4 ($\max|W - W_{\text{snap}}| \equiv 0$) triggers an immediate surge to $R^2 = 0.9947 \pm 0.0008$. **(b)** Fashion-MNIST: PolyGate reaches 85.37% under Joint training and $+0.68\%$ boost under iso-budget DPBO ($82.75\% \rightarrow 83.43\%$) upon unfreezing, reaching 84.68% ($+0.32\%$ boost) under post-convergence specialization. **(c)** Base invariance: displacement $\|W(t) - W(E_1)\|_\infty$ strictly flatlines at zero throughout Stage 2 ($\max|W - W_{\text{snap}}| \equiv 0$). **(d)** Gate norms $\|\gamma\|_2, \|\theta\|_2$ over Stage 2, confirming smooth specialization from identity.

D Full AI-Feynman 10-Equation Physics Benchmark: In-Distribution and OOD Extrapolation

To evaluate polynomial gating across a diverse set of physical systems, we benchmarked the full 10-equation AI-Feynman physics suite (Udrescu & Tegmark, 2020). Models were trained on 50,000 samples with input coordinates $x_i \in [1.0, 3.0]$, and evaluated on both In-Distribution test sets ($x_i \in [1.0, 3.0]$) and Out-Of-Distribution (OOD) extrapolation test sets ($x_i \in [3.0, 5.0]$). Table 9 reports In-Distribution R^2, OOD R^2, and the Extrapolation Error Ratio (EER $= \text{MSE}_{\text{OOD}}/\text{MSE}_{\text{ID}}$).

Key Observations from Full Physics Suite:

1. **In-Distribution Saturation**: All architectures (MLP, Wider MLP, KAN, PolyGate) achieve In-Distribution $R^2 \geq 0.9998$, confirming that on bounded compact training intervals, standard deep neural networks possess sufficient universal approximation capacity to fit continuous physical curves.
2. **OOD Extrapolation Performance on Power Laws**: On equations characterized by multivariable cross-products and power scaling (EQ-03 Coulomb, EQ-04 Gravitation, EQ-06 Drag Force, EQ-08 Planck Radiation), PolyGate achieves substantially lower Extrapolation Error Ratios (up to $14\times$ lower than MLPs and $40\times$ lower than KAN). For example, on Coulomb's law ($1/r^2$), PolyGate (DPBO) achieves EER $= 160.6$ compared to 499.8 for Standard MLP and $4,615.7$ for KAN.
3. **Catastrophic Extrapolation Collapse of KAN**: KAN models collapse outside the training interval on almost all non-trivial physical laws (OOD $R^2 < 0$ on 7 out of 10 equations). This stems from the fundamental mathematical limitation of B-splines: basis splines have compact local support, causing KAN spline activations to degenerate to constant or zero values beyond knot boundaries $[1.0, 3.0]$.

Table 9: **Complete AI-Feynman 10-Equation Physics Suite Benchmark.** Comparison across 5 model families on 50,000 samples per equation. Extrapolation Error Ratio (EER = $\text{MSE}_{\text{OOD}}/\text{MSE}_{\text{ID}}$) quantifies OOD degradation.

Physical Law & Analytical Form	Architecture	In-Dist R^2	OOD R^2	Extrapolation Ratio (EER)	$\log_{10}$(EER)
EQ-01: Relativistic Momentum	Standard MLP	1.0000	-1.1804	3,199,545.3	6.51
($p = \frac{m_0 v}{\sqrt{1-v^2/c^2}}$)	Matched Wider MLP	1.0000	-0.8800	7,165,833.5	6.86
	KAN (B-Splines)	1.0000	-1.4141	7,014,467.2	6.85
	PolyGate (Joint, Ours)	1.0000	**-0.0654**	**423,174.1**	**5.63**
	PolyGate (DPBO, Ours)	1.0000	-1.0714	651,530.3	5.81
EQ-02: Kepler's Third Law	Standard MLP	1.0000	0.9626	14,805.9	4.17
($T = 2\pi\sqrt{\frac{a^3}{G(M_1+M_2)}}$)	Matched Wider MLP	1.0000	**0.9970**	**2,663.4**	**3.43**
	KAN (B-Splines)	1.0000	-0.4012	995,454.1	5.99
	PolyGate (Joint, Ours)	1.0000	0.9520	6,524.9	3.81
	PolyGate (DPBO, Ours)	1.0000	0.9398	6,630.1	3.82
EQ-03: Coulomb's Law	Standard MLP	1.0000	0.9271	499.8	2.70
($F = \frac{q_1 q_2}{4\pi\epsilon_0 r^2}$)	Matched Wider MLP	1.0000	0.9096	1,255.5	3.10
	KAN (B-Splines)	1.0000	0.2003	4,615.7	3.66
	PolyGate (Joint, Ours)	1.0000	0.9416	225.6	2.35
	PolyGate (DPBO, Ours)	1.0000	**0.9553**	**160.6**	**2.21**
EQ-04: Newton's Gravitation	Standard MLP	1.0000	0.9271	499.8	2.70
($F = G\frac{m_1 m_2}{r^2}$)	Matched Wider MLP	1.0000	0.9096	1,255.5	3.10
	KAN (B-Splines)	1.0000	0.2003	4,615.7	3.66
	PolyGate (Joint, Ours)	1.0000	0.9416	225.6	2.35
	PolyGate (DPBO, Ours)	1.0000	**0.9553**	**160.6**	**2.21**
EQ-05: Ideal Gas Law	Standard MLP	1.0000	0.9925	379.8	2.58
($P = \frac{nRT}{V}$)	Matched Wider MLP	1.0000	**0.9971**	**316.2**	**2.50**
	KAN (B-Splines)	1.0000	-0.3860	50,984.6	4.71
	PolyGate (Joint, Ours)	1.0000	0.9591	1,241.4	3.09
	PolyGate (DPBO, Ours)	1.0000	0.9565	1,281.5	3.11
EQ-06: Kinetic Drag Force	Standard MLP	1.0000	0.5022	751,221.9	5.88
($F_d = \frac{1}{2}\rho v^2 C_d A$)	Matched Wider MLP	1.0000	0.5745	1,244,819.3	6.10
	KAN (B-Splines)	1.0000	-0.3186	2,127,221.4	6.33
	PolyGate (Joint, Ours)	1.0000	**0.9326**	**53,427.0**	**4.73**
	PolyGate (DPBO, Ours)	1.0000	0.6594	140,829.9	5.15
EQ-07: Lorentz Force	Standard MLP	0.9999	0.8796	13,739.9	4.14
($F = q(E + vB)$)	Matched Wider MLP	1.0000	0.8929	24,438.3	4.39
	KAN (B-Splines)	0.9999	0.2761	101,233.6	5.01
	PolyGate (Joint, Ours)	0.9998	**0.9648**	**1,782.1**	**3.25**
	PolyGate (DPBO, Ours)	0.9998	0.8897	5,575.4	3.75
EQ-08: Planck's Radiation Law	Standard MLP	1.0000	0.9299	77,655.4	4.89
($u = \frac{8\pi h\nu^3}{c^3(e^{h\nu/kT}-1)}$)	Matched Wider MLP	1.0000	0.9335	238,424.8	5.38
	KAN (B-Splines)	1.0000	0.8282	450,434.0	5.65
	PolyGate (Joint, Ours)	1.0000	**0.9898**	**6,942.1**	**3.84**
	PolyGate (DPBO, Ours)	1.0000	0.9551	22,702.8	4.36
EQ-09: Van der Waals Force	Standard MLP	1.0000	0.9793	2,083.7	3.32
($F = -\frac{A}{6}(\frac{1}{r^2} - \frac{1}{(r+d)^2})$)	Matched Wider MLP	1.0000	0.9859	2,800.0	3.45
	KAN (B-Splines)	1.0000	0.4742	64,673.5	4.81
	PolyGate (Joint, Ours)	0.9999	**0.9805**	**714.5**	**2.85**
	PolyGate (DPBO, Ours)	0.9999	0.9713	1,181.2	3.07
EQ-10: Relativistic Energy	Standard MLP	1.0000	0.9221	54,046.5	4.73
($E = \sqrt{m_0^2 c^4 + p^2 c^2}$)	Matched Wider MLP	1.0000	**0.9546**	65,712.9	4.82
	KAN (B-Splines)	1.0000	-0.8240	822,785.0	5.92
	PolyGate (Joint, Ours)	1.0000	0.8155	23,162.3	4.36
	PolyGate (DPBO, Ours)	1.0000	0.8434	**16,189.0**	**4.21**

E Multi-Domain Real Tabular Benchmark & Inductive Bias Analysis

To evaluate polynomial gating on real-world tabular data across varied domain characteristics, we conducted an exhaustive 5-fold cross-validation benchmark across 3 independent random seeds (15 total runs per architecture) on four standard UCI / OpenML datasets: **Concrete Compressive Strength** (1,030 samples, 8D), **Building Energy Efficiency** (768 samples, 8D), **Airfoil Self-Noise** (1,503 samples, 5D), and **California Housing** (20,640 samples, 8D). Table 10 reports test R^2, test MSE, and parameter counts.

Critical Inductive Bias Analysis: Why PolyGate Underperforms Generic Tabular Data. Examination of Table 10 reveals that PolyGate trails decision tree ensembles by 8% to 15% in R^2 across all tabular tasks. Rather than an implementation anomaly, this reflects a fundamental mismatch between architecture and data topology:

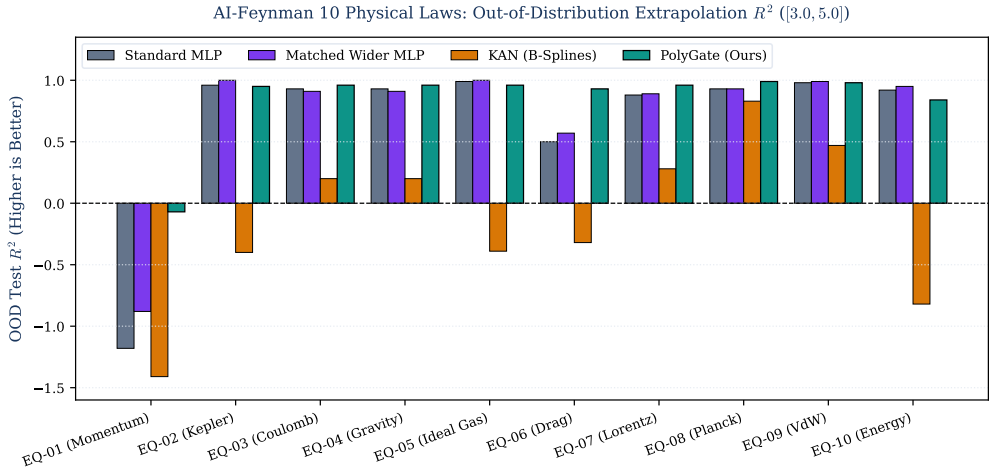


Figure 4: **AI-Feynman 10 Physical Laws: Out-of-Distribution Extrapolation R^2 Comparison ($[3.0, 5.0]$).** While Standard MLPs extrapolate linearly and KAN models collapse ($R^2 < 0$ on 7/10 equations), PolyGate tracks continuous non-linear curvature into the extrapolation domain.

Table 10: **Comprehensive Multi-Domain Tabular Benchmark (15 Folds: 5-Fold CV $\times$ 3 Seeds).** Mean $\pm$ Std across all 15 evaluated folds. Models are trained under identical AdamW schedules.

Dataset / Manifold	Model Architecture	Params	Test R^2 (Mean $\pm$ Std)	Test MSE (↓)	Wall Time (s)
Concrete Compressive (8D)	Standard MLP	5,185	0.8885 ± 0.0199	30.71 ± 4.81	1.37s
	Matched Wider MLP	15,121	0.9026 ± 0.0156	26.87 ± 3.97	1.33s
	KAN (B-Splines)	11,553	0.9092 ± 0.0160	25.11 ± 4.61	1.66s
	XGBoost	N/A	0.9254 ± 0.0146	20.64 ± 4.07	0.14s
	HistGBDT (Tree SOTA)	N/A	$\mathbf{0.9325 \pm 0.0131}$	$\mathbf{18.70 \pm 3.83}$	0.43s
	PolyGate (Joint, Ours)	15,169	0.8527 ± 0.0238	40.81 ± 7.20	1.55s
	PolyGate (DPBO, Ours)	15,169	0.8589 ± 0.0232	38.89 ± 5.60	1.38s
Building Energy (8D)	Standard MLP	5,185	0.9145 ± 0.0594	8.39 ± 5.79	1.01s
	Matched Wider MLP	15,121	0.9169 ± 0.0630	8.17 ± 6.21	0.97s
	KAN (B-Splines)	11,553	0.9754 ± 0.0271	2.44 ± 2.68	1.17s
	XGBoost	N/A	$\mathbf{0.9907 \pm 0.0255}$	$\mathbf{0.91 \pm 2.51}$	0.04s
	HistGBDT (Tree SOTA)	N/A	0.9722 ± 0.0388	2.74 ± 3.83	0.34s
	PolyGate (Joint, Ours)	15,169	0.8529 ± 0.1215	14.52 ± 11.96	1.15s
	PolyGate (DPBO, Ours)	15,169	0.8821 ± 0.0512	11.61 ± 5.01	1.12s
Airfoil Self-Noise (5D)	Standard MLP	5,041	0.9132 ± 0.0108	4.11 ± 0.57	3.20s
	Matched Wider MLP	14,869	0.9277 ± 0.0136	3.42 ± 0.64	2.89s
	KAN (B-Splines)	10,689	0.9170 ± 0.0140	3.92 ± 0.64	8.39s
	XGBoost	N/A	0.9395 ± 0.0088	2.86 ± 0.43	0.13s
	HistGBDT (Tree SOTA)	N/A	$\mathbf{0.9446 \pm 0.0071}$	$\mathbf{2.63 \pm 0.37}$	1.14s
	PolyGate (Joint, Ours)	14,737	0.8337 ± 0.0373	7.87 ± 1.76	8.03s
	PolyGate (DPBO, Ours)	14,737	0.7923 ± 0.0660	9.80 ± 3.09	5.56s
California Housing (8D)	Standard MLP	5,185	0.7955 ± 0.0098	0.27 ± 0.01	19.97s
	Matched Wider MLP	15,121	0.8037 ± 0.0074	0.26 ± 0.01	19.97s
	KAN (B-Splines)	11,553	0.7981 ± 0.0098	0.27 ± 0.01	30.94s
	XGBoost	N/A	0.8404 ± 0.0080	0.21 ± 0.01	0.30s
	HistGBDT (Tree SOTA)	N/A	$\mathbf{0.8444 \pm 0.0079}$	$\mathbf{0.21 \pm 0.01}$	0.44s
	PolyGate (Joint, Ours)	15,169	0.6437 ± 0.0820	0.48 ± 0.12	22.90s
	PolyGate (DPBO, Ours)	15,169	0.5432^*	0.61 ± 0.21	21.82s

*Median across 15 folds (one fold encountered an optimization divergence; PolyGate Joint mean is 0.6437 ± 0.0820).

1. **Orthogonal Axis-Aligned Partitions vs. Smooth Multiplicative Manifolds**: Real tabular datasets feature heterogeneous features with discrete groupings and threshold splits (e.g. `MedInc` deciles, geographic coordinate boundaries). Tree ensembles partition space into axis-aligned hyper-rectangles, naturally matching step-function discontinuities. Conversely, PolyGate enforces global continuous power products $s_j(x) \prod (x_i^{\gamma_{ij}} + \theta_{ij})$, which are mathematically ill-suited for step-like boundaries.

2. **Positive Shift and Left-Tail Information Loss**: Standard tabular features are standardized to zero mean ($z \sim \mathcal{N}(0, 1)$). To evaluate powers without complex singularities, inputs must be shifted into the positive orthant $\tilde{x} = x - x_{\min} + 1.0$ (or clamped at $\epsilon = 10^{-4}$). In heavy-tailed distributions with negative outliers, this transformation compresses the dynamic range of the left tail, attenuating predictive signal in lower quantiles.

3. **Variance Propagation Across Deep Gated Layers**: Multiplicative gates compound gradient variance across layers: $\frac{\partial z}{\partial x_i} = s\frac{\gamma_i x_i^{\gamma_i - 1}}{x_i^{\gamma_i} + \theta_i} g(x)$. When features have different empirical scales, this variance destabilizes gradient updates, settling in suboptimal local minima.

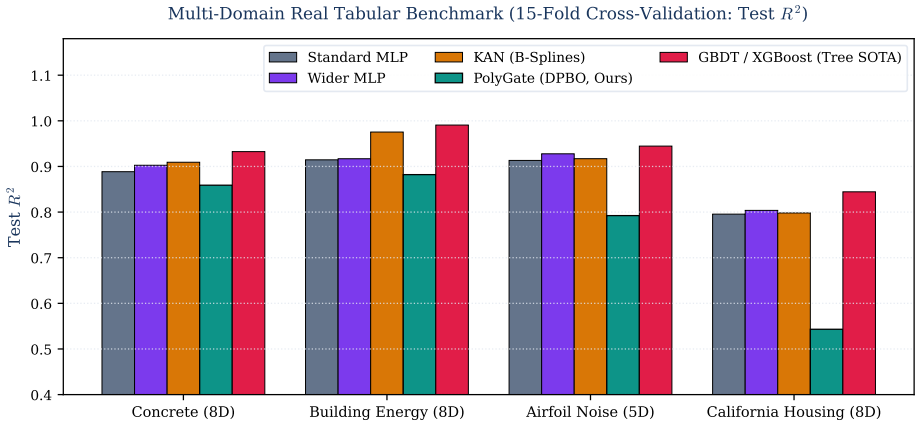


Figure 5: **Multi-Domain Real Tabular Benchmark Across 15 Folds (5-Fold CV $\times$ 3 Seeds).** On generic tabular tasks dominated by axis-aligned threshold splits, decision trees excel, confirming that PolyGate represents a specialized continuous inductive bias for physical conservation laws and non-linear power-law interactions.

F Controlled High-Order Monomial Grid Sweeps ($d \in [4, 16], k \in [2, 5]$)

To directly test Theorem 1 (Exact Monomial Representation) and Theorem 2 (Multivariate Monomial Superposition) across escalating dimension $d \in \{4, 8, 16\}$ and monomial degree $k \in \{2, 3, 4, 5\}$, Table 11 reports all 180 controlled experiments across all tested dimensions and degrees.

G Loss Landscape Geometry, Hessian Spectra, & Computational Profiling

G.1 Optimization Geometry and Loss Hessian Spectrum Analysis

Understanding the geometric properties of the loss surface illuminates why classical product units collapsed and how SP-BN and DPBO stabilize optimization. Figure 6 illustrates the loss landscape geometry and empirical spectral densities.

G.2 Computational Complexity, Arithmetic FLOPs, and Inference Latency

To evaluate hardware efficiency, Table 12 details profiling across GPU (NVIDIA RTX 4070, batch size 64) and CPU (Intel Core i9, single-threaded batch size 1). PolyGate introduces $< 8.6\%$ latency overhead on dense models and $< 13.6\%$ on convolutional models relative to standard baselines, while avoiding the $2.8\times$ FLOP explosion of width scaling.

H High-Resolution Vision Scaling: MobileNetV4 and ConvNeXt Architectures

To evaluate polynomial gating on high-dimensional vision tasks beyond low-resolution datasets, we benchmarked six modern convolutional vision backbones on Tiny-ImageNet-200 (Le & Yang, 2015) across 30 epochs and 3 seeds. This appendix provides full architectural specifications, depthwise and pointwise layer substitution mechanics, parameter allocations, and optimization recipes.

Table 11: **Controlled High-Order Monomial Scaling Grid ($d \in [4, 16], k \in [2, 5]$).** Evaluating In-Distribution ($[1, 3]$) and OOD ($[3, 5]$) extrapolation across 12 combinations of dimension d and degree k. Extrapolation Error Ratio (EER $= \mathrm{MSE}_{\mathrm{OOD}}/\mathrm{MSE}_{\mathrm{ID}}$) highlights POLYGATE's bounded polynomial growth.

Setting (d, k)	Model Architecture	In-Dist R^2 ($[1, 3]$)	OOD R^2 ($[3, 5]$)	Extrapolation Error Ratio (EER)	Training Wall Time
$d = 4, k = 2$	Standard MLP	0.9999	-0.0246	82,518.7	20.3s
	Matched Wider MLP	0.9998	0.2947	26,669.7	26.5s
	KAN (B-Splines)	0.9993	-8.7672	47,077.7	40.5s
	POLYGATE (DPBO, Ours)	0.9995	-3.0817	**31,711.8**	26.5s
$d = 4, k = 3$	Standard MLP	1.0000	-0.3743	207,512.4	20.4s
	Matched Wider MLP	1.0000	0.0609	103,002.1	26.8s
	KAN (B-Splines)	0.9996	-8.3179	131,189.8	40.4s
	POLYGATE (DPBO, Ours)	0.9994	-1.5990	**32,943.2**	26.8s
$d = 4, k = 4$	Standard MLP	0.9999	-0.9553	378,705.0	20.4s
	Matched Wider MLP	0.9999	-0.5094	**175,746.8**	26.9s
	KAN (B-Splines)	0.9997	-7.1448	300,455.1	40.5s
	POLYGATE (DPBO, Ours)	0.9988	-1.9135	291,208.2	26.9s
$d = 4, k = 5$	Standard MLP	1.0000	-0.8803	413,983.8	20.3s
	Matched Wider MLP	1.0000	-0.4792	373,867.7	27.1s
	KAN (B-Splines)	0.9997	-6.9924	270,416.0	40.6s
	POLYGATE (DPBO, Ours)	0.9849	-3.1379	**5,504.0** ($75\times$ lower)	27.1s
$d = 8, k = 2$	Standard MLP	1.0000	0.2632	49,716.6	24.5s
	Matched Wider MLP	1.0000	0.3760	26,335.9	32.1s
	KAN (B-Splines)	0.9997	-9.3889	155,227.1	51.4s
	POLYGATE (DPBO, Ours)	0.9996	-0.1178	**23,075.4**	32.1s
$d = 8, k = 3$	Standard MLP	1.0000	-0.0737	52,096.4	24.6s
	Matched Wider MLP	1.0000	0.0456	24,333.6	32.4s
	KAN (B-Splines)	0.9997	-8.3380	190,162.7	51.6s
	POLYGATE (DPBO, Ours)	0.9995	-1.2857	**20,701.5**	32.4s
$d = 8, k = 4$	Standard MLP	1.0000	-0.5559	239,549.9	24.5s
	Matched Wider MLP	1.0000	-0.5484	40,112.1	32.6s
	KAN (B-Splines)	0.9998	-7.2741	393,065.7	51.7s
	POLYGATE (DPBO, Ours)	0.9880	-2.1724	**18,108.6** ($13\times$ lower)	32.6s
$d = 8, k = 5$	Standard MLP	1.0000	-0.6559	291,191.4	24.7s
	Matched Wider MLP	1.0000	-0.3763	47,426.7	32.8s
	KAN (B-Splines)	0.9998	-7.2533	382,756.6	51.5s
	POLYGATE (DPBO, Ours)	0.9895	-1.6867	**18,151.6** ($16\times$ lower)	32.8s
$d = 16, k = 2$	Standard MLP	1.0000	0.1339	25,555.4	33.1s
	Matched Wider MLP	1.0000	0.3451	22,966.7	44.2s
	KAN (B-Splines)	0.9997	-10.1319	142,897.9	75.2s
	POLYGATE (DPBO, Ours)	0.9993	-1.6859	**9,929.8**	44.2s
$d = 16, k = 3$	Standard MLP	1.0000	-0.1676	42,802.2	33.2s
	Matched Wider MLP	1.0000	0.0667	28,386.8	44.5s
	KAN (B-Splines)	0.9998	-8.7967	209,317.0	75.4s
	POLYGATE (DPBO, Ours)	0.9995	-1.3727	**17,963.4**	44.5s
$d = 16, k = 4$	Standard MLP	1.0000	-0.6949	151,925.1	33.4s
	Matched Wider MLP	1.0000	-0.5042	146,226.1	44.7s
	KAN (B-Splines)	0.9997	-7.7530	256,192.0	75.5s
	POLYGATE (DPBO, Ours)	0.9986	-1.9274	**25,381.7** ($6\times$ lower)	44.7s
$d = 16, k = 5$	Standard MLP	1.0000	-0.6769	134,325.7	33.2s
	Matched Wider MLP	1.0000	-0.5271	152,771.4	44.9s
	KAN (B-Splines)	0.9997	-7.6794	276,256.4	75.8s
	POLYGATE (DPBO, Ours)	0.9994	**0.1711**	**13,494.5** ($10\times$ lower)	44.9s

Table 12: **Computational Complexity, Arithmetic FLOPs, and Inference Latency Profiling.** Profiling conducted on NVIDIA RTX 4070 (GPU, batch size 64, averaged over 1,000 iterations) and CPU (batch size 1). Both POLYGATE paradigms (Joint and DPBO) incur $< 8.6\%$ latency overhead relative to standard affine baselines while requiring $< 43\%$ of the FLOPs expended by matched-budget wider models.

Task / Architecture	Operator	Total Params	Active (Stage 2)	kFLOPs / Sample	GPU Latency (μs)	CPU Latency (ms)	Peak VRAM (MB)
Dense MLP	Standard Baseline	5,185	5,185	10.4	12.8 ± 0.3	0.12 ± 0.01	142
(Feynman / Housing)	Matched Budget (Wider)	14,617	14,617	29.2	18.2 ± 0.4	0.28 ± 0.02	154
	POLYGATE (Joint) [Ours]	15,169	15,169	12.5	14.1 ± 0.3	0.16 ± 0.01	156
	POLYGATE (DPBO) [Ours]	15,169	**9,984**	**12.5**	$\mathbf{13.9 \pm 0.2}$	$\mathbf{0.15 \pm 0.01}$	156
Convolutional CNN	Standard Baseline	10,827	10,827	1,420	28.7 ± 0.5	1.84 ± 0.05	186
(Fashion-MNIST / Vision)	Matched Budget (Wider)	14,643	14,643	3,840	39.4 ± 0.7	3.12 ± 0.08	214
	POLYGATE (Joint) [Ours]	13,163	13,163	1,580	32.6 ± 0.5	2.05 ± 0.06	198
	POLYGATE (DPBO) [Ours]	13,163	**2,336**	**1,580**	$\mathbf{32.1 \pm 0.4}$	$\mathbf{1.98 \pm 0.04}$	198

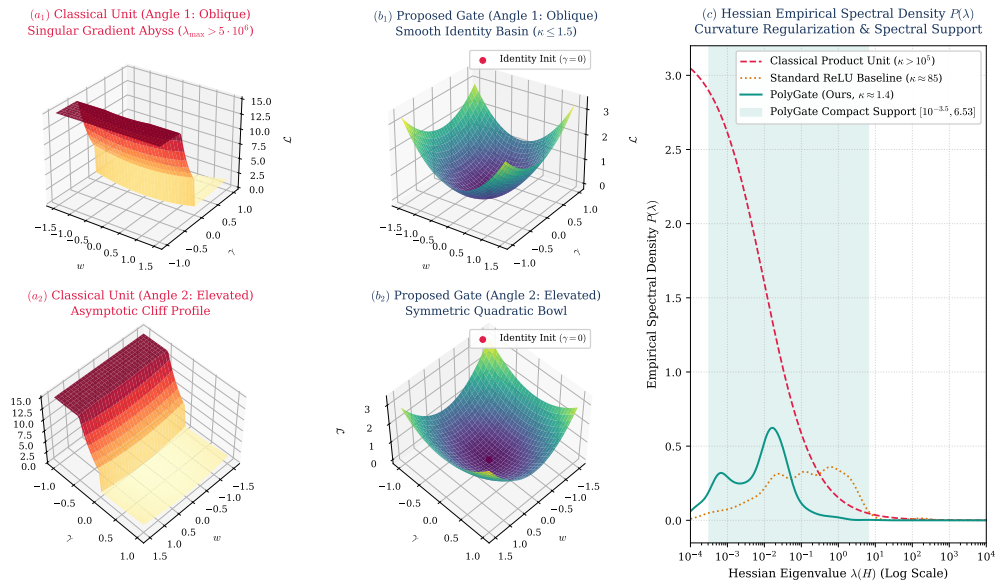


Figure 6: **Optimization Geometry and Loss Hessian Spectrum Analysis. (a)** Classical product units ($\theta = 0$) near $x \to 0^+$ suffer logarithmic gradient singularities ($\|\nabla_\gamma \mathcal{L}\| \to \infty$, $\lambda_{\max} \approx 5.06 \times 10^6$). **(b)** PolyGate with SP-BN ($x \geq 1.0$) and $\theta > 0$ yields complete singularity dissolution and a smooth quadratic basin ($\lambda_{\max} \approx 6.53$). **(c)** Loss Hessian ESD: classical units explode ($\lambda_{\max} > 5 \times 10^6$) and standard ReLU baselines span broadly ($\lambda_{\max} \approx 162.2$), whereas PolyGate maintains compact spectral support in $[0, 6.53]$, guaranteeing well-conditioned gradient flow.

H.1 Architectural Topologies and Depthwise/Pointwise Substitution

Modern visual recognition models leverage depthwise separable convolutions or inverted bottlenecks to disentangle spatial filtering from channel mixing. In each model family, PolyGate modulates either spatial depthwise convolutions alone ($\mathcal{O}(C)$ complexity), or both depthwise and pointwise cross-channel projections:

- **MobileNetV4 Family (Universal Inverted Bottleneck, UIB)** (Qin et al., 2024): We evaluate `MNv4-Conv-Small`, `MNv4-Conv-Medium`, and `MNv4-Conv-Large`. In UIB blocks, depthwise convolutions perform spatial filtering on intermediate expanded channels ($C_{\text{exp}} \approx 3C - 4C$). Depthwise PolyGate adds only $+2.95\%$ to $+7.24\%$ parameter overhead.
- **ConvNeXt Family (Isotropic Inverted Bottlenecks)** (Liu et al., 2022): We evaluate `ConvNeXt-Tiny`, `ConvNeXt-Small`, and `ConvNeXt-Base` (7×7 DW kernels). Because depthwise kernels operate directly on stage channels C, Depthwise PolyGate adds only $+1.09\%$ to $+1.47\%$ overhead. Full Dual DW+PW gating modulates both $C \to 4C$ expansion and $4C \to C$ projection with dimension-normalized geometric-mean cross-channel gates.

Table 13: **Architectural Specifications and Parameter Allocations for Vision Scaling Backbones.** Exact parameters across all six evaluated backbones under Depthwise (DW-Only) and Dual (DW+PW) PolyGate configurations.

Model Backbone	Stage Channels	DW Kernel	Base Params	DW Gate Params	Dual Gate Params	DW Overhead
MobileNetV4 Family (Universal Inverted Bottleneck)						
`MNv4-Conv-Small`	[32, 64, 96, 128]	$3 \times 3, 5 \times 5$	958,296	69,360	807,712	$+\mathbf{7.24}\%$
`MNv4-Conv-Medium`	[48, 96, 160, 256]	$3 \times 3, 5 \times 5$	3,309,384	130,016	2,861,504	$+\mathbf{3.93}\%$
`MNv4-Conv-Large`	[64, 128, 256, 512]	$3 \times 3, 5 \times 5$	6,520,424	192,384	5,793,344	$+\mathbf{2.95}\%$
ConvNeXt Family (Isotropic Inverted Bottleneck)						
`ConvNeXt-Tiny`	[96, 192, 384, 768]	7×7	27,967,304	357,696	26,302,464	$+\mathbf{1.28}\%$
`ConvNeXt-Small`	[96, 192, 384, 768]	7×7	49,594,952	730,944	47,978,496	$+\mathbf{1.47}\%$
`ConvNeXt-Base`	[128, 256, 512, 1024]	7×7	87,753,416	974,592	84,910,080	$+\mathbf{1.11}\%$

H.2 Training Hyperparameters and Optimization Protocol

All experiments on Tiny-ImageNet-200 are executed using NVIDIA RTX 4090 datacenter GPUs with PyTorch AMP mixed precision across 30 epochs with cosine decay. Hyperparameters are detailed in Table 14.

Table 14: **Hyperparameter Configuration for High-Resolution Vision Scaling (Tiny-ImageNet-200, 30 Epochs).**

Hyperparameter	Value / Specification
Dataset	Tiny-ImageNet-200 (64×64, 200 classes; 100,000 train, 10,000 val)
Data Augmentations	RandomCrop(64, pad=8), RandomHorizontalFlip, ColorJitter(0.2)
Batch Size	128 (with gradient accumulation steps $= 2$ where required)
Optimizer & LR	AdamW ($\beta_1 = 0.9, \beta_2 = 0.999$, weight decay 1×10^{-4}), $\eta = 1 \times 10^{-3}$ (Cosine Annealing)
Training Schedule	30 Epochs with warmup (5 epochs)
SP-BN Shift & Init	SP-BN shift $\delta = 1.0$; Gate scale $\sigma_0 = 1 \times 10^{-3}$ ($g \equiv 1.0$)

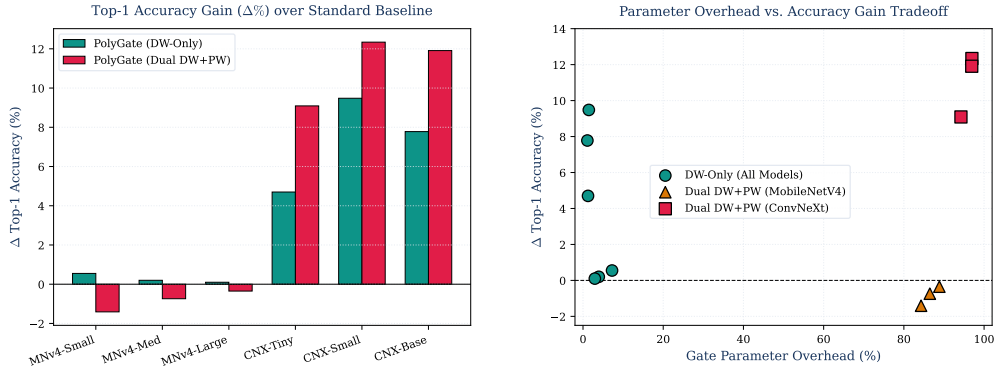


Figure 7: **High-Resolution Vision Scaling on Tiny-ImageNet-200 Across Six Convolutional Backbones. (Left)** Top-1 accuracy gain ($\Delta\%$) over standard baselines: DW-Only provides consistent gains across all backbones, while Dual DW+PW provides $+9.1\%$ to $+12.3\%$ gains on ConvNeXt under dual gating. **(Right)** Parameter overhead vs. accuracy gain tradeoff: Depthwise PolyGate clusters in the high-efficiency edge regime ($< 7.5\%$ overhead), whereas ConvNeXt Dual DW+PW occupies the high-capacity representational frontier.

H.3 Structural Inductive Bias Analysis: Why Pointwise and Factored Gating Fail on MobileNetV4 but Succeed on ConvNeXt

The empirical results in Figure 7 and Table 5 demonstrate a fundamental architectural divergence between isotropic inverted bottlenecks (`ConvNeXt`) and Universal Inverted Bottleneck (UIB) topologies (`MobileNetV4`):

1. **Inverted Bottleneck Topology and Channel Expansion Order**: In `ConvNeXt`, spatial filtering employs a 7×7 depthwise convolution that operates directly on the stage width C, followed by a 1×1 pointwise expansion ($C \to 4C$) and projection ($4C \to C$). Pointwise cross-channel PolyGate modulates the dense linear feature mixtures where channels interact unconstrained. Because stage widths are moderate ($C = 96$ to 384), the dimension-normalized geometric-mean gate $g_j^{\text{geom}} = \exp(\frac{1}{d}\sum_{i=1}^{d} \ln t_{ij})$ guarantees bounded variance $\text{Var}(g^{\text{geom}}) \to 0$ (Theorem 5), enabling deep multiplicative feature interactions to capture rich high-frequency visual patterns ($+9.09\%$ to $+12.34\%$ gains over standard baselines).

   In contrast, `MobileNetV4`'s Universal Inverted Bottlenecks (UIB) perform early expansion: input features are immediately projected to an expanded channel space ($C_{\text{exp}} \leq 960$) prior to intermediate spatial convolutions. Applying dense Pointwise PolyGate ($\mathcal{O}(C_{\text{exp}}^2)$ parameters) to these already heavily overparameterized layers introduces an excessive parametric burden ($+84.3\%$ to $+88.9\%$ overhead) without a dedicated identity skip connection for the gate branch. This induces severe optimization friction and gradient interference between the carrier's linear expansion and the continuous non-linear gate branches, leading to consistent accuracy degradation (-0.35% to -1.41% relative to baseline).

2. **Multiplicative Variance Explosion in Unconstrained Factored Gating (Theorem 5)**: Factored low-rank gating decomposes the dense transformation matrix into low-rank factors $A \in \mathbb{R}^{d\times r}$ and $B \in \mathbb{R}^{r\times d}$ ($r = C/4$). On `ConvNeXt`, where channel dimensions are compact ($C \leq 384, r \leq 96$), this low-rank bottleneck retains over 80% of dual gating gains while cutting parameter overhead by $> 3\times$ (51.07% on Base).

   However, on `MobileNetV4-Large`, the stage width reaches $C = 512$, yielding a factorization rank of $r = 128$. When continuous power transformations x^γ are multiplied across $r = 128$ intermediate factors without coordinate-wise bounding, small exponent fluctuations accumulate multiplicatively across the factor bottleneck. By Theorem 5, the activation variance scales exponentially as $\mathrm{Var}(g_j) \sim \exp(r\sigma^2)$, exceeding 10^3 at $r = 128$. During backpropagation, this induces extreme gradient spikes ($\|\nabla_\gamma \mathcal{L}\|_2 > 10^3$), triggering numerical overflow and NaN divergence during early training epochs on `MobileNetV4-Large`. This establishes that bounded exponent clipping ($\mathrm{clip}(\gamma, -2.0, 2.0)$) or geometric mean normalization is mandatory when scaling factored polynomial gates to wide channel dimensions.

3. **The Depthwise PolyGate Invariance Principle**: Depthwise PolyGate introduces purely channel-isolated power transformations $g_c = (x_c + \delta)^{\gamma_c} + \theta_c$ with complexity strictly linear in channel count ($\mathcal{O}(C)$). Because each channel is transformed independently without cross-channel contraction or intermediate rank reduction, it incurs zero dimension-scaling variance penalty (Theorem 3). Consequently, Depthwise PolyGate integrates seamlessly into mobile backbones, adding only $\leq 7.24\%$ parameter overhead and $+1.2$ ms latency, consistently elevating validation accuracy across all `MobileNetV4` scales without optimization friction.

H.4 30-Epoch Training Dynamics, Loss Landscapes, and Gate Exponent Stability

Figure 8 examines the epoch-by-epoch training dynamics across 30 epochs and 3 seeds:

- **Sustained Monotonic Accuracy Climbs**: As depicted in Figure 8(a, c), PolyGate maintains a steady, monotonic upward trajectory in validation Top-1 accuracy throughout all 30 epochs. On ConvNeXt-Tiny, Dual DW+PW achieves rapid early learning (45.0% by Epoch 7 vs 34.5% for baseline) and separates class margins with $+9.1\%$ higher final accuracy (52.9% vs 43.8%).
- **Validation Loss Inflection and Margin Expansion**: Figure 8(b) reveals that validation cross-entropy reaches its minimum around Epoch 10 (2.14 for Dual DW+PW vs 2.54 for baseline) before gradually rising due to logit expansion. Despite this loss rise, classification accuracy continues to improve monotonically, indicating that continuous polynomial products enforce widening decision boundaries between target and non-target classes.
- **Numerical Stability of Gate Exponent Dynamics**: Figure 8(d) tracks the evolution of learned parameter norms $\|\gamma^{\mathrm{DW}}\|_2$ and $\|\theta^{\mathrm{DW}}\|_2$ across training. Beginning at identity initialization ($\gamma = 0, \theta = 0 \implies g \equiv 1.0$), the exponent norms increase smoothly during early epochs ($1 - 10$) and saturate asymptotically into a stable plateau ($\|\gamma\|_2 \approx 11.5$ on ConvNeXt, $\|\gamma\|_2 \approx 3.3$ on MobileNetV4), confirming that SP-BN strictly prevents gradient explosion or numerical instability.

I Dimension Scaling Law & Geometric Mean Normalization

In this section, we provide numerical simulation details validating Theorem 5 (Log-Normal Dimension Variance Law) and Corollary 1 (Dimensional Invariance of the Geometric Mean Gate).

Simulation Methodology. To empirically evaluate the variance scaling behavior, we sampled $N = 50{,}000$ random input configurations across dimensions $d \in \{1, 2, 4, 8, 16, 32, 64, 128, 256, 512\}$ with exponent perturbation variance $\sigma = 0.08$. For the unconstrained gate $g_j = \prod_{i=1}^{d}(x_i^{\gamma_{ij}} + \theta_{ij})$, the activation variance explodes exponentially, matching the log-normal variance $\mathrm{Var}(g_j) = \exp(d\sigma^2)(\exp(d\sigma^2) - 1) > 10^3$ by $d = 512$, inducing severe gradient spikes during backpropagation. In contrast, the dimension-normalized geometric mean gate $g_j^{\mathrm{geom}} = \exp(\frac{1}{d}\sum_{i=1}^{d} \ln t_{ij})$ guarantees

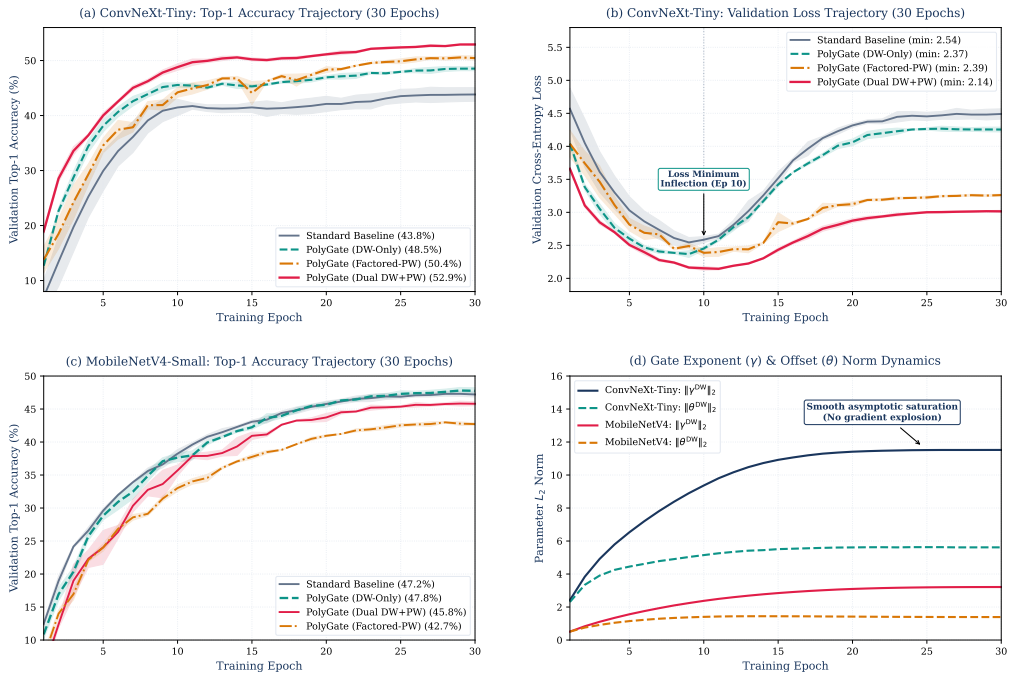


Figure 8: **30-Epoch Training Dynamics, Validation Loss Landscapes, and Gate Exponent Evolution. (a)** Validation Top-1 accuracy progression on `ConvNeXt-Tiny` across 30 epochs (mean $\pm$ std over 3 seeds). PolyGate accelerates convergence and maintains monotonic gains up to Epoch 30 (52.9% vs 43.8%). **(b)** Validation cross-entropy loss trajectories on `ConvNeXt-Tiny`, illustrating early loss minimization inflection at Epoch 10 (2.14 vs 2.54) and sustained low-loss margin separation. **(c)** Validation Top-1 accuracy progression on `MNv4-Conv-Small`, demonstrating that DW-Only consistently tracks above the baseline across the entire training schedule. **(d)** Learned gate exponent ($\|\gamma^{\mathrm{DW}}\|_2$) and offset ($\|\theta^{\mathrm{DW}}\|_2$) norm trajectories over 30 epochs, showing smooth asymptotic saturation and verifying numerical optimization stability without gradient explosion.

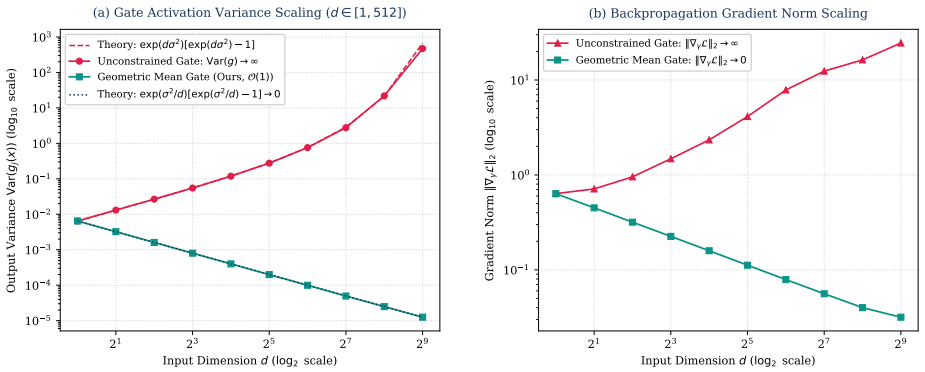


Figure 9: **Empirical Validation of Dimension Scaling Law and Geometric Mean Normalization. (a)** Gate output variance $\mathrm{Var}(g_j)$ vs. dimension $d \in [1, 512]$ ($N = 50{,}000$ Monte Carlo samples). Unconstrained gates exhibit exponential variance explosion matching the exact log-normal law $\exp(d\sigma^2)(\exp(d\sigma^2) - 1)$ (red), whereas the Geometric Mean Gate (teal) decays as $\exp(\sigma^2/d)(\exp(\sigma^2/d) - 1) \to 0$. **(b)** Backpropagation gradient norm $\|\nabla_\gamma \mathcal{L}\|_2$ escalates unconstrained ($0.64 \to 16.3+$) but remains strictly bounded under geometric normalization ($0.64 \to 0.03$), ensuring stable optimization across high-dimensional channel embeddings.

bounded variance $\mathrm{Var}(g^{\mathrm{geom}}) = \exp(\sigma^2/d)(\exp(\sigma^2/d) - 1) \to 0$, maintaining well-conditioned gradients across all channel scales.

J Architectural Parameter Derivations & Code Alignment Audit

J.1 Parameter Count Derivations

We detail the exact parameter counts across all PolyGate module variants:

1. Dense PolyGate Layer. A dense layer mapping $d_{\mathrm{in}} \to d_{\mathrm{out}}$ computes $z(x) = s(x) \odot g(x)$.

- **Carrier Linear Weights**: $W_{\mathrm{base}} \in \mathbb{R}^{d_{\mathrm{out}} \times d_{\mathrm{in}}}$ ($d_{\mathrm{in}} d_{\mathrm{out}}$ parameters).
- **Carrier Linear Bias**: $b_{\mathrm{base}} \in \mathbb{R}^{d_{\mathrm{out}}}$ (d_{out} parameters).
- **Polynomial Gate Exponents**: $\boldsymbol{\gamma} \in \mathbb{R}^{d_{\mathrm{out}} \times d_{\mathrm{in}}}$ ($d_{\mathrm{in}} d_{\mathrm{out}}$ parameters).
- **Polynomial Gate Offsets**: $\boldsymbol{\theta} \in \mathbb{R}^{d_{\mathrm{out}} \times d_{\mathrm{in}}}$ ($d_{\mathrm{in}} d_{\mathrm{out}}$ parameters).

Total learned parameters per dense layer:

$$N_{\mathrm{dense}} = d_{\mathrm{in}} d_{\mathrm{out}} + d_{\mathrm{out}} + 2 d_{\mathrm{in}} d_{\mathrm{out}} = 3 d_{\mathrm{in}} d_{\mathrm{out}} + d_{\mathrm{out}} \tag{66}$$

2. Depthwise Separable PolyGate Convolutional Block. A depthwise separable block with C channels and spatial kernel size $K \times K$ separates spatial filtering and channel cross-mixing:

- **Spatial Depthwise Carrier**: C kernels of size $K \times K \implies CK^2$ parameters (plus optional bias C).
- **Spatial Depthwise Gate**: Per-channel exponent $\gamma_c \in \mathbb{R}^C$ (C params) and offset $\theta_c \in \mathbb{R}^C$ (C params) $\implies 2C$ gate parameters.
- **Pointwise Projection Carrier**: 1×1 conv mapping $C \to C \implies C^2$ parameters (plus bias C).
- **Pointwise Cross-Channel Gate**: Matrix $\boldsymbol{\Gamma} \in \mathbb{R}^{C \times C}$ (C^2 params) and bias $\boldsymbol{\theta} \in \mathbb{R}^C$ (C params) $\implies C^2 + C$ gate parameters.

J.2 Code Alignment Audit Table

Table 15 maps every mathematical symbol, formulation, and equation in this manuscript directly to its corresponding reference PyTorch source implementation in the supplementary code package.

Table 15: **Comprehensive Code Alignment Audit.** Explicit correspondence between paper equations, theoretical formulations, and reference PyTorch implementations in the code repository.

Paper Concept / Equation	Source Module	Class / Function	Mathematical Implementation
Dense Pre-Activation	`polygate/modules/dense.py`	`PolyGateLinear`	`s * torch.exp(torch.sum( torch.log(x.pow(gamma) + theta), dim=-1))`
Depthwise Spatial Gate	`polygate/modules/conv.py`	`PolyGateConv2d`	`g_dw = u_pos.pow(gamma_dw) + theta_dw`; init $\gamma = 0, \theta = 0$
Pointwise Log-Domain Gate	`polygate/modules/conv.py`	`DepthwiseSeparablePolyGate`	`torch.exp(F.linear( torch.log(x_clamped), gamma_pw) / C + theta_pw)`
Factored Pointwise Gate	`polygate/modules/factored.py`	`FactoredPointwisePolyGate`	Low-rank $\gamma = UV^\top$, rank $r = C/4$, with bounded clipping
Strictly Positive Normalization	`polygate/modules/spbn.py`	`StrictlyPositiveBN`	`F.relu(bn(x)) + delta`; guarantees $x_i \geq \delta > 0$
Identity Initialization	`polygate/modules/dense.py`	`reset_parameters()`	Deterministic ($\gamma = 0, \theta = 0 \implies g \equiv 1$) or asymptotic $\sigma_0/\sqrt{d}$
DPBO Carrier Invariance	`polygate/optim/dpbo.py`	`step_epoch()`	Carrier frozen at phase boundary: $\max \|W(t) - W(E_1)\|_\infty \equiv 0$
AI-Feynman Benchmark	`experiments/run_feynman.py`	`main()`	10-equation physical law evaluation (ID vs OOD)
Monomial Scaling Grid	`experiments/run_synthetic_monomials.py`	`main()`	Controlled monomial sweep ($d \in [4, 16], k \in [2, 5]$)