RHONORM: LEARNING CHANNEL-ENERGY WEIGHTS IN TRANSFORMER NORMALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

RMSNorm uses fixed, uniform weights to aggregate channel energies into one normalization scale per token. We investigate the value of learning these weights inside modern Transformers. Building on weighted divisive normalization, RhoNorm uses an exponentially parameterized positive diagonal metric with identity initialization and one learnable vector per existing normalization site. Our analysis connects its task gradients and exact finite-spike radius response through the same channel-energy shares. Controlled comparisons support relative energy weighting beyond scalar and vector output gains; the diagonal form approaches dense-metric mean quality at much lower parameter cost. Across four decoder sizes, RhoNorm improves clean prediction and reduces spike-induced extra loss in middle and late layers. In a 2.82B-parameter model, $100\times$ spikes in approximately 1% of block-input coordinates at 75% depth add 6.3 nats of token-mean cross-entropy with RhoNorm, compared with 15.0 for RMSNorm. Predictive gains extend to image classification and multimodal fine-tuning. On nanochat's time-to-GPT-2 target, a 1.38B-parameter model exceeds the GPT-2 reference in mean full CORE after 88 minutes of training on eight H100 GPUs, compared with 99 minutes for the fastest published Autoresearch entry.

1 INTRODUCTION

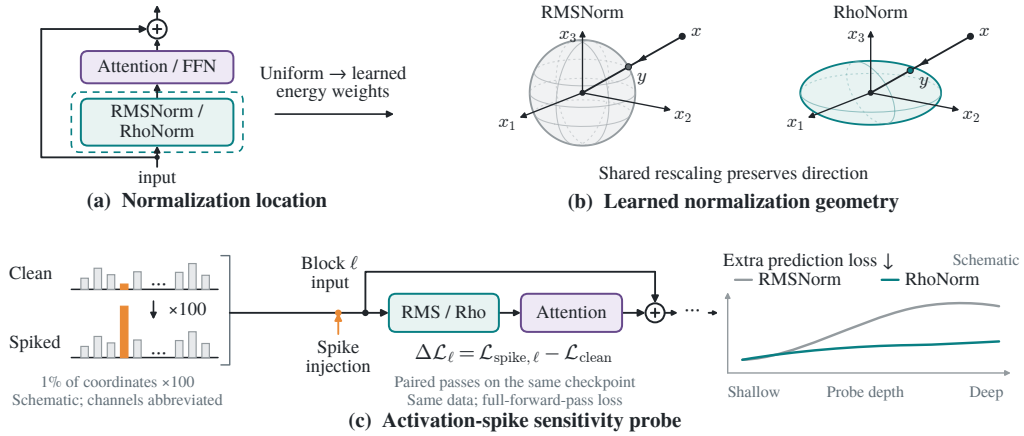
Transformers learn the features entering each normalizer and can learn an output gain, yet RMSNorm (Zhang & Sennrich, 2019) assigns a fixed, equal weight to every channel's squared activation. These weights determine how energy in one channel changes the scale applied to the entire token. Learning an output gain changes the result of the division; learning the energy weights changes the denominator itself. We ask whether this latter degree of freedom improves Transformer prediction beyond learning the surrounding features or output gains.

We introduce RHONORM, which learns this rule jointly with the network under its ordinary predictive objective. Starting from $x^\top Px$, $P = I$ recovers RMS normalization and $P \succ 0$ allows a learned positive-definite metric. RhoNorm chooses $P = \mathrm{diag}(e^w)$, using one vector per site in place of a dense matrix. Initialization at $w = 0$ recovers uniform weighting; exponential coordinates express multiplicative changes in positive weights. Two tokens with the same RMS but different channel-energy profiles can then receive different denominators. The learned vector is shared across tokens; the core rescales each input along its ray according to its feature composition (Figure 1(a,b)).

Weighted divisive normalization has established precedents in image statistics, density modeling and perceptual noise studies (Lyu & Simoncelli, 2008; Ballé et al., 2016). Our focus is a shared log-parameterized metric learned through ordinary Transformer prediction. Sparse multiplicative spikes test the resulting rule when energy becomes concentrated in a few coordinates. Our analysis identifies each channel's weighted energy share as the common quantity governing task gradients and finite-spike radius changes (Section 2.2). Because the shared rescaling also affects unperturbed channels, we assess its predictive consequences through the complete trained network.

Controlled comparisons test where the additional parameters are useful. Relative energy weighting improves clean quality beyond scalar and vector output gains (Table 2); a diagonal metric approaches the dense metric's observed mean quality with 0.006% added parameters (Table 3). Clean validation bits per byte (bpb) improve across four decoder sizes from 0.29B to 2.82B parameters. Activation probes show lower spike-induced extra loss, with the largest reductions in middle and late layers

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.



(a) **Normalization location** (b) **Learned normalization geometry** (c) **Activation-spike sensitivity probe**

Figure 1: **Learning and probing the shared metric.** (a) Normalization in a pre-norm sublayer. (b) The core rescales inputs along their rays to a learned metric level set, before any output gain. (c) A sparse block-input spike reaches both branches; clean/spiked losses are paired within each checkpoint after the full forward pass. Bars and response curves are schematic; measured results appear in Figure 2.

and variable shallow effects (Figure 2). These results support the normalization design through both ordinary prediction and a structured stress test.

DeiT classification (Touvron et al., 2021) and Qwen2.5-VL-7B fine-tuning extend the predictive evaluation beyond decoder pretraining. We also assess the design under a practical training-time budget. On nanochat's time-to-GPT-2 target (Karpathy, 2025), a 1.38B-parameter model exceeds the GPT-2 reference in mean full CORE after 88 minutes of training on eight H100 GPUs, compared with 99 minutes for the fastest Autoresearch entry in the September 22, 2026 snapshot (Karpathy, 2026). Section 3.7 specifies this comparison of complete configurations.

2 Learning a Shared Normalization Metric

2.1 Learning channel-energy weights

Let $x \in \mathbb{R}^n \setminus \{0\}$ be a token activation. RMS normalization uses the radius $\mathrm{RMS}(x) = \sqrt{x^\top I x/n}$. Replacing the identity by a symmetric positive-definite metric gives

$$D_P(x) = \sqrt{\frac{x^\top P x}{n}}, \qquad f_P(x) = \frac{x}{D_P(x)}, \qquad P \succ 0. \tag{1}$$

The positive denominator is shared across coordinates. A dense P adds cross-channel terms at quadratic cost; RhoNorm uses the diagonal form

$$P_w = \mathrm{diag}(e^{w_1}, \ldots, e^{w_n}), \qquad D_w(x) = \sqrt{\frac{1}{n}\sum_j e^{w_j} x_j^2}, \qquad f_w(x) = \frac{x}{D_w(x)}. \tag{2}$$

Each site stores one zero-initialized vector and computes a weighted-square reduction. We analyze the core without numerical stabilization or output gain. With a matched stabilizer and gain, $w = 0$ recovers the corresponding RMS mapping (Appendix A). This weighted L_2 form (Lyu & Simoncelli, 2008; Ballé et al., 2016) corresponds to tied pooling rows in generalized divisive normalization (GDN) (Appendix A.3).

Let $a_j(x) = x_j^2/\|x\|_2^2$ and $f_0(x) = x/\mathrm{RMS}(x)$. The relative radius is

$$\rho_w(x) = \frac{D_w(x)}{\mathrm{RMS}(x)} = \sqrt{\sum_j a_j(x) e^{w_j}}, \qquad f_w(x) = \rho_w(x)^{-1} f_0(x). \tag{3}$$

Thus a single parameter vector changes the shared rescaling according to each token's energy distribution, on both ordinary and perturbed activations. For $c > 0$ and $y = f_w(x)$,

$$f_w(cx) = f_w(x), \qquad f_w(-x) = -f_w(x), \qquad y^\top P_w y = n. \tag{4}$$

The output lies on the input ray and a learned ellipsoid. At Q/K sites, this changes each attention logit according to both tokens' energy profiles while retaining the query and key directions (Appendix A.2).

2.2 Channel influence and finite-spike response

Write $S = \sum_j e^{w_j} x_j^2$ and $q_j = e^{w_j} x_j^2 / S$ for the weighted energy shares, with $q_j \geq 0$ and $\sum_j q_j = 1$. A channel's share measures its influence on the common radius: $\partial \log D_w / \partial \log |x_j| = q_j$ for $x_j \neq 0$. Sparse spikes test this dependence beyond the local regime. For fixed w, let $x^{(k,M)}$ multiply coordinates in a set M by $k \geq 1$ and leave the others unchanged. Their combined clean energy share is $q_M(x) = \sum_{j \in M} q_j$. The spiked energy is $S[1 + (k^2 - 1)q_M(x)]$, giving the exact response

$$\frac{D_w(x^{(k,M)})}{D_w(x)} = \sqrt{1 + (k^2 - 1)q_M(x)}. \tag{5}$$

For $k = 100$, the factor is $\sqrt{1 + 9999 q_M(x)}$. Relative to the clean output, affected coordinates are multiplied by k and every coordinate is divided by this factor, exactly even for large spikes.

RMSNorm obeys the same relation with uniform metric weights. RhoNorm learns their relative contributions to the response. A larger share in the affected coordinates produces stronger denominator growth and also reduces the scale of unaffected coordinates. Its effect on network loss depends on the jointly trained metric and surrounding layers. The input Jacobian, derived in Appendix A.4, describes the corresponding local response; Appendix A.5 expands the finite-spike argument.

2.3 Predictive learning in exponential coordinates

The log-weights parameterize $\lambda_j(w_j) = e^{w_j} > 0$ for finite w_j in exact arithmetic. At zero, both λ_j and its derivative equal one, and

$$\frac{\lambda_j(w_j + \Delta w_j)}{\lambda_j(w_j)} = e^{\Delta w_j}, \qquad \frac{\partial \log D_w(x)}{\partial w_j} = \frac{1}{2} q_j. \tag{6}$$

For upstream loss gradient $g = \nabla_y \mathcal{L}$, the task gradient is $\partial \mathcal{L} / \partial w_j = -(g^\top y) q_j / 2$. The common scalar sets this token's gradient sign, while q_j allocates it across channels. Different tokens contribute different energy profiles, allowing the predictive objective to learn relative weights. These same weights determine the finite-spike radius response. Alternative positive maps with the same effective weights give the same forward operator and input Jacobian; they change the optimization coordinates. We compare these maps for predictive quality in Table 10.

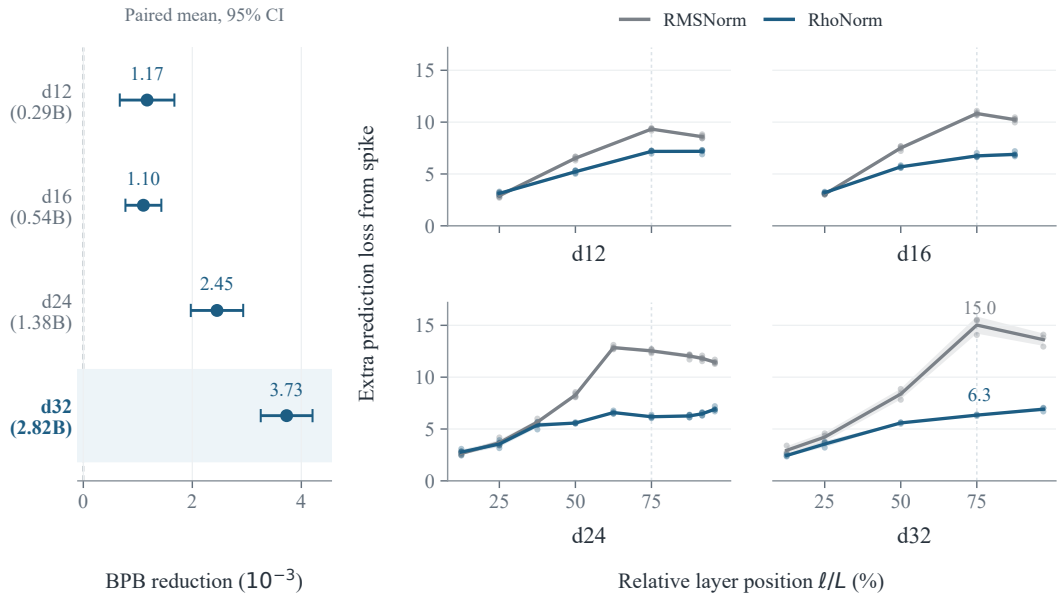
2.4 Separating relative weights from output scale

Write $w = \alpha \mathbf{1} + v$, where $\alpha = n^{-1} \sum_j w_j$ and $\mathbf{1}^\top v = 0$. Then

$$D_w(x) = e^{\alpha/2} D_v(x), \qquad f_w(x) = e^{-\alpha/2} f_v(x). \tag{7}$$

The mean log-weight controls an overall scale; v controls how the radius varies with channel-energy composition. The energy shares q_j are unchanged by α. The scalar case $v = 0$ is in the ScaleNorm family (Nguyen & Salazar, 2019), while $\alpha = 0$ retains relative weights with $\det P_w = 1$. These restrictions define the controls in Table 2.

A static output gain Γ can be folded into the next linear map: $W\Gamma f_0(x) = (W\Gamma) f_0(x)$. At fixed input coordinates, a non-isotropic metric cannot be absorbed into a following nonzero projection in this way (Proposition 1, Appendix A.1). Free adjacent linear maps permit $f_P(x) = P^{-1/2} f_0(P^{1/2} x)$. The proposition concerns fixed input coordinates; it does not establish greater network-wide expressiveness or a convergence advantage.



(a) Model scale

(b) Spike sensitivity

Figure 2: **Clean quality and the extra loss caused by activation spikes.** (a) Clean validation bpb reduction (RMSNorm minus RhoNorm, 10^{-3}): mean and 95% paired Student interval. (b) Token-mean cross-entropy increase in nats, spiked minus clean at the same checkpoint; lower is better. Gray denotes RMSNorm and blue RhoNorm. Lines and bands show mean ± 1 sample SD, and pale points show individual runs. Each block is probed separately. Vertical guides mark 75% depth; d32 annotations give rounded loss increments. Recipes vary across sizes, and the d24 quality and probe cohorts differ. Counts and protocols are in Appendices B.2 and E.1.

3 Experiments

3.1 Evaluation design

Decoder training uses the ordinary language-model objective without spike augmentation or a robustness loss. We evaluate fixed checkpoints' clean quality and extra loss under internal spikes, retaining cohort-specific controls without pooling baselines across training batches.

Language-model cohorts. We use nanochat (Karpathy, 2025); dL denotes L decoder blocks. Probes cover d12/d16/d24/d32. The principal d24 quality setting uses 1.384B parameters, sequence length 2048, and 5,568 updates over 5.838B tokens, with a global batch of 1,048,576 tokens on eight Hopper GPUs. Both arms use tensorwise 8-bit floating-point (FP8) arithmetic for eligible linear layers and FlashAttention-3 (FA3). Clean quality is minimum recorded validation bits per byte (bpb), summarized by means, sample SDs and paired Student intervals. Appendix B gives architecture and cohort details.

The baseline uses affine-free RMS normalization at all 98 d24 sites. *Block* replaces only the 48 normalizers before attention and feed-forward networks (FFNs), retaining RMS normalization elsewhere. *Full* replaces all 98 existing sites, including the 48 query/key (Q/K), embedding-output and final normalizers. Both scopes preserve the baseline's normalization locations. Q/K parameters are shared across heads over 128 channels. Full is the principal quality setting; Block supplies output-gain and structured-metric controls.

3.2 Predictive quality and spike sensitivity

Figure 2 evaluates the normalization design under ordinary activations and channel-energy spikes; each outcome retains its own cohort.

Table 1: **Language-model quality across normalization choices.** All entries use d24, 5,568 updates, and five seeds; bpb is mean ± sample SD. Block/Full replace 48/98 existing RMS normalization sites. The labels use r for the low-rank setting and h for SeeDNorm groups. Blue marks RhoNorm; added parameters are relative to the 1.384B backbone. Each Δ_{RMS} uses the matching Block or Full RMSNorm reference. SeeDNorm scores come from separate batches without matched deltas. Protocols and additional variants appear in Appendices B and C.

Method	Scope	Added params (%)	Val. bpb ↓	Δ_{RMS} (10^{-3})
RMSNorm (Block ref.)	–	0.0000	0.717218 ± 0.000258	–
RMSNorm (Full ref.)	–	0.0000	0.717272 ± 0.000196	–
Scalar gain	Block	< 0.0001	0.716587 ± 0.000240	↓ 0.631
Vector gain	Block	0.0053	0.716457 ± 0.000138	↓ 0.761
Rotated metric	Block	0.0080	0.716104 ± 0.000176	↓ 1.114
Low-rank exp., $r = 8$	Block	0.0479	0.715898 ± 0.000144	↓ 1.320
SeeDNorm, $h = 1$	Block	0.0160	0.722115 ± 0.001461	–
SeeDNorm, $h = 8$	Block	0.0160	0.719256 ± 0.000668	–
SeeDNorm, $h = 1$	Full	0.0180	0.730998 ± 0.005481	–
RhoNorm	Block	0.0053	$\mathbf{0.715966} \pm 0.000191$	↓ 1.252
RhoNorm	Full	0.0060	$\mathbf{0.714820} \pm 0.000260$	↓ 2.453

Ordinary prediction. Within-size paired bpb reductions are 0.001170, 0.001102, 0.002453 and 0.003730 for d12, d16, d24 and d32 (Figure 2(a)). Every seed pair improves, and each size's paired 95% interval excludes zero. The d32 means are 0.654701 for RMSNorm and 0.650970 for RhoNorm, a 0.57% relative reduction; its mean reduction has interval $[0.003255, 0.004205]$. Both d32 arms use FP8, 9,600 updates and 20.133B tokens in the same CUDA-Graph codebase. Between-size differences also reflect implementation, precision and training budget (Appendix B.2).

Sparse-energy stress test. We measure $\Delta\mathcal{L} = \mathcal{L}_{\mathrm{spike}} - \mathcal{L}_{\mathrm{clean}}$ as token-mean cross-entropy in nats on the same batch and targets; lower values indicate less predictive damage. Figure 1(c) sketches the protocol; Figure 2(b) reports measurements. Because injection precedes residual branching, the spike enters both the normalized branch and the bypass. The probe measures end-to-end checkpoint tolerance, rather than attenuation by one normalizer.

Each element of the $[B, T, C]$ block-input tensor is selected with probability 0.01 and multiplied by 100. Layer positions are tested independently with a fixed mask seed and four sequences of length 2048. We compare five trained pairs at each of d12/d16/d24 and three at d32; these training pairs determine the displayed SDs. The batch and mask are fixed conditions, not additional independent replications.

At d32 block 24 (75% depth), RMSNorm incurs 15.025 ± 0.827 extra loss and RhoNorm 6.338 ± 0.078, less than half as much. RhoNorm's mean is lower at all five sampled d32 positions; every pair improves from half depth onward. Appendix E.1 gives complete paired statistics.

Across four sizes, reductions are largest in middle and late layers. Both shallow d32 improvement intervals include zero, and one pair worsens at 25% depth; the benefit is not monotonic across layers.

3.3 Controls for learning the denominator

These controls examine clean-quality gains; they do not attribute spike tolerance to individual design choices.

Normalization choices. The principal d24 Full-scope comparison reduces bpb from 0.717272 to 0.714820, improving all five pairs (Table 1). The mean difference is -0.002453, with 95% interval $[-0.002933, -0.001972]$: a 0.34% relative reduction with 0.006% additional parameters.

The Block cohort compares metrics with output gains at 48 native sites. Low-rank metrics give similar means to RhoNorm at larger parameter cost (Appendix C). SeeDNorm uses our nanochat implementations and recipes, with its separate-batch scope stated in the table.

Table 2: **Denominator learning improves beyond output gains.** Each contrast uses five same-seed differences within its cohort. Units are 10^{-3} bpb; arrows show the first-minus-second change. Downward changes favor the first method. Gray intervals cross zero.

Trained comparison	Scope	Δ bpb (10^{-3})	95% paired CI
RhoNorm − vector gain	Block	$\downarrow$ 0.490	$[-0.800, -0.181]$
Zero-mean metric − scalar gain	Full	$\downarrow$ 1.846	$[-2.303, -1.388]$
Zero-mean metric − RhoNorm	Full	$\downarrow$ 0.105	$[-0.753, +0.543]$
RhoNorm + gain − RhoNorm	Full	$\uparrow$ 0.211	$[-0.247, +0.668]$

Table 3: **Metric complexity and allocation of additional capacity.** Four d24 arms, 5,568 updates, 5.838B tokens, and the same eight-GPU setup. Minimum logged validation bpb is mean $\pm$ sample SD over seeds 111/222/333. FFN depth counts linear layers; hidden width is 6144 throughout. Deep FFN uses RMSNorm and a fully dense middle projection, with more parameters than Dense Metric. Δ_{RMS} is the mean change from this cohort's RMSNorm. Dense parameterization and protocols are in Appendix C.4.

Model / metric	Params (B)	Added (%)	FFN depth	Val. bpb $\downarrow$	Δ_{RMS} (10^{-3})
RMSNorm	1.384	0.0000	2	0.717246 ± 0.000260	–
RhoNorm (diagonal)	1.384	0.0060	2	$\mathbf{0.714991} \pm 0.000086$	$\downarrow$ 2.255
Dense metric	1.503	8.5795	2	0.714946 ± 0.000516	$\downarrow$ 2.300
Deep FFN	2.290	65.4545	3	0.756747 ± 0.042510	$\uparrow$ 39.501

Matched output-gain controls. Table 2 asks whether the extra parameters improve quality through the denominator rather than ordinary output scaling. RhoNorm improves over an equal-parameter vector gain. The zero-mean metric uses $w = u - \mathrm{mean}(u)$, removing the independent global scale while retaining relative channel weights; it also improves over scalar-only normalization. These results support learning relative channel-energy weights in Equation (7). Removing the global-scale component or adding a vector output gain yields intervals spanning zero, so those contrasts do not establish equivalence.

3.4 Metric complexity and training cost

Table 3 compares diagonal and dense metrics, RMSNorm, and an RMSNorm backbone with a three-layer FFN at d24. All use 5,568 updates, 5.838B tokens and three common seeds, fixing data exposure while varying the added capacity.

RhoNorm reduces mean bpb from 0.717246 to 0.714991 with 0.0060% added parameters. Dense reaches 0.714946 with 8.58% added parameters. Its mean advantage of 0.000045 is unresolved by the paired RhoNorm-minus-dense interval $[-0.001120, +0.001210]$. The observed quality and parameter cost support the diagonal choice at this budget; they do not establish statistical equivalence. Deep FFN adds 65.45% parameters and yields 0.756747 ± 0.042510 bpb, worsening all three seeds. That result concerns the full-width three-layer FFN under the common schedule, with unequal capacity and substantial seed variation.

Parameterization. Exp has the lowest observed mean bpb in the Full-scope positive-weight study (Appendix Table 10). Its differences from softplus and clamped-linear weights survive the reported Holm correction; the square variant is not statistically distinguished. Initial derivatives and near-floor behavior vary across maps, so the study supports a practical parameterization without isolating a single optimization mechanism.

Runtime. A systems cohort compares compiled RMSNorm with RhoNorm using Q/K fusion and CUDA Graph replay, with the optimizer outside capture. At 5,568 updates, RhoNorm improves bpb by 0.002601 but increases training-step time from 99.56 to 104.62 minutes (5.1%; Appendix Table 17).

Table 4: **ImageNet-1k with size-specific settings.** Validation top-1 (%), mean ± sample SD over three seeds. LayerNorm/RhoNorm use best validation checkpoints from 300-epoch runs. Tiny uses native and Q/K sites; Small uses timm and EMA evaluation; Base uses native sites with stronger regularization. Each column has its corresponding LayerNorm reference. Derf uses our standard DeiT recipe; RhoNorm uses the size-specific settings above. Δ_{LN} is the change in percentage points, computed before rounding.

Method	DeiT-Tiny	DeiT-Small	DeiT-Base
LayerNorm	72.20 ± 0.11	80.31 ± 0.14	81.59 ± 0.20
Derf (DeiT standard)	65.17 ± 0.24	76.16 ± 0.13	80.87 ± 0.15
RhoNorm	$\mathbf{74.71} \pm 0.18$	$\mathbf{81.24} \pm 0.07$	$\mathbf{82.03} \pm 0.11$
Δ_{LN} (pp)	↑ 2.51	↑ 0.93	↑ 0.43

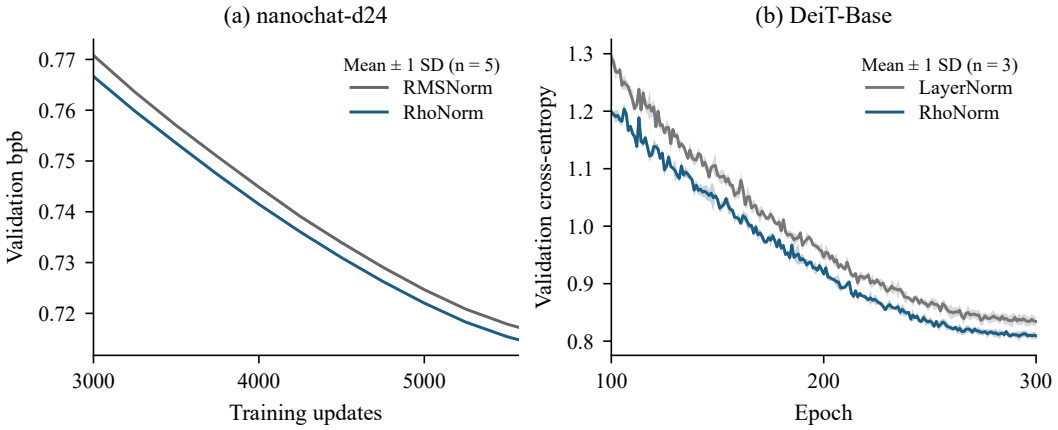


Figure 3: **Validation trajectories of the tested configurations.** Lines show across-seed means; shading denotes ±1 sample SD, without smoothing. (a) Five paired seeds from the principal Full-scope language-model experiment. (b) DeiT-Base: standard LayerNorm and stronger-regularization RhoNorm recipes, with three successful runs each (Appendix D). Base seed sets are not fully paired (LayerNorm: 42/43/45; RhoNorm: 42/43/44).

3.5 IMAGENET CLASSIFICATION

DeiT-Tiny/Small/Base (Touvron et al., 2021) are trained on ImageNet-1k (Deng et al., 2009) at 224×224 for 300 epochs without distillation, using AdamW (Loshchilov & Hutter, 2019) at learning rate 10^{-3}. For each size, we report the completed three-seed RhoNorm configuration with the highest mean best validation accuracy. Tiny/Base use the DeiT runner; Small uses timm v3 and exponential moving average (EMA) evaluation with its corresponding LayerNorm reference (Appendix D.1).

The selected RhoNorm configurations reach 74.71%, 81.24% and 82.03% top-1, above their LayerNorm references (Table 4). Derf (Chen et al., 2026) is evaluated with our standard DeiT recipe, which may differ from its original training protocol. The table characterizes these implementations and settings rather than an identical-recipe ranking of all methods.

For Base, the selected recipe increases drop path from 0.1 to 0.2 and mixup from 0.8 to 1.0. Under the original recipe, RhoNorm has lower training loss but higher validation loss than LayerNorm; stronger regularization improves its validation trajectory (Figure 3). The selected result combines the normalizer and adjusted recipe. Appendix D retains within-RhoNorm contrasts and site-dependent failures.

3.6 MULTIMODAL TASK ADAPTATION

The pretrained-model study replaces every `Qwen2RMSNorm` in Qwen2.5-VL-7B (Bai et al., 2025) with DiagExpNorm (RhoNorm). Each replacement retains the pretrained gain γ and initializes

$w = 0$; with the original stabilizer, the initial mapping is unchanged. Both parameters and the rest of the model are updated during full-parameter supervised fine-tuning (SFT).

The PARA (Yang et al., 2022) subset contains 17,500 examples sampled uniformly across mean opinion score (MOS) intervals; the VQAv2 (Goyal et al., 2017) subset contains 23,000 examples. Each uses an 80:20 train/test split and one run per configuration: bfloat16 (BF16), three epochs, learning rate 10^{-5}, cosine decay and warmup ratio 0.03. Protocol details are in Appendix B.5.

Table 5: **Qwen2.5-VL-7B full-parameter SFT.** One run per configuration on the sampled test sets. The replacement retains the pretrained gain and initializes its log-metric to zero. Δ is the descriptive change from baseline; pp denotes percentage points. VQAv2 uses majority-label exact match.

Dataset	Metric	Baseline	RhoNorm	Δ
PARA	PLCC	0.954	**0.957**	$\uparrow$ 0.003
VQAv2 subset	Accuracy (%)	76.83	**78.00**	$\uparrow$ 1.17 pp

PARA's Pearson linear correlation coefficient (PLCC) with human MOS rises from 0.954 to 0.957. On sampled VQAv2, exact match to each question's most frequent annotated answer rises from 76.83% to 78.00% (Table 5). These single-run task scores support adaptation from an equivalent initial mapping; they are neither official full-benchmark VQA scores nor additional activation-spike measurements.

3.7 Time to GPT-2-level CORE

To assess practical training budgets, we use nanochat's time-to-GPT-2 target. GPT-2 is a capability reference, and the benchmark permits architectural changes (Karpathy, 2026); the trained model is the nanochat decoder. Full CORE averages 22 chance-normalized downstream task scores, using all evaluation examples rather than a per-task subsample.

RhoNorm reaches mean full CORE 0.2582 ± 0.0080 after 88.0 training minutes, above the GPT-2 reference of 0.256525. The 4,700-update horizon is the shortest tested horizon whose mean crosses it (Table 6). Each horizon is independently trained over five seeds; Appendix E.3 gives all horizons and evaluation costs.

The approximate 95% mean interval at 4,700 updates is $[0.2483, 0.2681]$, which includes the reference. The result is an observed mean crossing, not a certified minimum time or an all-seed pass. The September 22, 2026 leaderboard snapshot reports Autoresearch times of 108 and 99 minutes. Compared with its fastest entry, 88 minutes is 11 minutes, or 11.1%, less reported training time on matched hardware. This compares complete configurations, rather than isolating a normalizer-only speedup (Appendix E.3).

4 Related Work

Scale control and Transformer design. LayerNorm and RMSNorm (Ba et al., 2016; Zhang & Sennrich, 2019) control activation scale; ScaleNorm (Nguyen & Salazar, 2019) learns an overall radius. nGPT (Loshchilov et al., 2025) constrains representations and weights and changes residual updates on a hypersphere. QK normalization, NormFormer and Peri-LN (Henry et al., 2020; Shleifer et al., 2021; Kim et al., 2025) change normalization placement or extent. Our focus is the energy-weighting rule within each denominator and its predictive behavior on ordinary and spiked activations.

Learned divisive normalization and noise. Lyu & Simoncelli (2008) derive weighted normalization from Gaussian scale mixtures and evaluate statistical dependence and perceptual distortion under additive representation noise. Ballé et al. (2016) learn a linear transform, pooling weights, offsets and exponents for Gaussianization, density modeling and image denoising. RhoNorm's shared core is their weighted L_2 form in the scale-invariant zero-offset limit: a GDN expression with fixed exponents and tied pooling rows. We study its log-parameterization under a predictive objective and measure the loss caused by sparse multiplicative spikes inside Transformers. Appendix A.3 gives the algebraic mapping, its invertibility boundary, and the distinction between dense quadratic

Table 6: **Full-CORE quality and training budgets.** All runs use the same eight-H100-GPU configuration; published records use the September 22, 2026 snapshot. RhoNorm reports mean ± SD over five seeds at independently trained horizons. $\Delta_{\text{GPT-2}}$ is mean CORE minus 0.256525. The blue row is the first tested horizon with a mean above this reference. The reported times count training; evaluation costs appear in the appendix.

Model / updates	Train (min)	Full CORE ↑	$\Delta_{\text{GPT-2}}$
Published nanochat records (8 H100 GPUs)			
GPT-2 reference	–	0.256525	–
nanochat + ClimbMix	121.2	0.2571	↑ 0.0006
Autoresearch round 1	108.0	0.2690	↑ 0.0125
Autoresearch round 2	99.0	0.2626	↑ 0.0061
RhoNorm (8 H100 GPUs): independent horizons, five seeds			
RhoNorm, 4,300	81.3	0.2538 ± 0.0072	↓ 0.0027
RhoNorm, 4,700	**88.0**	0.2582 ± 0.0080	↑ 0.0017
RhoNorm, 5,568	105.5	0.2639 ± 0.0060	↑ 0.0074

metrics and pooling squared coordinates. Ellipsoidal projection also appears in attention-dynamics analyses (Rodríguez Abella et al., 2025).

Output gains and coordinate-wise alternatives. AdaNorm (Xu et al., 2019) and SeeDNorm (Cai et al., 2026) use input-dependent output scaling. DyT and Derf (Zhu et al., 2025; Chen et al., 2026) replace normalization statistics with learned pointwise tanh and erf functions. NormalNorm and IBNorm (Eftekhari & Papyan, 2025; Zou et al., 2025) reshape coordinate distributions through power/noise or compression operations. RhoNorm retains shared radial rescaling; Proposition 1 distinguishes metric learning from output gains at fixed input coordinates. Normalization–optimizer interactions (Abouzeid, 2026) further motivate interpreting quality rankings within their training recipes.

5 Discussion and Conclusion

The results support channel-energy aggregation as a useful place to add trainable parameters in Transformers. RhoNorm improves clean prediction with one vector per site, while checkpoint probes show lower sensitivity to middle- and late-layer spikes. Gain and capacity controls motivate relative weighting and the diagonal implementation for predictive quality.

The depth sweep conditions on one batch, mask and spike strength in a single decoder family; it does not measure training reliability or general input-corruption robustness. Vision uses selected recipes, multimodal SFT uses sampled test sets and single runs, and reported CORE times compare complete configurations.

Reproducibility statement

The supplement includes source records, analysis scripts, reference implementations, and instructions to regenerate tables, figures and the PDF. Appendix B documents protocols and endpoints.

AI use statement

Generative AI assisted with literature review, conceptual framing, mathematical exposition and proofs, experimental-design feedback, implementation, result interpretation, manuscript drafting and editing, and analysis and visualization code. Empirical values are traceable to the records in Appendix B. The authors are responsible for all claims, code, and conclusions.

REFERENCES

Abdelrahman Abouzeid. Does Your Optimizer Care How You Normalize? Normalization-Optimizer Coupling in LLM Training. *arXiv preprint arXiv:2604.01563*, 2026. URL https://arxiv.org/abs/2604.01563.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. URL https://arxiv.org/abs/1607.06450.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025. URL https://arxiv.org/abs/2502.13923.

Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. Density Modeling of Images using a Generalized Normalization Transformation. In *International Conference on Learning Representations*, 2016. URL https://arxiv.org/abs/1511.06281.

Wenrui Cai, Defa Zhu, Qingjie Liu, and Qiyang Min. SeeDNorm: Self-Rescaled Dynamic Normalization. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2510.22777.

Mingzhi Chen, Taiming Lu, Jiachen Zhu, Mingjie Sun, and Zhuang Liu. Stronger Normalization-Free Transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27418–27428, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Chen_Stronger_Normalization-Free_Transformers_CVPR_2026_paper.html.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848. URL https://doi.org/10.1109/CVPR.2009.5206848.

Daniel Eftekhari and Vardan Papyan. On the Importance of Gaussianizing Representations. In *International Conference on Machine Learning*, volume 267, pp. 15018–15037. PMLR, 2025. URL https://proceedings.mlr.press/v267/eftekhari25a.html.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6904–6913, 2017. URL https://openaccess.thecvf.com/content_cvpr_2017/html/Goyal_Making_the_v_CVPR_2017_paper.html.

Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. Query-Key Normalization for Transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4246–4253, 2020. doi: 10.18653/v1/2020.findings-emnlp.379. URL https://aclanthology.org/2020.findings-emnlp.379/.

Andrej Karpathy. nanochat. https://github.com/karpathy/nanochat, 2025.

Andrej Karpathy. nanochat Time-to-GPT-2 Leaderboard. https://github.com/karpathy/nanochat/blob/92d63d4e8bb4df75c3b71618f31ddde2378b2bcd/README.md, 2026. Snapshot accessed September 22, 2026; protocol documented in dev/LEADERBOARD.md.

Jeonghoon Kim, Byeongchan Lee, Cheonbok Park, Yeontaek Oh, Beomjun Kim, Taehwan Yoo, Seongjin Shin, Dongyoon Han, Jinwoo Shin, and Kang Min Yoo. Peri-LN: Revisiting Normalization Layer in the Transformer Architecture. In *International Conference on Machine Learning*, volume 267, pp. 30400–30436. PMLR, 2025. URL https://proceedings.mlr.press/v267/kim25u.html.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Ilya Loshchilov, Cheng-Ping Hsieh, Simeng Sun, and Boris Ginsburg. nGPT: Normalized Transformer with Representation Learning on the Hypersphere. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.01131.

Siwei Lyu and Eero P. Simoncelli. Nonlinear Image Representation Using Divisive Normalization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008. doi: 10.1109/CVPR.2008.4587821. URL https://doi.org/10.1109/CVPR.2008.4587821.

Toan Q. Nguyen and Julian Salazar. Transformers without Tears: Improving the Normalization of Self-Attention. In *Proceedings of the 16th International Conference on Spoken Language Translation*, 2019. URL https://aclanthology.org/2019.iwslt-1.17/.

Álvaro Rodríguez Abella, João Pedro Silvestre, and Paulo Tabuada. Consensus is all you get: The role of attention in transformers. In *International Conference on Machine Learning*, volume 267, pp. 174–184. PMLR, 2025. URL https://proceedings.mlr.press/v267/abella25a.html.

Sam Shleifer, Jason Weston, and Myle Ott. NormFormer: Improved Transformer Pretraining with Extra Normalization. *arXiv preprint arXiv:2110.09456*, 2021. URL https://arxiv.org/abs/2110.09456.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, volume 139, pp. 10347–10357. PMLR, 2021. URL https://proceedings.mlr.press/v139/touvron21a.html.

Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and Junyang Lin. Understanding and Improving Layer Normalization. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/2f4fe03d77724a7217006e5d16728874-Abstract.html.

Yuzhe Yang, Liwu Xu, Leida Li, Nan Qie, Yaqian Li, Peng Zhang, and Yandong Guo. Personalized image aesthetics assessment with rich attributes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19861–19869, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Yang_Personalized_Image_Aesthetics_Assessment_With_Rich_Attributes_CVPR_2022_paper.html.

Biao Zhang and Rico Sennrich. Root Mean Square Layer Normalization. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/1e8a19426224ca89e83cef47f1e7f53b-Abstract.html.

Jiachen Zhu, Xinlei Chen, Kaiming He, Yann LeCun, and Zhuang Liu. Transformers without Normalization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14901–14911, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Zhu_Transformers_without_Normalization_CVPR_2025_paper.html.

Xiandong Zou, Jia Li, Xiaotong Yuan, and Pan Zhou. IBNorm: Information-Bottleneck Inspired Normalization for Representation Learning. *arXiv preprint arXiv:2510.25262*, 2025. URL https://arxiv.org/abs/2510.25262.

A Perturbation Analysis and Numerical Details

A.1 Metric placement in fixed coordinates

Proposition 1 (Metric placement in fixed coordinates) *Let $n \geq 2$ and $P \succ 0$, using the ideal zero-floor operators. (i) The map $x \mapsto Af_0(x)$, $A \in \mathbb{R}^{n\times n}$, preserves every positive input ray if and only if $A = aI$ for some $a > 0$. (ii) For any nonzero $W \in \mathbb{R}^{m\times n}$, there exists a fixed $\widetilde{W}$ satisfying $Wf_P(x) = \widetilde{W}f_0(x)$ for every $x \neq 0$ if and only if $P = pI$ for some $p > 0$.*

Let e_i be the ith coordinate basis vector. For Proposition 1(i), $Af_0(e_i) = \sqrt{n}Ae_i$ must be a positive multiple of e_i, so A is diagonal with entries $a_i > 0$. For any $i \neq j$, applying the condition to $e_i + e_j$ forces $a_i = a_j$. Thus $A = aI$, and the converse is immediate.

For (ii), assume $Wf_P(x) = \widetilde{W}f_0(x)$ for every $x \neq 0$. Set $c(x) = \mathrm{RMS}(x)/D_P(x) > 0$, so $\widetilde{W}x = c(x)Wx$. Choose a nonzero row $w^\top$ of W, and let $b^\top$ be the corresponding row of $\widetilde{W}$. On the hyperplane $w^\top x = 0$, the equality gives $b^\top x = 0$. Hence $b = aw$ for some scalar a. Outside that hyperplane, $a = c(x) > 0$. Continuity gives $c(x) = a$ on the complete nonzero domain. Squaring $D_P(x) = \mathrm{RMS}(x)/a$ yields $x^\top Px = a^{-2}x^\top x$ for every x, so symmetry implies $P = a^{-2}I$. Conversely, for $P = pI$, choose $\widetilde{W} = W/\sqrt{p}$. The assumption $W \neq 0$ is necessary: the zero projection absorbs every metric by taking $\widetilde{W} = 0$.

This result holds for the complete nonzero input domain in fixed coordinates. It does not rule out equivalence on a restricted activation distribution, and the zero projection is excluded. It is also compatible with the two-sided identity $f_P = P^{-1/2}f_0 \circ P^{1/2}$: if the preceding and following linear maps are both freely changed, the coordinate transforms can be absorbed there. A residual network shares intermediate activations across branches, so that identity alone is not a proof of a global network reparameterization or of an optimization advantage.

Pointwise alternatives make a different design choice. A continuous coordinate-wise map $F(x)_i = \phi_i(x_i)$ satisfying $F(cx) = F(x)$ for every $c > 0$ must have each ϕ_i constant on each half-line; continuity at zero makes those constants equal. A constant vector cannot preserve every input ray with nonzero output. Thus nontrivial continuous pointwise maps such as DyT and Derf give up the joint constraints studied here. This algebraic distinction does not predict their empirical quality.

A.2 The radial response at Q/K sites

For a metric P, write $\rho_P(x) = D_P(x)/\mathrm{RMS}(x)$, extending Equation (3) beyond diagonal metrics. For query q and key k with metrics P_q and P_k, suppressing the common attention scale gives

$$f_{P_q}(q)^\top f_{P_k}(k) = \frac{f_0(q)^\top f_0(k)}{\rho_{P_q}(q)\rho_{P_k}(k)}. \tag{8}$$

The normalized vectors retain their Euclidean directions, while the positive factor rescales their dot product according to both energy profiles. A pair of static vector gains instead changes the numerator to $q^\top \Gamma_q^\top \Gamma_k k$ before the isotropic denominators. This illustrates the operator-level distinction in the design constraints. The key-dependent radius can still change the ordering of attention logits; ray preservation does not mean unchanged attention weights, and it does not imply that adding Q/K normalization improves every architecture.

A.3 Weighted normalization, shared metrics, and GDN

Original constructions. Combining Equations (8)–(11) of Lyu & Simoncelli (2008), with their intermediate activation renamed v, gives

$$v = (I - A)x, \qquad r_i = \frac{v_i}{\sqrt{b + \sum_j C_{ij}v_j^2}}.$$

Here A performs linear decorrelation, C pools squared coefficients, and b is an additive constant. The neighborhood and spatial-homogeneity assumptions give A, C sparse block-Toeplitz structure.

Section 3 of Ballé et al. (2016) attributes the shared weighted L_2 form $y_i = x_i/(\beta^2 + \sum_j \gamma_j x_j^2)^{1/2}$ to this work, after decorrelation. This compact form should be distinguished from the full local image transform above.

The generalized transform in Equation (5) of Ballé et al. (2016) is

$$z = Hx, \qquad y_i = \frac{z_i}{(\beta_i + \sum_j \gamma_{ij}|z_j|^{\alpha_{ij}})^{\varepsilon_i}}.$$

It learns the preceding transform, pooling weights, offsets and exponents for Gaussianization and density modeling. The paper enforces $\beta_i > 0$ and seeks an invertible mapping.

Exact algebraic correspondence. The shared weighted L_2 expression equals ideal RhoNorm for $\beta = 0$, $\gamma_j = e^{w_j}/n$ and $x \neq 0$. In the generalized expression, the corresponding settings are

$$H = I, \qquad \alpha_{ij} = 2, \qquad \varepsilon_i = \tfrac{1}{2}, \qquad \beta_i = 0, \qquad \gamma_{ij} = e^{w_j}/n \quad \text{for every } i.$$

This is a homogeneous boundary of the algebraic family: positive input rescalings have identical outputs, precluding an invertible density transformation. For additive stabilization $\delta > 0$, setting $\beta_i = \delta$ (or $\beta^2 = \delta$ in the shared expression) matches the stabilized radius. A clamped energy floor instead has distinct near-zero behavior (Appendix A.6). Exponential coordinates cover strictly positive weights; they change optimization coordinates without creating a new positive-weight forward family.

In the square-pooling GDN form with fixed coordinates ($H = I$), the pooling matrix Γ multiplies the vector of squared activations. Tying its rows, $\Gamma = \mathbf{1}\gamma^\top$, produces a shared denominator. Thus a diagonal quadratic metric corresponds to a row-tied GDN pool, not a diagonal pool. An unrestricted shared quadratic metric instead includes $x_j x_k$ terms through the off-diagonal entries of P. Entrywise nonnegativity of Γ and positive definiteness of P are different constraints. This comparison fixes the input coordinates. A learned transform H can mix them before pooling, so the absence of cross terms in the table is not a restriction on every composition involving general GDN.

Table 7: Operator distinctions in fixed input coordinates. GDN here denotes square pooling with $H = I$; its general linear-transform form has additional degrees of freedom. All denominators are assumed positive.

Operator	Shared denominator	$x_j x_k$ terms, $j \neq k$	Preserves input ray
Weighted L_2 / RhoNorm	Yes	No	Yes
Shared dense metric $P \succ 0$	Yes	Allowed	Yes
Square-pooling GDN, $H = I$	Generally no	No	Generally no

Different perturbations and endpoints. Section 6 of Lyu & Simoncelli (2008) explicitly tests resilience to noise: white Gaussian noise is added to representation coefficients, and inverse transforms reconstruct the images. Noise amplitudes are adjusted to match pixel-domain MSE/PSNR before comparing perceptual distortion and SSIM. The image-denoising experiment of Ballé et al. (2016) instead fits the noisy-image density for empirical Bayes estimation. Our decoder probes multiply sparse block-input coordinates by 100 and continue the forward prediction, reporting $\mathcal{L}_{\text{spike}} - \mathcal{L}_{\text{clean}}$ for each fixed checkpoint. The checkpoints use ordinary predictive training; the probes do not supply a training loss. These distinctions specify the scope of the present evidence without claiming that earlier divisive normalization work omitted noise or perturbation studies.

A.4 Gradients, tangent spaces, and scale response

Let $S = x^\top P_w x$ and define the metric energy shares $q_j = e^{w_j} x_j^2/S$. Let $J_x = \partial y/\partial x$ be the input Jacobian. For an upstream gradient $g = \nabla_y \mathcal{L}$,

$$J_x = \frac{1}{D_w}\left(I - \frac{x(P_w x)^\top}{S}\right), \qquad \nabla_x \mathcal{L} = \frac{1}{D_w}\left(g - \frac{P_w x}{S} x^\top g\right), \tag{9}$$

$$\frac{\partial y}{\partial w} = -\frac{1}{2} y q^\top, \qquad \nabla_w \mathcal{L} = -\frac{x^\top g}{2 D_w} q. \tag{10}$$

The input Jacobian removes radial perturbations, $J_x x = 0$, and maps surviving perturbations into the ellipsoid's tangent space, $(P_w x)^\top J_x = 0$. The metric gradient distributes a common upstream alignment signal according to channel-energy shares. For a fixed upstream gradient,

$$J_x(cx) = c^{-1} J_x(x), \qquad \nabla_w \mathcal{L}(cx) = \nabla_w \mathcal{L}(x), \qquad c > 0. \tag{11}$$

RhoNorm retains the inverse-scale response of the RMS activation gradient, while its metric gradient is invariant to positive rescaling of the token. A static post-RMS gain is also scale invariant, but its parameter gradient is coordinate-wise, $g \odot f_0(x)$. The distinction is the radial coupling through the denominator. These are local properties of the normalizer; the derivation follows below.

Derivation. Let $P = \mathrm{diag}(e^w)$, $S = x^\top P x$, and $D = \sqrt{S/n}$ for $x \neq 0$. Direct differentiation gives

$$\frac{\partial D}{\partial x_j} = \frac{e^{w_j} x_j}{nD}, \qquad \frac{\partial D}{\partial w_j} = \frac{e^{w_j} x_j^2}{2nD}. \tag{12}$$

Using $y_i = x_i / D$,

$$\frac{\partial y_i}{\partial x_j} = \frac{\delta_{ij}}{D} - \frac{x_i e^{w_j} x_j}{nD^3}, \qquad \frac{\partial y_i}{\partial w_j} = -\frac{x_i e^{w_j} x_j^2}{2nD^3}. \tag{13}$$

Here δ_{ij} is the Kronecker delta. Substituting $S = nD^2$ and $q_j = e^{w_j} x_j^2 / S$ gives Equations (9)–(10). Contracting with the upstream gradient gives the stated backpropagation rules.

The matrix $I - x(Px)^\top / S$ is an oblique projection of rank $n - 1$. Its null space is the span of x and its range is orthogonal to Px, the normal vector of the output ellipsoid. Its transpose maps the loss gradient into the Euclidean plane orthogonal to x, so $x^\top \nabla_x \mathcal{L} = 0$. These statements concern the forward perturbation and backward gradient spaces, respectively. Because $S(cx) = c^2 S(x)$, $D(cx) = cD(x)$, and $q(cx) = q(x)$, the gradient scale laws follow for $c > 0$ and a fixed upstream gradient.

With $u = P^{1/2} x$, we have $P^{1/2} y = u / (\|u\|_2 / \sqrt{n})$, proving the ellipsoid constraint. Moreover, $f_w(x) = P^{-1/2} f_0(P^{1/2} x)$. In a free linear–normalization–linear chain, the coordinate transforms can be absorbed into the adjacent matrices; the metric then changes parameterization rather than necessarily enlarging the function class. For comparison, $z = \Gamma x / \mathrm{RMS}(x)$ with invertible diagonal Γ satisfies $z^\top \Gamma^{-2} z = n$, but generally does not preserve the input ray. The log-radius has derivatives

$$\nabla_w \log D = \tfrac{1}{2} q, \qquad \nabla_w^2 \log D = \tfrac{1}{2} [\mathrm{diag}(q) - q q^\top] \succeq 0, \tag{14}$$

since $\partial q_i / \partial w_j = q_i (\delta_{ij} - q_j)$. Its Hessian is half a categorical covariance matrix. Convexity of this local quantity does not imply convexity of the network training objective.

For a fixed nonzero token, $\partial \log \|y\|_2 / \partial w_j = -q_j / 2$. Increasing a coordinate weight raises the denominator and reduces output magnitude on that token. Task gradients need not increase that weight, however, and arbitrarily negative common shifts of w produce arbitrarily large outputs in the zero-floor model. The coupling is not a global boundedness claim.

A.5 Exact response to finite activation spikes

Fix w and a nonzero input x in the ideal operator. Let $x^{(k,M)}$ multiply coordinates in a set M by $k \geq 1$, leaving all others unchanged. The clean input's weighted energy share in M is

$$q_M(x) = \frac{\sum_{j \in M} e^{w_j} x_j^2}{\sum_j e^{w_j} x_j^2}. \tag{15}$$

The weighted energy changes from S to $S[1 + (k^2 - 1) q_M(x)]$, so Equation (5) follows by taking the square root of this energy ratio. For the probe's $k = 100$, this ratio is $\sqrt{1 + 9999 q_M(x)}$. The expression is exact for finite spikes; it does not invoke the small-perturbation approximation based on the input Jacobian. Relative to the clean output, coordinates in M are multiplied by k and all coordinates are divided by the common ratio in Equation (5).

RMSNorm satisfies the same identity with $w = 0$. Learning the metric changes the energy shares and hence this response, but the identity alone does not guarantee smaller output deviations or task-loss increases. It describes the normalizer at fixed parameters, while the checkpoint probes measure the complete trained network. The expressions below specify how additive and clamped stabilization modify the finite-spike response.

A.6 NUMERICAL-FLOOR CONVENTIONS

The implementations use an additive radius $D_\varepsilon = \sqrt{S/n + \varepsilon}$ or a clamped energy floor $D_\tau = \sqrt{\max(S/n, \tau)}$. For $A = 1 + (k^2 - 1)q_M(x)$, their exact finite-spike radius ratios are

$$\frac{D_\varepsilon(x^{(k,M)})}{D_\varepsilon(x)} = \sqrt{\frac{SA + n\varepsilon}{S + n\varepsilon}}, \qquad \frac{D_\tau(x^{(k,M)})}{D_\tau(x)} = \sqrt{\frac{\max(SA/n, \tau)}{\max(S/n, \tau)}}. \tag{16}$$

The clamped form recovers Equation (5) when both energies exceed the floor. The additive form approaches it when $n\varepsilon$ is negligible relative to S; it is not exactly identical at a nonzero ε. The exact additive derivatives are

$$\nabla_x \mathcal{L} = \frac{g}{D_\varepsilon} - \frac{Px(x^\top g)}{nD_\varepsilon^3}, \qquad \frac{\partial \mathcal{L}}{\partial w_j} = -\frac{(x^\top g)e^{w_j} x_j^2}{2nD_\varepsilon^3}. \tag{17}$$

In particular, $x^\top \nabla_x \mathcal{L} = \varepsilon(x^\top g)/D_\varepsilon^3$ and $y^\top P y = nS/(S + n\varepsilon)$. For the clamped energy floor, the ideal derivatives apply above the threshold. Below it, $J_x = I/\sqrt{\tau}$ and $\nabla_w \mathcal{L} = 0$; the threshold is nondifferentiable. Clamping the radius at δ corresponds to $\tau = \delta^2$. All exact invariance claims in the main text use the zero-floor definition.

The following reference illustrates the operator with 32-bit floating-point (FP32) accumulation and an example energy floor of 10^{-12}. The archived standalone `DiagExpNorm` implementation instead defaults to an energy floor of 10^{-6} and computes in the input dtype. Reproduction requires the floor and dtype of the actual run, and matching conventions when checking identity initialization against a baseline.

```
class RhoNorm(nn.Module):
    def __init__(self, dim, energy_floor=1e-12):
        super().__init__()
        self.w = nn.Parameter(torch.zeros(dim))
        self.energy_floor = energy_floor

    def forward(self, x):
        z = x.float() if x.dtype in (
            torch.float16, torch.bfloat16) else x
        energy = (self.w.to(z.dtype).exp() * z.square())
        energy = energy.mean(-1, keepdim=True)
        y = z * energy.clamp_min(self.energy_floor).rsqrt()
        return y.to(x.dtype)
```

This reference is not a statement that every production kernel uses the same accumulation path. The fused implementations evaluate the same weighted quadratic energy without materializing a matrix.

A.7 SCALAR AND ZERO-MEAN CONTROLS

The scalar control sets $w = \alpha\mathbf{1}$; the zero-mean control uses $w = u - \mathrm{mean}(u)$. The latter has $n - 1$ effective degrees of freedom when stored as a length-n vector.

B PROTOCOLS, ENDPOINTS, AND PROVENANCE

B.1 LANGUAGE-MODEL COHORTS

The principal language-model experiment uses eight NVIDIA H800 GPUs; the separate systems batch uses eight H100 GPUs. Both have device batch 16, sequence length 2048, gradient accumulation four, and 5,568 updates, for exactly 5,838,471,168 training tokens. FP8 applies to eligible linear layers, not to all model arithmetic. The backbone contains 1,384,122,122 total parameters, including value embeddings. The approximately 730M scaling parameters used by the pipeline's token-ratio rule differ from the total parameter count. Native parameter-group routing assigns RhoNorm's log-weight vectors to AdamW and Transformer projection matrices to Muon.

The principal comparison uses the complete `fa3i_base_5568` / `fa3i_diagexp_full_5568` families. These are archived run identifiers; both arms use the FlashAttention-3 attention implementation described in Section 3.1. Another baseline family, `fa3_base`, has mean 0.717203; it is not mixed with the selected baseline mean 0.717272. The Block cohort uses the explicit seed-level `main_raw.csv` baseline of 0.717218. The `d4combo` and `d13b` experiments each retain their own controls. Cross-cohort absolute differences are not treated as isolated effects of normalization. The local SeeDNorm comparison spans batches and does not reproduce all of its original model-specific training settings. These results do not establish an optimizer-independent ranking of normalization methods. Full and Block RhoNorm are also not a single factorial experiment; their difference is not assigned to individual Q/K, embedding, or final sites.

Table 8: Seed-level bpb, grouped by experiment. The principal comparison, systems comparison, and independent metric-plus-gain study each retain their own RMSNorm reference.

Method	111	222	333	444	555
Principal comparison					
RMSNorm	0.717179	0.717192	0.717437	0.717041	0.717512
RhoNorm	0.714854	0.714910	0.715075	0.714879	0.714380
Systems comparison					
RMSNorm	0.716695	0.716542	0.716586	0.716884	0.716543
RhoNorm	0.713801	0.714122	0.714160	0.713989	0.714171
Metric-plus-gain study					
RMSNorm	0.717061	0.717134	0.717544	0.716843	0.717535
RhoNorm	0.714923	0.714963	0.715088	0.714762	0.714255
RhoNorm + gain	0.714944	0.715114	0.715105	0.714766	0.715116

B.2 Model-scale protocols and seed results

The d12/d16 Full-scope cells use BF16, seeds 42/111/456/333/555, global batch 524,288 tokens, and 2,520/5,376 updates. Their token budgets are 1,321,205,760 and 2,818,572,288, respectively. The completed d32 study uses a reported 2.82B-parameter model, Full scope, and seeds 111/222/333. The d32 protocol uses FP8, 9,600 updates, and global batch 2,097,152, for 20,132,659,200 tokens per arm. Both d32 arms use the `nanochat_cuda_graph` codebase. This implementation and precision boundary separates d32 from the smaller BF16 cells; within-size paired contrasts are the relevant comparisons. The aggregate clean-quality contrasts are presented in the main-text Figure 2(a).

The d32 source is a summary of completed runs, archived with its source image. We compute means, sample SDs, and paired statistics from its six seed-level bpb values. The paired mean difference is -0.00373033, with sample SD 0.00019127 and 95% Student interval $[-0.00420546, -0.00325520]$ over the three training seeds.

Table 9: Completed d32 seed-level bpb. The paired row is RhoNorm minus RMSNorm. Values retain the precision of the supplied summary.

d32 / arm	111	222	333
RMSNorm	0.654703	0.654513	0.654886
RhoNorm	0.650903	0.650999	0.651009
Paired Δ	$\downarrow$ 0.003800	$\downarrow$ 0.003514	$\downarrow$ 0.003877

B.3 Statistical endpoints

The bpb endpoint is the minimum logged validation bpb over the training horizon, including the standard endpoint used for Table 3. It is validation selected, not a held-out test estimate. Means and SDs describe training seeds; paired intervals use the Student distribution on same-seed differences, without treating tokens as independent replications. Parameterization-family comparisons use the

multiple-comparison treatment described in Appendix C. The principal d24 clean-quality estimate uses the complete five-seed comparison; the ablation collection is exploratory. Its intervals are not adjusted across the entire history of configuration search, and the many rows in the main comparison are not independent replications of one treatment effect.

B.4 DeiT recipe and seed accounting

The DeiT recipe uses AdamW, weight decay 0.05, a nominal learning rate 5×10^{-4} scaled by global batch$/512$ to 10^{-3}, cosine decay, and five warmup epochs. Augmentations are RandAugment `rand-m9-mstd0.5-inc1`, mixup 0.8, CutMix 1.0, random erasing 0.25, color jitter 0.4, label smoothing 0.1, repeated augmentation, and bicubic interpolation. Model EMA is enabled with decay 0.99996. The stronger recipe changes drop path to 0.2 and mixup to 1.0. Both RhoNorm arms in the Base regularization study use the FP32-statistic implementation and seeds 42/43/44.

DeiT-runner accuracy results use each complete run's final logged `Max accuracy`, the best validation accuracy seen over 300 epochs. The selected Small result instead uses the timm runner's `Best metric` endpoint, as specified in Appendix D.1. The final training and validation losses instead use epoch 299. The distinction matters: the stronger-regularization configuration has mean best top-1 82.0267% and mean final top-1 81.9407%. Base LayerNorm seed 44 completes training with best accuracy 21.17%. The main reference uses successful seeds 42/43/45, with mean 81.5933%. This successful-run reference describes achievable accuracy; its seed set does not support a paired three-seed test against the selected RhoNorm runs.

B.5 Multimodal task protocol

PARA is a dataset for image aesthetics assessment with rich annotations (Yang et al., 2022). Our task uses image scores and their MOS targets; we do not claim evaluation of every personalized-attribute task supported by the dataset. The 17,500 sampled examples cover MOS intervals uniformly and are divided into 14,000 training and 3,500 test examples. We assess prediction–annotation agreement using PLCC, as reported in the main table.

The VQAv2 subset contains 23,000 examples, split into 18,400 training and 4,600 test examples. For each question, the most frequent annotated answer is selected as the target. Accuracy is the fraction of model outputs that exactly match this target. The experiment does not use the official soft-consensus VQA scoring rule or claim a result on the full official test split.

Both tasks use Qwen2.5-VL-7B, BF16 full-parameter SFT for three epochs, learning rate 10^{-5}, cosine decay, and warmup ratio 0.03, with one run per configuration. All `Qwen2RMSNorm` instances are replaced with `DiagExpNorm`, the implementation name for RhoNorm, while retaining their elementwise output gains.

Pretrained-gain initialization. Let γ_{pre} be the original layer's gain. The replacement sets $w = 0$ and clones $\gamma = \gamma_{\mathrm{pre}}$. With the same RMS stabilizer ε, its forward map is

$$y = \gamma \odot \frac{x}{\sqrt{n^{-1} \sum_j e^{w_j} x_j^2 + \varepsilon}}. \tag{18}$$

At initialization, $e^{w_j} = 1$, so this equals the pretrained RMSNorm mapping, including its output gain. Training updates w, γ, and all other model parameters without freezing. The direction-preservation properties of Section 2 describe the normalization core before the elementwise gain.

The matched initialization removes an initial mapping difference; one run per configuration does not estimate seed variation. The reported metrics are PLCC and majority-answer exact match; no SRCC values are available.

B.6 Result sources and protocol records

The principal language-model, d12/d16, LayerNorm/RhoNorm vision, and systems statistics are computed from training logs or the Block seed-level CSV. The d32 comparison uses the completed three-seed summary and protocol in Appendix B.2. Derf vision

statistics use the three-seed summary, retaining the two-decimal precision of its top-1 values. `paper_assets/statistics.json` records the exact source files and checksums. The directional decomposition uses the seed-level deltas in the supplied D10 summary because the complete anchor log families are not present. The d12/d16/d24 depth sweep is recomputed from 85 paired checkpoint JSON results, with provenance and statistics in `paper_assets/probe_depth_statistics.json`. The d32 extension adds 15 raw paired outcomes, recomputed in `paper_assets/d32_probe_depth_statistics.json`. Multimodal results are task-level scores with the protocol in Appendix B.5. The d24 four-arm summary contains all twelve seed-level bpb values and parameter counts; its statistics are recomputed in `paper_assets/d24_metric_four_arm_data.json`. The archive distinguishes log-derived estimates, seed-level summaries, and aggregate-only reports.

Reproduction limits of the task summaries. The multimodal summaries do not specify the optimizer, batch size, or sampled split manifests. The principal nanochat training-corpus snapshot is also absent from the archived records. These omissions limit exact reproduction of the reported runs.

C Additional Capacity and Parameterization Ablations

C.1 Global-scale removal and gain addition

In the supplied Full-scope decomposition, the paired changes from RMSNorm are -0.002356 for exp, -0.002461 for the zero-mean log-metric, and -0.000615 for scalar-only normalization. Zero-mean minus scalar is -0.001846, with 95% interval $[-0.002303, -0.001388]$. Zero-mean minus unconstrained exp is -0.000105, with interval $[-0.000753, +0.000543]$. The interval includes zero and permits nontrivial effects of either sign; it does not establish equivalence at a tightly specified tolerance.

The independent gain-addition cohort gives 0.717223 ± 0.000308 for RMSNorm, 0.714798 ± 0.000325 for RhoNorm, and 0.715009 ± 0.000154 for RhoNorm plus vector gain. The paired combo-minus-RhoNorm interval is $[-0.000247, +0.000668]$. We interpret this as no measured additive improvement, not as proof that all output gains are redundant.

C.2 Positive-weight family

The supplied parameterization study keeps one stored parameter per channel, identity initialization, the same fast path, and the same training cell. It compares e^w, $(1 + w/2)^2$, $\mathrm{softplus}(1 + w)/\mathrm{softplus}(1)$, and $\max(1 + w, 10^{-4})$. All have unbounded positive ranges above; the square form can attain zero. The softplus derivative at initialization is approximately 0.557, while the other three have derivative one.

Table 10: Supplied Full-scope positive-weight study, five seeds. Paired deltas and Holm-adjusted p values retain the source summary's reported values. Comparisons are against exp over a family of three; gray marks the comparison not distinguished at the 0.05 level. It is not an adjacent-pair ranking.

Weight map	Validation bpb	Δ_{exp}	Adjusted p
Exponential	**0.714846** $\pm$ 0.000205	–	–
Square	0.715058 $\pm$ 0.000320	$\uparrow$ 0.000212	0.054
Softplus	0.715150 $\pm$ 0.000282	$\uparrow$ 0.000304	0.0084
Clamped linear	0.715346 $\pm$ 0.000412	$\uparrow$ 0.000499	0.0230

The mean bpb differences from exp range from 0.000212 to 0.000499. These are comparisons of deployed positive-weight parameterizations: initial derivatives and near-floor behavior are not all matched. They do not isolate a single optimization mechanism or partition the total improvement into directional and parameterization contributions.

C.3 STRUCTURED METRICS AND RANK SWEEP

All metric variants retain the shared output $x/D_P(x)$. The rotated metric uses

$$P = Q^\top \operatorname{diag}(e^{2s})Q, \qquad Q = \operatorname{blockdiag}(R(\theta_1), \ldots, R(\theta_{n/2})),$$

where $R(\theta)$ is a two-dimensional Givens rotation on adjacent channels. Zero angles and $s = 0$ give $P = I$; the $3n/2$ angle and scale parameters are vectors routed to AdamW. The low-rank exponential family uses

$$P = UU^\top + \operatorname{diag}(e^w), \qquad U \in \mathbb{R}^{n\times r},$$

with $nr + n$ parameters per site, a Muon matrix group for U, and an AdamW vector group for w. The Block-scope rows replace the 48 existing pre-attention/FFN RMS normalizers and retain RMS normalization at all other sites.

Table 11 varies the low-rank dimension. These Block models improve bpb by approximately 0.00121–0.00132 relative to their baseline across ranks 8–128, without a clear rank trend. The minimal-rank member appears in Table 1; the complete sweep retains each rank's actual seed count.

Table 11: Low-rank metric sweep at Block scope. Validation bpb is mean ± sample SD; added parameters are relative to the backbone. The rank-32 and rank-128 rows have four and three seeds, respectively; the other ranks have five.

Rank	Seeds	Added params (%)	Val. bpb
8	5	0.0479	0.715898 ± 0.000144
16	5	0.0906	0.715906 ± 0.000139
32	4	0.1758	0.715952 ± 0.000237
64	5	0.3462	0.715911 ± 0.000152
128	3	0.6871	0.716044 ± 0.000128

C.4 FOUR-ARM METRIC PROTOCOLS

Fixed-budget d24 comparison. Table 3 uses d24, width 1536, FP8, and common seeds 111/222/333. All arms run 5,568 updates at 1,048,576 tokens per update, for exactly 5,838,471,168 training tokens, on the same eight-GPU setup. RMSNorm, RhoNorm, and Dense Metric retain the two-layer $1536 \to 6144 \to 1536$ FFN. Deep FFN uses $1536 \to 6144 \to 6144 \to 1536$ with a fully dense middle projection in all 24 blocks and ordinary RMSNorm. Its additional 24×6144^2 parameters give a total of 2,290,091,786, compared with 1,502,873,354 for Dense Metric. The comparison matches tokens and updates, with different parameter counts and FLOPs; no wall-time measurements accompany this summary.

Dense parameterization and initialization. The dense comparator stores an unconstrained matrix $L \in \mathbb{R}^{n\times n}$ at each normalization site and computes

$$P = LL^\top, \qquad D_L(x) = \sqrt{\max\left(\frac{x^\top LL^\top x}{n}, \tau\right)}, \qquad y = \frac{x}{D_L(x)}, \qquad \tau = 10^{-12}.$$

The implementation materializes P in the input dtype, then reduces $(x^\top P) \odot x$ over channels. It adds no ϵI term. After device allocation, L is initialized to I, recovering the RMS metric under the matched floor convention. This parameterization spans the positive-semidefinite closure of Equation (1); it is strictly positive definite when L is full rank. The numerical floor keeps the denominator positive even for a singular factor. It is not the separate triangular Cholesky mode.

RhoNorm and Dense replace all 98 existing RMS normalization sites (Full): 48 block, 48 Q/K, embedding-output and final sites. Dense therefore adds $50 \times 1536^2 + 48 \times 128^2 = 118{,}751{,}232$ matrix parameters. Its factor matrices use the native Muon groups; diagonal log-weights use AdamW, following the same parameter-type routing as the other comparisons.

RhoNorm improves over RMSNorm in all three pairs, with mean change -0.002255 and paired 95% interval $[-0.002689, -0.001821]$. Dense also improves all three, with mean change -0.002300 and

Table 12: Seed-level results for the d24 comparison in Table 3. Each row lists all three runs and the reported total parameter count.

Model / metric	Total parameters	Seed 111	Seed 222	Seed 333
RMSNorm	1,384,122,122	0.717061	0.717134	0.717544
RhoNorm (diagonal)	1,384,205,066	0.714923	0.714963	0.715088
Dense metric	1,502,873,354	0.714354	0.715298	0.715187
Deep FFN	2,290,091,786	0.805734	0.729549	0.734959

interval $[-0.003389, -0.001211]$. RhoNorm versus dense is not distinguished ($p = 0.883$, two-sided paired Student test); with three seeds, this is not an equivalence test. Deep FFN's paired interval versus RMSNorm is $[-0.066466, +0.145468]$, despite all three observed changes being worse. These intervals are descriptive, unadjusted pairwise intervals.

D Additional Vision Results

D.1 Selected settings and evaluation endpoints

Table 4 groups the shared positive diagonal denominator under a single RhoNorm row and selects one configuration per model size. Selection compares mean best validation accuracy among completed three-seed, 300-epoch, full-ImageNet runs. The table reports per-size tuned performance. Its sample SD measures training-seed variation at the selected recipe and does not account for configuration or checkpoint selection. Denominator-versus-gain attribution is evaluated in the language-model controls; the vision study assesses task performance and recipe dependence.

Tiny uses `rho32all_tiny`: FP32 normalization statistics at the 25 native sites and all 24 Q/K sites, with drop path 0.1 and mixup 0.8. Its LayerNorm reference is `baseline_tiny` in the same DeiT runner. Base uses `reg_dp2mix_base`: FP32 statistics at the 25 native sites, drop path 0.2, and mixup 1.0. Its standard-recipe LayerNorm reference uses drop path 0.1 and mixup 0.8; a matched stronger-recipe LayerNorm result is unavailable. Both sizes use best logged `Max accuracy` and crop ratio 0.875.

Small uses `deit_smallv3_allsites` and its corresponding `deit_smallv3_layernorm` reference in the timm v3 batch. Both use native AMP, compilation, AdamW with learning rate 10^{-3} and weight decay 0.05, and evaluation crop ratio 0.9. The endpoint is the final logged `Best metric`, which selects the best EMA validation checkpoint. Reconstruction from these endpoints gives $81.2413 \pm 0.0684\%$ for RhoNorm and $80.3100 \pm 0.1362\%$ for LayerNorm. The older supplied TSV used the last non-EMA evaluation and therefore has slightly different values; those last-epoch summaries are not used in Table 4.

Derf under the standard DeiT recipe. Derf is evaluated in our ImageNet benchmark environment using the standard DeiT training recipe and seeds 42/43/44. The top-1 results are $65.17 \pm 0.24\%$, $76.16 \pm 0.13\%$, and $80.87 \pm 0.15\%$ for Tiny, Small, and Base. This local recipe may differ from the published protocol of Chen et al. (2026); RhoNorm uses the per-size site, precision, and regularization choices above. The comparison concerns these tested configurations. The Derf summary does not specify checkpoint selection or EMA accounting; its source and precision are retained in the archive.

D.2 Normalization-site comparison

Table 14 compares the native-site and Q/K-site configurations within the DeiT runner, including its Small baseline of 79.96%, and remain separate from the selected timm Small comparison. At the same 25 native sites, the original RhoNorm-minus-LayerNorm mean differences are $+0.53$, $+0.14$, and -0.15 percentage points from Tiny to Base. This fixed-setting diagnostic motivates the regularization analysis below; it is not the selected-results row.

Adding Q/K sites changes the attention architecture, which is separate from changing the normalization statistic or regularization. The selected Base comparison changes the recipe as well as the

Table 13: Seed-level results for the selected ImageNet configurations. All selected RhoNorm runs use seeds 42/43/44; the Base LayerNorm reference uses the disclosed successful-run set 42/43/45. Small LayerNorm/RhoNorm use the best EMA validation checkpoint; Derf seed values retain their reported precision.

Size	Method	Seeds	Validation top-1 (%)
Tiny	LayerNorm	42, 43, 44	72.310, 72.180, 72.100
Tiny	Derf (DeiT standard)	42, 43, 44	65.06, 65.44, 65.01
Tiny	**RhoNorm**	42, 43, 44	74.920, 74.620, 74.590
Small	LayerNorm	42, 43, 44	80.458, 80.282, 80.190
Small	Derf (DeiT standard)	42, 43, 44	76.06, 76.11, 76.31
Small	**RhoNorm**	42, 43, 44	81.196, 81.208, 81.320
Base	LayerNorm	42, 43, 45	81.500, 81.820, 81.460
Base	Derf (DeiT standard)	42, 43, 44	80.81, 80.75, 81.04
Base	**RhoNorm**	42, 43, 44	81.960, 82.150, 81.970
Base	LayerNorm, failed run	44	21.170

Table 14: Normalization-site comparison under the standard DeiT recipe. Validation top-1 (%), mean $\pm$ sample SD of best validation scores from three successful 300-epoch runs. Base LayerNorm uses seeds 42/43/45; the other rows use 42/43/44.

Method / configuration	Sites	Recipe	Tiny	Small	Base
LayerNorm	25	Standard	72.20 ± 0.11	79.96 ± 0.22	81.59 ± 0.20
RhoNorm	25	Standard	$\mathbf{72.72} \pm 0.05$	$\mathbf{80.09} \pm 0.13$	$\mathbf{81.45} \pm 0.53$
RhoNorm + all Q/K	49	Standard	74.39 ± 0.10	79.74 ± 0.34	77.09 ± 0.28

normalizer; without a matched stronger-recipe LayerNorm result, that difference cannot be assigned to normalization alone.

Adding Q/K normalization gives a large Tiny gain and a Base degradation. For Base seed 42, parameter-free Q/K RMSNorm reaches 76.95%, close to the learned-Q/K run's best accuracy of 76.98%. Thus the degradation cannot be assigned specifically to learned Q/K weights. The stronger-regularization result in the main text concerns 25 native sites and is not a recovery experiment on the 49-site configuration.

D.3 Base fitting behavior and regularization

Under the same standard recipe, Base RhoNorm with FP32 statistics has lower final training loss than LayerNorm (2.544 versus 2.677), but higher final validation loss (0.866 versus 0.834). This supports an overfitting diagnosis. The training losses are online epoch averages under the training augmentations, so this observation concerns the optimized training objective. Increasing drop path from 0.1 to 0.2 and mixup from 0.8 to 1.0 reduces the validation loss to 0.809 and raises best top-1 from $81.43 \pm 0.55\%$ to $82.03 \pm 0.11\%$, with the other native-site settings held fixed.

The within-RhoNorm recipe contrast pairs seeds 42/43/44 and does not use the replacement-seed LayerNorm reference. All three best-accuracy changes are positive, but the sample is small: the 95% Student interval for the mean change includes zero (Table 15). The loss trajectory and endpoint differences support a regularization diagnosis; they are not a precise estimate of an architecture-independent improvement.

E Activation-Spike Evaluation and Systems Results

E.1 Activation-spike protocol and complete sweeps

The original depth sweep contains five paired training seeds at each size: 42/111/333/456/555 for d12/d16 and 111/222/333/444/555 for d24. The d12/d16 checkpoints use

Table 15: Base regularization diagnostics and within-RhoNorm contrasts. Standard uses drop path 0.1 and mixup 0.8; stronger uses 0.2 and 1.0. Both RhoNorm recipes use 25 native sites. Training and validation loss use the final epoch; accuracy is the best validation value. The lower panel pairs seeds 42/43/44, with gray intervals crossing zero.

Method	Recipe	Train loss	Val. loss	Best top-1 (%)
LayerNorm	Standard	2.677	0.834	81.59 ± 0.20
RhoNorm	Standard	2.544	0.866	81.43 ± 0.55
RhoNorm	Stronger	2.687	0.809	$\mathbf{82.03} \pm 0.11$

Within-RhoNorm: stronger minus standard, three paired seeds

Outcome	Mean change	95% paired interval
Best top-1 (pp)	$\uparrow 0.593$	$[-0.516, +1.703]$
Final validation loss	$\downarrow 0.056$	$[-0.136, +0.024]$

`conv{12,16}_{base,rhonorm}_r12` tags; d24 uses `d4combo_{base,rhonorm}_5568`. The supplied manifest gives 2,520, 5,376, and 5,568 training updates, respectively. All 85 result files contain both arms: four sampled layers each for d12/d16 and nine for d24. The d32 extension adds 15 paired outcomes: seeds 111/222/333 at layers 4, 8, 16, 24, and 31, from the completed FP8 checkpoints described in Appendix B.2. Its supplied protocol specifies the same input batch and mask procedure, using the CUDA-Graph codebase with graph replay disabled so that injection hooks execute. Layer indices are zero-based; Figure 2(b) plots ℓ/L.

At the selected block's residual-stream input $h \in \mathbb{R}^{B\times T\times C}$, the evaluator independently selects each tensor element with probability 0.01 and applies

$$M_{btc} = \mathbf{1}\{U_{btc} < 0.01\}, \quad U_{btc} \sim \mathrm{Uniform}[0,1), \qquad h'_{btc} = (1 + 99M_{btc})h_{btc}.$$

The mask has the full $[B, T, C]$ shape; selected channels are not broadcast across tokens. A device-local generator is reset to seed 0 for each probe, so identical tensor shapes reproduce the same mask across arms, sampled layers and training seeds. Different model widths have different mask shapes. The mask is generated once and reused for the loss and logit forwards. A `forward_pre_hook` injects before residual branching, so both the normalized branch and bypass receive h'.

Loss is PyTorch cross-entropy with mean reduction over valid target tokens, using natural logarithms: $\mathcal{L} = -|\mathcal{V}|^{-1} \sum_{(b,t)\in\mathcal{V}} \log p_\theta(y_{bt})$, with ignored targets excluded. Thus both loss and its spike-induced increment are in nats per token, not bpb. The evaluation uses four sequences of length 2,048; probe A uses only the first loaded batch (8,192 tokens), although four batches are loaded. Training seeds index checkpoint replications, not new masks. A reporting-only d16 checkpoint-tag error is documented with the source data; the model depth, norm-module count, and numerical observations are unaffected.

Table 16 reports both arms' mean loss increases and $G_\ell = \Delta\mathcal{L}_{\mathrm{RMS}} - \Delta\mathcal{L}_{\mathrm{Rho}}$. All means and sample SDs are computed from the training-seed values. Figure 2(b) plots each arm's extra loss directly, with its own sample SD. Relative reductions quoted in the text instead normalize each gain by its pair's RMSNorm loss increase before averaging; these are not ratios of aggregate means. All twelve d12/d16/d24 conditions at or beyond half depth improve in all five pairs, as do all three such d32 conditions in each of three pairs. At 75% depth, d32's paired gain is 8.686 ± 0.756, and its relative reduction is $57.7 \pm 1.9\%$. The smaller observed SD of the RhoNorm loss increase is descriptive across three checkpoints; it does not establish a population variance reduction.

Shallow effects should be interpreted cautiously. The negative d12/d16 contrasts do not survive Holm correction across all 22 measured conditions at 0.05. For d32, shallow paired t statistics are 2.845 and 1.851, with two-sided unadjusted $p = 0.105$ and 0.205 (two degrees of freedom). At layer 8, one pair worsens. Thus positive shallow means do not establish significant improvement. These probes condition on the evaluation inputs and mask protocol, and do not determine whether shallow normalizers should be removed during training.

Clean d12 activation probes provide a negative control. Mean FP8-input absolute maxima are 105.4 for RMSNorm and 408.0 for RhoNorm, with one projection site contributing much of the difference. Excluding that site leaves means of 48.2 and 58.1. These data provide no basis for claiming smaller natural activation tails or improved tensorwise FP8 dynamic-range utilization.

Table 16: Depth sweep at $k = 100$: mean $\pm$ sample SD across five paired training seeds for d12/d16/d24 and three for d32. Each loss increase uses its model's own clean loss. Positive G_ℓ favors RhoNorm.

Model	Layer ℓ	Pairs	RMSNorm $\Delta\mathcal{L}$	RhoNorm $\Delta\mathcal{L}$	Gain G_ℓ
d12	3	5	2.906 ± 0.202	**3.127** ± 0.131	$\downarrow 0.220 \pm 0.141$
d12	6	5	6.521 ± 0.158	**5.217** ± 0.150	$\uparrow 1.304 \pm 0.061$
d12	9	5	9.328 ± 0.100	**7.178** ± 0.121	$\uparrow 2.150 \pm 0.064$
d12	11	5	8.595 ± 0.166	**7.182** ± 0.170	$\uparrow 1.413 \pm 0.125$
d16	4	5	3.042 ± 0.030	**3.197** ± 0.061	$\downarrow 0.155 \pm 0.076$
d16	8	5	7.506 ± 0.189	**5.689** ± 0.103	$\uparrow 1.817 \pm 0.222$
d16	12	5	10.830 ± 0.173	**6.744** ± 0.161	$\uparrow 4.086 \pm 0.194$
d16	14	5	10.240 ± 0.180	**6.891** ± 0.188	$\uparrow 3.349 \pm 0.187$
d24	3	5	2.641 ± 0.155	**2.778** ± 0.232	$\downarrow 0.136 \pm 0.302$
d24	6	5	3.675 ± 0.300	**3.551** ± 0.287	$\uparrow 0.123 \pm 0.521$
d24	9	5	5.669 ± 0.201	**5.379** ± 0.257	$\uparrow 0.291 \pm 0.246$
d24	12	5	8.274 ± 0.211	**5.576** ± 0.051	$\uparrow 2.698 \pm 0.232$
d24	15	5	12.852 ± 0.163	**6.586** ± 0.124	$\uparrow 6.266 \pm 0.094$
d24	18	5	12.536 ± 0.178	**6.184** ± 0.128	$\uparrow 6.352 \pm 0.198$
d24	21	5	12.040 ± 0.197	**6.261** ± 0.131	$\uparrow 5.779 \pm 0.248$
d24	22	5	11.813 ± 0.195	**6.470** ± 0.155	$\uparrow 5.343 \pm 0.254$
d24	23	5	11.447 ± 0.156	**6.907** ± 0.184	$\uparrow 4.540 \pm 0.298$
d32	4	3	2.945 ± 0.398	**2.445** ± 0.108	$\uparrow 0.500 \pm 0.305$
d32	8	3	4.219 ± 0.439	**3.551** ± 0.300	$\uparrow 0.668 \pm 0.625$
d32	16	3	8.372 ± 0.521	**5.588** ± 0.072	$\uparrow 2.785 \pm 0.587$
d32	24	3	15.025 ± 0.827	**6.338** ± 0.078	$\uparrow 8.686 \pm 0.756$
d32	31	3	13.618 ± 0.599	**6.911** ± 0.191	$\uparrow 6.707 \pm 0.727$

E.2 Systems comparisons and timing boundaries

Table 17 reports the separate systems cohort on eight H100 GPUs. At 5,568 updates, RhoNorm reduces bpb by 0.002601 while increasing training-step time by 5.1%. These runs use their own paired controls and are distinct from the full-CORE horizons below. Times exclude graph capture, evaluation, and logging; they are not complete job durations.

Table 17: Paired systems comparison: d24, Full scope, 5,568 updates, five seeds, eight H100 GPUs. Values are mean $\pm$ sample SD. Times count training steps, excluding graph capture and evaluation.

Method	Validation bpb	Training-step time (min)
RMSNorm (compile)	0.716650 ± 0.000145	99.56 ± 0.03
RhoNorm (graph + fusion)	**0.714049** ± 0.000156	104.62 ± 0.03

E.3 Complete full-CORE horizon results

Table 18 contains all five independently trained horizons, with five seeds and full CORE per horizon, from the reported aggregate summary. Each row retains that cohort's own bpb, CORE mean and SD, and reported training/evaluation/total times. Per-seed and per-task outputs are unavailable, so these aggregates do not supply seed pass counts or paired cross-horizon tests.

Full coverage corresponds to `--core-metric-max-per-task=-1` during training or `--max-per-task=-1` in the separate evaluator. The recorded protocol specifies full coverage. Approximate intervals use $\bar{C} \pm t_{4,0.975} s/\sqrt{5}$, treating the reported SD as sample SD. They are individual horizon intervals, not a confidence interval for a selected time-to-target crossing. The 4,700-update mean exceeds the reference by 0.001675, but its interval includes it. This establishes an observed mean crossing on the tested grid, not statistical certification of an exact minimum training time.

Table 18: All full-CORE horizons and their uncertainty. Each horizon has five training seeds. Intervals are unadjusted Student intervals for mean CORE from the rounded aggregates. Gray intervals include the GPT-2 reference 0.256525. Times are the separately reported training, evaluation, and total minutes; the 4,300-step total is retained despite rounding of its components.

Updates	Train min	Eval min	Total min	Val. bpb	Full CORE	95% mean interval
4,000	75.3	9.2	84.5	0.726750	0.2507 ± 0.0048	$[0.2447, 0.2567]$
4,300	81.3	9.4	90.6	0.723813	0.2538 ± 0.0072	$[0.2449, 0.2627]$
4,700	88.0	8.7	96.7	0.720344	0.2582 ± 0.0080	$[0.2483, 0.2681]$
5,000	93.5	10.4	103.9	0.717947	0.2603 ± 0.0040	$[0.2553, 0.2653]$
5,568	105.5	11.4	116.9	0.714022	0.2639 ± 0.0060	$[0.2565, 0.2713]$

The RhoNorm horizons and published Autoresearch comparison use the same hardware configuration: eight NVIDIA H100 GPUs, as confirmed by the investigators. The benchmark counts training iterations, excluding evaluation and logging; the horizon summary separately records training, evaluation and total time. The 88-versus-99-minute comparison retains the training-time category on matched hardware. It compares complete training configurations and is distinct from the paired systems comparison above.