TRAINABLE AFFINE PARAMETERS IN NORMALIZATION CAN SILENTLY HARM ANALOG TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Analog in-memory computing (AIMC) offers a route to energy-efficient training by accelerating matrix–vector multiplications (MVMs) in memory. With finite-resolution analog MVMs in both forward and backward passes, trainable normalization gain and bias before the final analog linear layer can degrade training, despite remaining digital and unquantized. Part of the loss gap relative to fixed gain and bias persists when the trained head is replayed with exact multiplication, indicating damage retained after training. To understand this failure, we show how affine parameters can increase absolute logit error through changes in input scale and direction, despite dynamic max-absolute scaling. Training can fail to correct these effects because the supplied gain gradient can predict a loss response opposite to that measured by forward evaluations. These findings motivate parameter-free normalization before the digital-to-analog converter, with any needed affine placed after the analog-to-digital converter. Normalizing the digitized output can also mitigate the damage. Language and vision experiments reproduce the failure and its repairs across converter precisions, optimizers, and model scales, with repair benefits depending on the task, optimizer, and operating point.

1 INTRODUCTION

Language and vision models commonly normalize the hidden state with trainable affine parameters immediately before the final linear layer, using LayerNorm with trainable gain and bias or root-mean-square normalization (RMSNorm) with trainable gain (Zhang & Sennrich, 2019; Radford et al., 2019; Dosovitskiy et al., 2021; Touvron et al., 2023). Gain scales hidden channels and bias shifts them; together they form the *affine*. The final linear layer and its surrounding operations make up the model's *readout*, which produces logits for softmax and cross-entropy (CE).

On analog hardware, this affine directly shapes the signal that finite-resolution converters must encode. Analog in-memory computing (AIMC) implements linear layers on crossbar arrays that compute matrix–vector multiplication (MVM) within a fixed dynamic range (Gokmen & Vlasov, 2016; Sebastian et al., 2020). Mixed-precision training uses analog MVMs in both the forward and backward passes, while accumulating weight updates digitally at high precision and transferring them to the devices as programming pulses (Nandakumar et al., 2020). A digital-to-analog converter (DAC) encodes the input after rescaling it to that range, and an analog-to-digital converter (ADC) digitizes the output before its scale is restored. The affine itself stays digital and unquantized, yet it sits just before the DAC, the placement used by deployed analog language models, whose mapped head follows a digital normalization with trainable gain (Büchel et al., 2025).

The trainable affine adds parameters to the readout, so one would expect it to help, or at worst be harmless. In a Transformer (Vaswani et al., 2017) trained on enwik8 (Mahoney, 2011), the final LayerNorm affine (Ba et al., 2016) indeed barely changes digital validation trajectories. With an analog readout and 6-bit forward and backward MVMs, however, it stalls improvement (Figure 1), whereas removing it, or training an affine after the ADC, allows validation loss to keep improving.

This observation raises a natural question:

How does a pre-DAC trainable affine silently harm analog training, and how can it be repaired?

In this work, we investigate this failure and its remedies through four main aspects:

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.

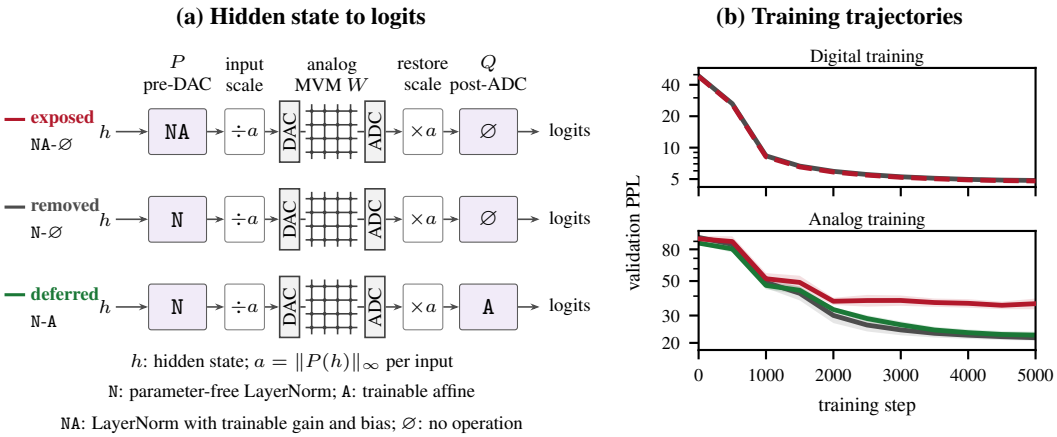


Figure 1: **Trainable affine parameters have little effect in digital training but impair analog training at the primary setting.** (a) Three separately trained readouts differing in affine placement. (b) Their validation perplexity (PPL) under digital training (top) and analog training with 6-bit converters in both forward and backward MVMs (bottom). Colors match the readouts in (a).

Q1) Is this degradation only the error that the converters add when the trained head is evaluated? Replacing only the trained head with exact multiplication removes only part of the loss gap, and the model trained with the affine still has higher loss. Conversely, when we retain the analog trunk but use an exact final linear layer throughout training, we detect no degradation from the affine. The degradation therefore combines an immediate conversion cost with damage accumulated while training through the analog readout. Separate layerwise controls show the largest increase in local conversion damage at the final readout (Section 3).

Q2) Why should a digital affine enlarge this conversion cost when every input is rescaled to the converter range anyway? When input scale increases uniformly, the scale restored after the ADC amplifies absolute logit error while relative error stays unchanged. A channelwise gain can also enlarge the largest coordinates relative to the rest; because that maximum sets the shared converter range, the quantization grid becomes coarser relative to the bulk of the input (Section 4).

Q3) Why does training fail to correct these effects? The supplied gain gradient can predict a loss response opposite to that measured by forward evaluations. Replacing only its readout-penalty component with a finite-difference estimate nearly closes the gap to fixed normalization at our primary setting, supporting a causal role for misleading gain feedback (Section 5).

Readout design rules and scope. For readouts with per-input max-absolute scaling, these findings suggest keeping normalization parameter-free before the DAC and placing any needed affine after the ADC. If a trainable affine must stay before the DAC, consider normalizing the digitized output (Section 6). The failure recurs across model widths and depths, across learning rules, at 39.4M parameters, and on image classification. Converter precision sets its severity, while the benefit of each remedy depends on the task, optimizer, and operating point (Section 7).

2 Related Work

Readout placement in analog computing. AIMC systems recover accuracy with hardware-aware retraining (Sebastian et al., 2020; Gokmen & Vlasov, 2016; Rasch et al., 2021; 2023), but differ in where they place the digital–analog boundary around the readout. Reshape and Adapt for Output Quantization (RAOQ) keeps the first and last layers digital while using ADC-aware quantization-aware training (QAT) (Zhang et al., 2024). Analog Foundation Models expose a mapped language-model head after digital RMSNorm with trainable gain (Büchel et al., 2025), whereas Li et al. (2026a) apply a digital affine after the mapped output layer's ADCs. We show that where the trainable affine sits relative to the converters changes the outcome of low-precision training.

Activation rescaling and quantization geometry. Channel-scale reparameterization can improve quantization while preserving a trained model's floating-point function. Outlier Suppression identifies LayerNorm gain as an outlier amplifier and migrates it into the next weight matrix while preserving the floating-point function (Wei et al., 2022), and SmoothQuant redistributes channel scale between activations and weights for post-training quantization (Xiao et al., 2023). These function-preserving reparameterizations address conversion of a trained model. Repeated gain migration during training also repairs the failure at the primary operating point (Appendix G.5). Our ideal-head replay shows that part of the loss gap persists in the trained weights even after final-head conversion error is removed, motivating a safer readout during training (Section 3.2). The code invariance of dynamic max-absolute scaling under uniform rescaling is likewise shared by digital quantizers; what we add is how an unquantized gain changes the training loss through such a readout.

Analog training and surrogate gradients. Existing methods correct nonideal analog weight updates and estimate gradients through quantization. Asymmetric conductance response can make analog stochastic gradient descent (SGD) converge inexactly, and residual schemes remove the induced penalty (Wu et al., 2024; 2025), including with few conductance states (Li et al., 2026b). Quantized training commonly uses straight-through estimates (Bengio et al., 2013), and curvature-aware gradient estimation (CAGE) balances task loss against quantization constraints (Tabesh et al., 2026). The gain we study is neither quantized nor stored on the device, yet its feedback passes through the analog readout; we replace only that feedback to test whether it causes training damage, while the practical remedies change the readout structure.

3 The Failure Arises at the Final Analog Linear Layer

We first define the readout computation, then ask whether switching to a floating-point head after training recovers performance, whether using it throughout training avoids the affine's disadvantage, and whether internal normalization affines produce similar conversion damage.

3.1 From hidden representations to prediction scores

We write the readout as three steps so that its analog step can be isolated. For one item with target y, the *trunk* produces $h \in \mathbb{R}^d$, and the final linear layer, the *head*, has effective weights $W \in \mathbb{R}^{V \times d}$, where V is the vocabulary size or number of image classes. With digital input processing P and output processing Q, neither of which is a quantizer, the bias-free ideal readout is

$$h \xrightarrow{P} h_P := P(h) \xrightarrow{W} z^\star := W h_P \xrightarrow{Q} Q(z^\star).$$

In the analog readout, DAC encoding, analog MVM on crossbar tiles, and ADC digitization replace the W arrow. Digital P precedes input scaling and the DAC; digital Q follows the ADC and scale restoration.

We describe the digital processing before and after the analog linear layer using four operations. By default, `N` denotes parameter-free centered LayerNorm (Ba et al., 2016), and `NA` denotes LayerNorm with trainable gain and bias. `A` applies a trainable affine without normalization, and $\varnothing$ applies no operation. A configuration P-Q specifies the input operation before the DAC and the output operation after the ADC. For example, `NA`-$\varnothing$ trains the pre-DAC normalization gain and bias and applies no post-ADC operation. Its fixed-affine counterpart, `N`-$\varnothing$, keeps the gain at one and the bias at zero. These separately trained configurations differ only in whether the pre-DAC affine parameters are trained. The *deferred* configuration, `N-A`, instead uses parameter-free normalization before the DAC and a trainable affine after the ADC. This output affine acts on the V logits, whereas the input affine acts on the d hidden channels. The two therefore have different parameterizations: deferred training introduces an output affine rather than moving the original input affine through the weight matrix.

The affine changes both the vector presented to the converters and the input-dependent scale used to encode it. For $P = \texttt{NA}$, the affine before the DAC gives

$$h_P = P(h) = \boldsymbol{\gamma} \odot \texttt{N}(h) + \boldsymbol{\beta}, \qquad a := \|h_P\|_\infty. \tag{1}$$

Here $\odot$ is coordinatewise multiplication; $P = \texttt{N}$ fixes $\boldsymbol{\gamma} = \mathbf{1}$ and $\boldsymbol{\beta} = \mathbf{0}$. For a bias-free, noiseless single tile with $a > 0$, the converters $\mathrm{Quant}_{\mathrm{DAC}}$ and $\mathrm{Quant}_{\mathrm{ADC}}$ give

$$\tilde{z} := a\ \mathrm{Quant}_{\mathrm{ADC}}(W\ \mathrm{Quant}_{\mathrm{DAC}}(h_P/a)), \qquad \ell(\tilde{z}, y) = \mathrm{CE}(Q(\tilde{z}), y). \tag{2}$$

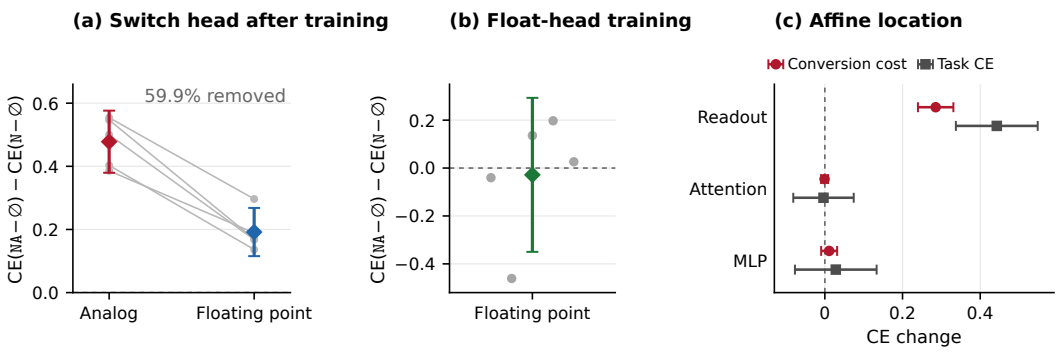


Figure 2: **Head replay, floating-point-head training, and layerwise localization.** Trainable-minus-fixed affine effects for (a) analog-trained models before and after ideal-head replay, (b) models trained with a floating-point head, and (c) local conversion cost and task CE across normalization layers.

Here $\tilde{z}$ is the digitized head output after scale restoration, including conversion error relative to the ideal output $z^\star = Wh_P$. Applying Q gives the prediction scores (logits), and $\ell(z, y) := \mathrm{CE}(Q(z), y)$ is the cross-entropy loss for one prediction item with ground-truth label y. The converter bounds are fixed, with mapping gains absorbed into the matrix and converter units, so the input-dependent a is divided out before conversion and restored afterward.

Unless stated otherwise, results use backpropagation (BP) on enwik8 with a four-block, width-128 Transformer (d128-L4) and 6-bit converters in both forward and backward MVMs. All parameters, including the analog head, share one learning rate. Appendix A gives the full model, converter, and training settings. Table cells give mean ± standard deviation (SD), paired differences give mean ± standard error (SE), and brackets give two-sided 95% confidence intervals (CIs).

3.2 A FLOATING-POINT HEAD DOES NOT FULLY REPAIR TRAINING DAMAGE

Damage from training through the analog head persists in the learned network parameters. We freeze the trained NA-∅ and N-∅ models and replace only the final analog multiplication with exact multiplication using the same learned weights. This *ideal-head replay* reduces but does not eliminate the affine's loss gap (Figure 2(a)). Switching the head to digital floating-point computation after training therefore provides only a partial repair.

With a floating-point head throughout training, we detect no performance disadvantage from training the affine. We train models with trainable or fixed final normalization affines, retaining the analog trunk but using floating-point head computation from initialization (Figure 2(b)).

The added conversion damage is concentrated at the final readout. Under the primary training pipeline, we independently fix or train the normalization affines at the readout, attention, and multilayer-perceptron (MLP) blocks. For each group, we measure its local conversion cost as the decrease in CE when only the linear layers immediately following its normalizations are evaluated by exact multiplication with unchanged weights. Figure 2(c) shows trainable-minus-fixed differences in this cost (red) and in task CE (gray), averaged over the other groups' affine settings. Training the readout affine increases both quantities, with effects significantly larger than those of either internal group. The averaged attention and MLP effects are not statistically distinguished from zero.

4 WHY MAX-ABSOLUTE SCALING DOES NOT PROTECT THE READOUT

We first isolate the effect of uniform rescaling, then examine how channelwise affine parameters change the input direction. Both can change the absolute logit perturbation, which we then relate to readout loss.

Both effects act on the logit error before Q, which we measure with the checkpoint, device realization, replay random state, and evaluation items $\mathcal{E} := \{(x_n, y_n)\}_{n=1}^{N}$ held fixed. Item n has ideal logits

$z_n^\star = Wh_{P,n} \in \mathbb{R}^V$, digitized logits $\tilde{z}_n$, and conversion error in the logits $\epsilon_n := \tilde{z}_n - z_n^\star$, with coordinates $z_{n,v}^\star$ and $\epsilon_{n,v}$ for $v = 1, \ldots, V$. We use n to index evaluation items and v to index logit coordinates, omitting these indices when referring to the logit vectors of a single item. For a vector $u = (u_1, \ldots, u_m)$, its root-mean-square (RMS) value is $\mathrm{RMS}(u) := \sqrt{\frac{1}{m}\sum_{i=1}^m u_i^2}$. For each checkpoint, we compute RMS over all evaluation items and logit coordinates:

$$z_{\mathrm{rms}}(h_P) := \mathrm{RMS}(z^\star), \qquad \delta(h_P) := \mathrm{RMS}(\epsilon), \qquad \rho(h_P) := \frac{\delta(h_P)}{z_{\mathrm{rms}}(h_P)}. \tag{3}$$

These are ideal-logit RMS, absolute perturbation, and relative corruption, with $\delta = \rho z_{\mathrm{rms}}$.

4.1 Uniform scaling amplifies absolute conversion error

Dynamic max-absolute scaling divides the input by $a = \|h_P\|_\infty$ before conversion and restores the scale afterward. Multiplying h_P by $c > 0$ also multiplies a by c, so the converter receives the same vector $ch_P/(ca) = h_P/a$. With fixed weights, no head bias, and zero read noise, the ADC codes stay unchanged. The restored output and ideal logits both scale by c, as does their difference, giving

$$\rho(c\,h_P) = \rho(h_P), \qquad z_{\mathrm{rms}}(c\,h_P) = c\,z_{\mathrm{rms}}(h_P), \qquad \delta(c\,h_P) = c\,\delta(h_P). \tag{4}$$

4.2 Channelwise affine parameters can change the input direction

A channelwise affine can amplify some coordinates more than others, changing the input direction and the normalized vector presented to the DAC. LayerNorm gains can amplify outlier channels in Transformers (Wei et al., 2022). For h_P, define its scale $s(h_P) := \|h_P\|_2/\sqrt{d}$, direction $h_P/s(h_P)$, and peak-to-RMS ratio $r_{\mathrm{pk}}(h_P) := \|h_P\|_\infty/s(h_P)$. Under max-absolute scaling, the largest coordinate sets the quantization range shared by all channels. If this coordinate grows relative to the input RMS, one RMS spans fewer quantization intervals. This coarsens input resolution relative to the signal and can increase relative corruption ρ, although the resulting logit perturbation also depends on the head weights and ADC.

The increased perturbation cannot be explained by uniform scaling alone. Figure 3 compares separately trained checkpoints on fixed evaluation batches. In every paired seed, `NA-∅` has a larger logit scale than `N-∅` (Figure 3(a)). Its input peak-to-RMS also increases (Figure 3(b)), indicating a change in input direction. Relative corruption is also higher (Figure 3(c)). Uniform rescaling would preserve both peak-to-RMS and relative corruption. Together, larger z_{rms} and ρ nearly double the absolute perturbation $\delta = \rho z_{\mathrm{rms}}$ (Figure 3(d)).

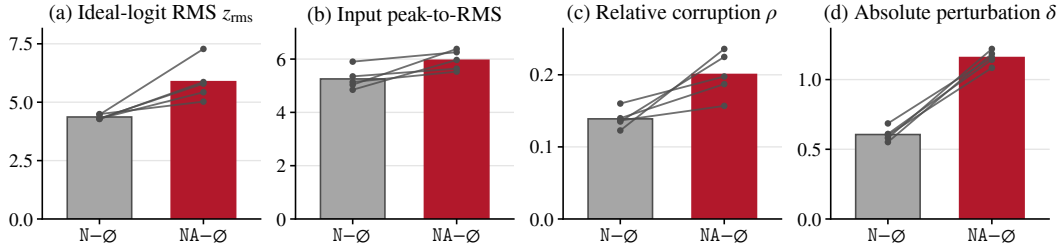


Figure 3: **Paired scale and input-direction diagnostics.** Bars are five-seed means; lines join each seed's paired checkpoints.

Matching the logit scale does not eliminate the difference. We uniformly rescale each frozen `N-∅` checkpoint's input to match its paired `NA-∅` checkpoint's ideal-logit RMS. The trained `NA-∅` models still have greater absolute perturbation and readout loss penalty (Appendix D.4). The residual includes input-direction changes and trunk/head co-adaptation.

4.3 A squared-margin estimate of readout damage

Both scale and input direction can change the absolute perturbation δ. Uniform scaling changes z_{rms} while preserving ρ; input-direction changes can alter ρ as well as z_{rms}. Thus $\delta = \rho z_{\mathrm{rms}}$ measures the resulting logit perturbation from both effects.

To relate δ to task loss, we evaluate the same frozen checkpoint and items with either the analog head or floating-point multiplication using the same weights, keeping the trunk, P, and Q unchanged. We write $\mathcal{L}(z) := \mathbb{E}_n[\ell(z_n, y_n)]$, where $\mathbb{E}_n$ is the uniform average over the fixed evaluation items. Thus $\mathcal{L}(\tilde{z})$ is the loss with analog head computation, and $\mathcal{L}(z^\star)$ is the loss with floating-point head computation using the same weights. Their difference $D_{\mathrm{ro}} := \mathcal{L}(\tilde{z}) - \mathcal{L}(z^\star)$ is the *readout loss penalty*. Cross-entropy depends on logit differences, so δ alone does not determine this penalty.

Proposition 1 (Squared-margin estimate of readout damage, informal)**.** *For small logit errors and a bare output, $Q = \varnothing$, suppose the linear and higher-order loss terms are small relative to the quadratic term. Also suppose error directions do not favor unusually high or low loss curvature, and error energy does not systematically vary with loss curvature across prediction items. Then*

$$D_{\mathrm{ro}} \approx \tfrac{1}{2}\,\delta^2\,\mathbb{E}_n[1 - \|p_n\|^2], \tag{5}$$

where $p_n := \mathrm{softmax}(z_n^\star)$ and $\mathbb{E}_n$ averages over evaluation items.

Under these conditions, either pathway can increase the quadratic contribution to readout loss by increasing δ. The ideal predictions set the coefficient, which can also change with the affine. Appendix D.6 gives the exact loss decomposition and the scope of this approximation.

5 Why Training Does Not Fix the Failure

The preceding analysis shows how affine parameters can affect conversion error and readout loss. Training should account for these effects when updating the affine parameters. We therefore investigate whether the gradient propagated through the analog readout correctly predicts the loss response, and whether correcting this gradient improves training.

5.1 The supplied gain gradient predicts the wrong loss response

For the pre-DAC affine $h_P = \boldsymbol{\gamma} \odot \mathtt{N}(h) + \boldsymbol{\beta}$, backpropagation proceeds through the analog readout and then through the digital affine. The resulting gradient with respect to $\boldsymbol{\gamma}$ is the *supplied gain gradient*, denoted by $\widetilde{\nabla}_{\boldsymbol{\gamma}}\mathcal{L}(\tilde{z})$. During training, this gradient is passed to the optimizer to update $\boldsymbol{\gamma}$. To test whether this gradient correctly predicts the loss response, we vary the log-gain change s through $\boldsymbol{\gamma} \mapsto e^s\boldsymbol{\gamma}$ at a frozen checkpoint and on a fixed batch, keeping the bias $\boldsymbol{\beta}$ and all other parameters fixed. We denote the analog and ideal logits after replacing $\boldsymbol{\gamma}$ with $e^s\boldsymbol{\gamma}$ by $\tilde{z}(e^s\boldsymbol{\gamma})$ and $z^\star(e^s\boldsymbol{\gamma})$, keeping all other parameters fixed. Figure 4(a) compares the measured CE change $\Delta\mathcal{L}_{\mathrm{measured}}$ and predicted CE change $\Delta\mathcal{L}_{\mathrm{predicted}}$,

$$\Delta\mathcal{L}_{\mathrm{measured}}(s) := \mathcal{L}(\tilde{z}(e^s\boldsymbol{\gamma})) - \mathcal{L}(\tilde{z}), \qquad \Delta\mathcal{L}_{\mathrm{predicted}}(s) := s\left\langle \widetilde{\nabla}_{\boldsymbol{\gamma}}\mathcal{L}(\tilde{z}), \boldsymbol{\gamma}\right\rangle. \tag{6}$$

Here $\Delta\mathcal{L}_{\mathrm{measured}}$ is the actual loss change caused by rescaling $\boldsymbol{\gamma}$ and $\Delta\mathcal{L}_{\mathrm{predicted}}$ is the first-order prediction from the supplied gain gradient at the unperturbed point $s = 0$. For $s < 0$, the measured change is negative but the predicted change is positive. Thus, shrinking $\boldsymbol{\gamma}$ lowers CE even though the supplied gain gradient predicts the opposite.

Figure 4(b) tracks this disagreement throughout training using the loss slope, the rate of loss change with respect to s at $s = 0$ along $\boldsymbol{\gamma} \mapsto e^s\boldsymbol{\gamma}$. The predicted slope $\bar{D}_{\mathrm{predicted}}$ is the supplied gain gradient's prediction of the loss response to gain rescaling. The ideal-loss slope $\bar{D}_{\mathrm{ideal}}$ measures the CE response to gain rescaling when only the final head is replaced by exact multiplication using the same weights. For d gain channels,

$$\begin{aligned}
\bar{D}_{\mathrm{ideal}} &:= \frac{1}{d}\,\frac{\mathrm{d}}{\mathrm{d}s}\mathcal{L}(z^\star(e^s\boldsymbol{\gamma}))\Big|_{s=0} = \frac{1}{d}\sum_{i=1}^{d}\frac{\partial\mathcal{L}(z^\star)}{\partial\gamma_i}\gamma_i = \frac{1}{d}\langle\nabla_{\boldsymbol{\gamma}}\mathcal{L}(z^\star), \boldsymbol{\gamma}\rangle, \\
\bar{D}_{\mathrm{predicted}} &:= \frac{1}{d}\sum_{i=1}^{d}\big[\widetilde{\nabla}_{\boldsymbol{\gamma}}\mathcal{L}(\tilde{z})\big]_i\gamma_i = \frac{1}{d}\langle\widetilde{\nabla}_{\boldsymbol{\gamma}}\mathcal{L}(\tilde{z}), \boldsymbol{\gamma}\rangle.
\end{aligned} \tag{7}$$

We estimate the actual slope by combining the ideal-loss slope with the finite-difference response of the readout penalty. For a small log-gain perturbation η,

$$\bar{D}_{\text{measured}} := \bar{D}_{\text{ideal}} + \frac{D_{\text{ro}}(e^{\eta}\boldsymbol{\gamma}) - D_{\text{ro}}(e^{-\eta}\boldsymbol{\gamma})}{2\eta d}. \tag{8}$$

Negative slopes favor gain growth; positive slopes favor shrinkage. Late in training, $\bar{D}_{\text{predicted}} < 0$ while $\bar{D}_{\text{measured}} > 0$, so the supplied gain gradient predicts that increasing gain lowers the loss, whereas the measured slope indicates the opposite. Therefore, the supplied gain gradient favors a change in $\boldsymbol{\gamma}$ that increases the loss instead of decreasing it.

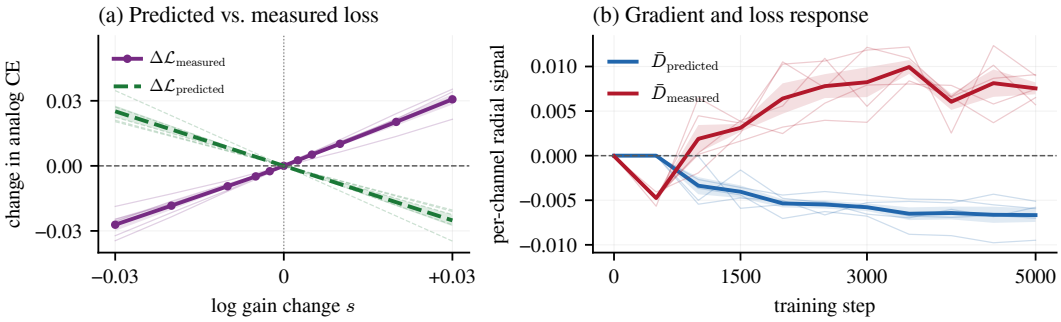


Figure 4: **Gain-gradient diagnostics.** (a) Measured and predicted CE changes under gain rescaling. (b) Predicted and measured-response estimates of the total-loss slope during training.

5.2 A SCALAR EXAMPLE EXPLAINS A SOURCE OF MISMATCH

We simulate analog layers with AIHWKit, an analog crossbar simulation toolkit (Rasch et al., 2021), which supplies a separate transposed-MVM *surrogate backward*; normalization parameters stay digital and update from its output. For one prediction item in the exposed cell, the CE gradient with respect to the analog logits $\tilde{z}$ is $e := \nabla_{\tilde{z}}\ell(\tilde{z}, y) = \mathrm{softmax}(\tilde{z}) - e_y$, where e_y is the one-hot target. Let $\mathcal{B}_W(e)$ be the possibly quantized input-gradient signal returned by the backward MVM. The supplied gain and bias gradients are

$$\widetilde{\nabla}_{\boldsymbol{\gamma}}\ell(\tilde{z}, y) = \mathtt{N}(h) \odot \mathcal{B}_W(e), \qquad \widetilde{\nabla}_{\boldsymbol{\beta}}\ell(\tilde{z}, y) = \mathcal{B}_W(e). \tag{9}$$

To isolate a source of mismatch independent of backward quantization, consider a floating-point backward MVM, $\mathcal{B}_W(e) = W^{\top}e$. To study uniform input scaling, we restrict the gain vector to a shared scalar $g > 0$, so $\boldsymbol{\gamma} = g\mathbf{1}$, and set $\boldsymbol{\beta} = 0$. Here $\mathbf{1}$ is the all-ones vector, and $h_P(g) = g\mathtt{N}(h)$. Under dynamic scaling, with fixed $\mathtt{N}(h)$ and W and zero read noise, the analog logits satisfy $\tilde{z}(g) = g\tilde{z}(1)$ and the ideal logits satisfy $z^{\star}(g) = gz^{\star}(1)$. Consequently, the logit error satisfies $\epsilon(g) = g\epsilon(1)$, so its magnitude grows linearly with g. For one prediction item, $\mathrm{d}\ell(\tilde{z}, y)/\mathrm{d}g$ measures how the analog CE changes with the shared gain g. Backpropagating the input-gradient signal $\mathcal{B}_W(e)$ through $h_P = g\mathtt{N}(h)$ gives the supplied scalar gain gradient, $\widetilde{\nabla}_g\ell(\tilde{z}, y) = \langle\widetilde{\nabla}_{\boldsymbol{\gamma}}\ell(\tilde{z}, y), \mathbf{1}\rangle$. The CE gradient at gain g is $e(g) = \mathrm{softmax}(\tilde{z}(g)) - e_y$. The actual loss derivative and the supplied scalar gain gradient satisfy

$$\begin{aligned} \frac{\mathrm{d}\ell(\tilde{z}(g), y)}{\mathrm{d}g} &= e(g)^{\top}\frac{\mathrm{d}\tilde{z}(g)}{\mathrm{d}g} = e(g)^{\top}\tilde{z}(1), \\ \widetilde{\nabla}_g\ell(\tilde{z}(g), y) &= (W^{\top}e(g))^{\top}\frac{\mathrm{d}h_P(g)}{\mathrm{d}g} = e(g)^{\top}W\mathtt{N}(h) = e(g)^{\top}z^{\star}(1). \end{aligned} \tag{10}$$

Therefore,

$$\frac{\mathrm{d}\ell(\tilde{z}(g), y)}{\mathrm{d}g} - \widetilde{\nabla}_g\ell(\tilde{z}(g), y) = e(g)^{\top}\left[\tilde{z}(1) - z^{\star}(1)\right] = e(g)^{\top}\epsilon(1). \tag{11}$$

Equation (11) shows that the supplied gain gradient misses the loss change caused by scaling the conversion error. For the gain changes used in the preceding diagnostic, $g \mapsto e^s g$,

$$\begin{aligned} \left[\frac{\mathrm{d}\ell(\tilde{z}(e^s g), y)}{\mathrm{d}s} - \widetilde{\nabla}_s\ell(\tilde{z}(e^s g), y)\right]\Big|_{s=0} &= g\left[\frac{\mathrm{d}\ell(\tilde{z}(g), y)}{\mathrm{d}g} - \widetilde{\nabla}_g\ell(\tilde{z}(g), y)\right] \\ &= e(g)^{\top}\epsilon(g) = g\,e(g)^{\top}\epsilon(1). \end{aligned} \tag{12}$$

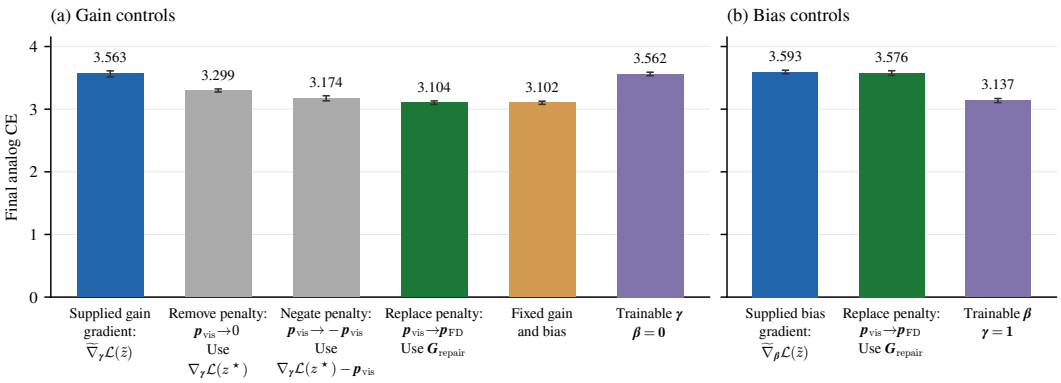


Figure 5: **Gain and bias controls.** Mean final analog CE with standard-error bars. Purple bars show gain-only and bias-only training from a separate paired experiment.

Increasing g scales both the ideal logits and the conversion error, which together form the analog logits $\tilde{z}(g) = z^{\star}(g) + \epsilon(g)$. However, the backward rule omits the loss contribution from the change in conversion error. Computing $W^{\top} e(g)$ more accurately does not recover this missing contribution. If the missing contribution makes the supplied gradient opposite in sign to the actual derivative, then a sufficiently small gain update along the negative supplied gradient increases the actual loss. Thus, even with floating-point backward, training can receive a gain gradient that points the wrong way.

5.3 Gradient interventions test the training consequences

Based on the preceding analysis, we expect that correcting the supplied gradients will reduce training damage. To test this expectation, we replace the penalty component of the supplied gain or bias gradient with a finite-difference estimate, while retaining the ideal-head gradient.

For the selected parameter $\boldsymbol{\theta} \in \{\boldsymbol{\gamma}, \boldsymbol{\beta}\}$, the corrected gradient $\boldsymbol{G}_{\text{repair}}$ is

$$\begin{aligned} \boldsymbol{p}_{\text{vis}} &:= \widetilde{\nabla}_{\boldsymbol{\theta}}\mathcal{L}(\tilde{z}) - \nabla_{\boldsymbol{\theta}}\mathcal{L}(z^{\star}), \qquad \boldsymbol{p}_{\text{FD}} := \widehat{\nabla}_{\boldsymbol{\theta}}^{\text{FD}} D_{\text{ro}}, \\ \boldsymbol{G}_{\text{repair}} &:= \widetilde{\nabla}_{\boldsymbol{\theta}}\mathcal{L}(\tilde{z}) - \boldsymbol{p}_{\text{vis}} + \boldsymbol{p}_{\text{FD}} = \nabla_{\boldsymbol{\theta}}\mathcal{L}(z^{\star}) + \boldsymbol{p}_{\text{FD}}. \end{aligned} \tag{13}$$

Here $\widehat{\nabla}_{\boldsymbol{\theta}}^{\text{FD}}$ denotes a finite-difference gradient estimate computed from forward losses. Each probe perturbs only the selected parameter and compares analog and ideal-head losses on the same batch. After backpropagation, we replace the selected parameter's gradient with $\boldsymbol{G}_{\text{repair}}$ from Eq. (13), leaving all other gradients unchanged. We then clip gradients and perform the optimizer update.

Figure 5(a) shows that replacing the gain-penalty component nearly closes the analog CE gap to fixed normalization. Removing or negating it also improves CE, supporting a causal role for harmful gain feedback. Training gain alone with bias fixed at zero also reproduces the degradation. In Figure 5(b), bias-penalty replacement yields no detected CE improvement over its own baseline, while training bias alone with gain fixed at one yields no detected CE increase over its matched fixed-affine baseline.

6 Breaking the Failure Before and After the Converters

Post-ADC normalization can mitigate readout damage. When the input is uniformly enlarged, dynamic scaling keeps the converter input unchanged, but restoring the scale afterward amplifies the absolute logit error (Section 4.1). Normalizing after the ADC cancels this common amplification of signal and error. Specifically, the ideal and digitized logits become $cz^{\star}$ and $c(z^{\star} + \epsilon)$, respectively. Ignoring the numerical stabilizer of $\mathtt{N}$, $\mathtt{N}(cz) = \mathtt{N}(z)$ for $c > 0$, so the error after normalization satisfies $\mathtt{N}(c(z^{\star} + \epsilon)) - \mathtt{N}(cz^{\star}) = \mathtt{N}(z^{\star} + \epsilon) - \mathtt{N}(z^{\star})$. Normalization therefore blocks uniform-scale amplification. A subsequent affine $\mathtt{A}(u) := \boldsymbol{a} \odot u + \boldsymbol{b}$ provides trainable output scale and bias while preserving this protection, since $\mathtt{A}(\mathtt{N}(cz)) = \mathtt{A}(\mathtt{N}(z))$. Conversion error remains, and changes in input direction can still affect it. However, post-ADC normalization also changes logit differences and hence softmax probabilities, so this protection does not guarantee lower task loss.

Comparing pre-DAC and post-ADC operations yields a design rule. We now test two structural remedies—removing the pre-DAC affine and normalizing the digitized output—without additional gradient estimation, using the 2×4 comparison of $P \in \{\texttt{N}, \texttt{NA}\}$ and $Q \in \{\varnothing, \texttt{N}, \texttt{A}, \texttt{NA}\}$ in Table 1. Fixing the pre-DAC affine lowers both mean analog PPL and readout PPL cost relative to the exposed NA-$\varnothing$ readout. A trainable post-ADC affine retains this improvement. When the pre-DAC affine remains trainable, normalizing the digitized output also lowers both metrics.

Table 1: **Operations before the DAC and after the ADC on enwik8.** Rows give P and columns Q. Entries give analog validation PPL / readout PPL cost $\mathrm{PPL}_{\mathrm{analog}} - \mathrm{PPL}_{\mathrm{ideal}}$ at the same checkpoint.

$P \backslash Q$	$\varnothing$	N	A	NA
N	21.64 ± 2.33 / $+3.17$	22.35 ± 1.35 / $+1.40$	22.29 ± 2.66 / $+2.40$	22.43 ± 4.00 / $+1.78$
NA	34.96 ± 4.51 / $+12.61$	22.63 ± 0.81 / $+1.45$	26.40 ± 4.72 / $+4.84$	19.75 ± 1.11 / $+1.15$

> **TRAINING-TIME INTERFACE RULE.** *With per-input max-absolute scaling, use parameter-free normalization before the DAC and place any needed affine after the ADC. If a trainable affine must remain before the DAC, consider post-ADC normalization when it lowers the final task loss.*

7 EXTENDED RESULTS

Table 2 extends the readout comparisons across models, tasks, and training methods. The d256-L4, 32k-vocabulary, RMSNorm 39.4M, and vision comparisons use 15,000 steps; LayerNorm 39.4M uses 20,000 steps and ten seeds. All other rows use five seeds. Vision uses compact vision Transformers (ViTs). RMSNorm changes only readout normalization, with gain-only affines. Local cross-entropy (LCE) uses block-local losses; quantization-aware training (QAT) adapts digitally pretrained models. Appendix H gives full settings.

Table 2: **Readout placement across settings.** Analog endpoints are mean $\pm$ SD; lower is better.

Setting	Metric	NA-$\varnothing$ (exposed)	N-$\varnothing$ (removed)	N-A (deferred)	NA-N
enwik8, d256-L4	PPL	65.04 ± 36.67	22.14 ± 2.32	13.03 ± 0.76	17.20 ± 0.83
enwik8, 32k subword vocab.	PPL	3358 ± 300	327 ± 48	386 ± 35	461 ± 42
enwik8, 39.4M parameters	PPL	350.8 ± 382.5	39.5 ± 7.0	14.2 ± 0.5	17.5 ± 1.0
CIFAR-100, ViT	CE	2.7748 ± 0.0524	2.7503 ± 0.0456	2.7219 ± 0.0421	2.8615 ± 0.0268
TinyImageNet-200, ViT	CE	3.5276 ± 0.0392	3.4570 ± 0.0287	3.4592 ± 0.0137	3.6735 ± 0.0194
enwik8, RMSNorm 39.4M	CE	4.8647 ± 0.4547	3.7272 ± 0.2282	2.8725 ± 0.0719	3.3187 ± 0.0473
enwik8, LCE	PPL	32.99 ± 1.10	23.64 ± 2.00	23.04 ± 1.54	25.07 ± 1.51
enwik8, QAT	CE	2.5173 ± 0.0248	2.4314 ± 0.0141	2.4786 ± 0.0146	2.7596 ± 0.0598
CIFAR-100 ViT, QAT	CE	2.2241 ± 0.0247	2.1408 ± 0.0237	2.1710 ± 0.0237	2.3544 ± 0.0190

8 LIMITATIONS AND CONCLUSION

Limitations. Our evidence comes from AIHWKit (Rasch et al., 2021) simulations of Transformers up to 39.4M parameters and compact vision models; validation on larger models and analog hardware remains open. Repair benefits depend on the task and optimizer. The finite-difference diagnostic requires ideal-head losses and gradients, and its feasibility as a practical hardware training method remains unevaluated. The relative contributions of gain magnitude, channelwise allocation, and trunk/head co-adaptation to the training gap remain unresolved.

Conclusion. An exposed trainable affine can damage analog training beyond its immediate conversion cost. Max-absolute scaling does not protect the readout because the affine's scale returns as amplified absolute logit error and its channelwise gain can coarsen input resolution. Training can fail to repair this damage because the gain receives misleading feedback. Correcting that feedback can remedy the damage. The results support parameter-free normalization before the DAC and any needed affine after the ADC, with benefits depending on the task and operating point.

AI USE STATEMENT

We used generative AI tools to assist with developing and checking mathematical arguments, designing experiments, implementing and debugging methods, and interpreting results. We also used these tools for manuscript drafting and revision, figure preparation, literature searches, and reference formatting. Verification included comparison with simulator code, numerical controls, and checks of reported values against saved experimental outputs. The authors take responsibility for the final manuscript, analyses, and results, including content produced with generative AI assistance.

ETHICS STATEMENT

This work studies numerical effects in analog-hardware simulation using existing public benchmarks (enwik8, CIFAR-100, and TinyImageNet-200). It does not involve human-subject experiments or the collection of new personal data.

REPRODUCIBILITY STATEMENT

Appendix A specifies the datasets, models, converter and device settings, training schedules, software versions, compute hardware, and statistical conventions. The appendices place assumptions and derivations alongside the corresponding analyses and experiments. The experiment-specific appendices document the intervention and evaluation protocols, pairing procedures, and settings that differ from the primary configuration.

REFERENCES

Mustafa Ali, Indranil Chakraborty, Utkarsh Saxena, Amogh Agrawal, Aayush Ankit, and Kaushik Roy. A 35.5–127.2 TOPS/W dynamic sparsity-aware reconfigurable-precision compute-in-memory SRAM macro for machine learning. *IEEE Solid-State Circuits Letters*, 4:129–132, 2021.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Luca Baroni, Galvin Khara, Joachim Schaeffer, Marat Subkhankulov, and Stefan Heimersheim. Small transformers don't need LayerNorm at inference time: Scaling LayerNorm removal to GPT-2 XL and implications for mechanistic interpretability. In *Proc. International Conference on Learning Representations*, Rio de Janeiro, Brazil, 2026.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, Usvsn Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. Pythia: A suite for analyzing large language models across training and scaling. In *Proc. International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 2397–2430, Honolulu, HI, 2023.

BigScience Workshop. BLOOM: A 176b-parameter open-access multilingual language model. *Journal of Machine Learning Research*, 25(422):1–74, 2024.

Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, Usvsn Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. GPT-NeoX-20B: An open-source autoregressive language model. In *Proc. BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, pp. 95–136, Dublin, Ireland, 2022.

Julian Büchel, Iason Chalas, Giovanni Acampa, An Chen, Omobayode Fagbohungbe, Sidney Tsai, Kaoutar El Maghraoui, Manuel Le Gallo, Abbas Rahimi, and Abu Sebastian. Analog foundation models. In *Proc. Advances in Neural Information Processing Systems*, volume 38, pp. 61179–61228, San Diego, CA, 2025.

Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, and Quoc V. Le. Symbolic discovery of optimization algorithms. In *Proc. Advances in Neural Information Processing Systems*, volume 36, pp. 49205–49233, New Orleans, LA, 2023.

DeepSeek-AI. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4171–4186, Minneapolis, MN, 2019.

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

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(61):2121–2159, 2011.

Gemma Team. Gemma: Open models based on Gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

Tayfun Gokmen and Yurii Vlasov. Acceleration of deep neural network training with resistive cross-point devices: Design considerations. *Frontiers in Neuroscience*, 10:333, 2016.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Raghavi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, Tushar Khot, William Merrill, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Valentina Pyatkin, Abhilasha Ravichander, Dustin Schwenk, Saurabh Shah, Will Smith, Emma Strubell, Nishant Subramani, Mitchell Wortsman, Pradeep Dasigi, Nathan Lambert, Kyle Richardson, Luke Zettlemoyer, Jesse Dodge, Kyle Lo, Luca Soldaini, Noah A. Smith, and Hannaneh Hajishirzi. OLMo: Accelerating the science of language models. In *Proc. Annual Meeting of the Association for Computational Linguistics*, pp. 15789–15809, Bangkok, Thailand, 2024.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Proc. Conference on Language Modeling*, Philadelphia, PA, 2024.

Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned stochastic tensor optimization. In *Proc. International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1842–1850, Stockholm, Sweden, 2018.

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

Haiqiao Hong, Zhiyuan Du, Mingrui Jiang, Ruibin Mao, Yuan Ren, Fuyi Li, Wei Mao, Muyuan Peng, Wei Zhang, Zhengwu Liu, Can Li, and Ngai Wong. Memristor-based adaptive analog-to-digital conversion for efficient and accurate compute-in-memory. *Nature Communications*, 16(1):9749, 2025.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mistral 7B. *arXiv preprint arXiv:2310.06825*, 2023.

Olga Kovaleva, Saurabh Kulshreshtha, Anna Rogers, and Anna Rumshisky. BERT busters: Outlier dimensions that disrupt transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3392–3405, Virtual, 2021.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In *Proc. International Conference on Learning Representations*, Virtual, 2020.

Ya Le and Xuan Yang. Tiny ImageNet visual recognition challenge. Technical report, Stanford University, 2015. CS231N course report.

Manuel Le Gallo, Riduan Khaddam-Aljameh, Milos Stanisavljevic, Athanasios Vasilopoulos, Benedikt Kersting, Martino Dazzi, Geethan Karunaratne, Matthias Brändli, Abhairaj Singh, Silvia M. Müller, Julian Büchel, Xavier Timoneda, Vinay Joshi, Malte J. Rasch, Urs Egger, Angelo Garofalo, Anastasios Petropoulos, Theodore Antonakopoulos, Kevin Brew, Samuel Choi, Injo Ok, Timothy Philip, Victor Chan, Claire Silvestre, Ishtiaq Ahsan, Nicole Saulnier, Vijay Narayanan, Pier Andrea Francese, Evangelos Eleftheriou, and Abu Sebastian. A 64-core mixed-signal in-memory compute chip based on phase-change memory for deep neural network inference. *Nature Electronics*, 6(9):680–693, 2023.

Chen Li, Elena Ferro, Corey Lammie, Manuel Le Gallo, Irem Boybat, and Bipin Rajendran. Efficient transformer adaptation for analog in-memory computing via low-rank adapters. *Neuromorphic Computing and Engineering*, 6(1):014011, 2026a.

Jindan Li, Zhaoxian Wu, Gaowen Liu, Tayfun Gokmen, and Tianyi Chen. In-memory training on analog devices with limited conductance states via multi-tile residual learning. In *Proc. International Conference on Artificial Intelligence and Statistics*, volume 300 of *Proceedings of Machine Learning Research*, pp. 3664–3672, Tangier, Morocco, 2026b.

Zhikai Li, Junrui Xiao, Lianwei Yang, and Qingyi Gu. RepQ-ViT: Scale reparameterization for post-training quantization of vision transformers. In *Proc. IEEE/CVF International Conference on Computer Vision*, pp. 17227–17236, Paris, France, 2023.

Jingyuan Liu, Jianlin Su, Xingcheng Yao, Zhejun Jiang, Guokun Lai, Yulun Du, Yidao Qin, Weixin Xu, Enzhe Lu, Junjie Yan, Yanru Chen, Huabin Zheng, Yibo Liu, Shaowei Liu, Bohong Yin, Weiran He, Han Zhu, Yuzhi Wang, Jianzhou Wang, Mengnan Dong, Zheng Zhang, Yongsheng Kang, Hao Zhang, Xinran Xu, Yutao Zhang, Yuxin Wu, Xinyu Zhou, and Zhilin Yang. Muon is scalable for LLM training. *arXiv preprint arXiv:2502.16982*, 2025.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proc. IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, Virtual, 2021.

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

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proc. International Conference on Learning Representations*, New Orleans, LA, 2019.

Matt Mahoney. About the test data. `https://mattmahoney.net/dc/textdata.html`, 2011.

S. R. Nandakumar, Manuel Le Gallo, Christophe Piveteau, Vinay Joshi, Giovanni Mariani, Irem Boybat, Geethan Karunaratne, Riduan Khaddam-Aljameh, Urs Egger, Anastasios Petropoulos, Theodore Antonakopoulos, Bipin Rajendran, Abu Sebastian, and Evangelos Eleftheriou. Mixed-precision deep learning based on computational memory. *Frontiers in Neuroscience*, 14:406, 2020.

Arild Nøkland and Lars Hiller Eidnes. Training neural networks with local error signals. In *Proc. International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 4839–4850, Long Beach, CA, 2019.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proc. International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763, Virtual, 2021.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

Malte J Rasch, Diego Moreda, Tayfun Gokmen, Manuel Le Gallo, Fabio Carta, Cindy Goldberg, Kaoutar El Maghraoui, Abu Sebastian, and Vijay Narayanan. A flexible and fast PyTorch toolkit for simulating training and inference on analog crossbar arrays. In *Proc. IEEE International Conference on Artificial Intelligence Circuits and Systems*, pp. 1–4, Washington, DC, 2021.

Malte J Rasch, Charles Mackin, Manuel Le Gallo, An Chen, Andrea Fasoli, Frédéric Odermatt, Ning Li, S R Nandakumar, Pritish Narayanan, Hsinyu Tsai, Geoffrey W Burr, Abu Sebastian, and Vijay Narayanan. Hardware-aware training for large-scale and diverse deep learning inference workloads using in-memory computing-based accelerators. *Nature Communications*, 14(1):5282, 2023.

Abu Sebastian, Manuel Le Gallo, Riduan Khaddam-Aljameh, and Evangelos Eleftheriou. Memory devices and applications for in-memory computing. *Nature Nanotechnology*, 15(7):529–544, 2020.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proc. Annual Meeting of the Association for Computational Linguistics*, pp. 1715–1725, Berlin, Germany, 2016.

Soroush Tabesh, Mher Safaryan, Andrei Panferov, Alexandra Volkova, and Dan Alistarh. CAGE: Curvature-aware gradient estimation for accurate quantization-aware training. In *Proc. Machine Learning and Systems*, volume 8, pp. 626–638, Bellevue, WA, 2026.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *Proc. International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10347–10357, Virtual, 2021.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

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

Xiuying Wei, Yunchen Zhang, Xiangguo Zhang, Ruihao Gong, Shanghang Zhang, Qi Zhang, Fengwei Yu, and Xianglong Liu. Outlier suppression: Pushing the limit of low-bit transformer language models. In *Proc. Advances in Neural Information Processing Systems*, volume 35, pp. 17402–17414, New Orleans, LA, 2022.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proc. Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Virtual, 2020.

Zhaoxian Wu, Tayfun Gokmen, Malte J. Rasch, and Tianyi Chen. Towards exact gradient-based training on analog in-memory computing. In *Proc. Advances in Neural Information Processing Systems*, volume 37, pp. 37264–37304, Vancouver, Canada, 2024.

Zhaoxian Wu, Quan Xiao, Tayfun Gokmen, Omobayode Fagbohungbe, and Tianyi Chen. Analog in-memory training on general non-ideal resistive elements: The impact of response functions. In *Proc. Advances in Neural Information Processing Systems*, volume 38, pp. 105439–105492, San Diego, CA, 2025.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. SmoothQuant: Accurate and efficient post-training quantization for large language models. In *Proc. International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 38087–38099, Honolulu, HI, 2023.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture. In *Proc. International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 10524–10533, Virtual, 2020.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Proc. Advances in Neural Information Processing Systems*, volume 32, pp. 12360–12371, Vancouver, Canada, 2019.

Bonan Zhang, Chia-Yu Chen, and Naveen Verma. Reshape and adapt for output quantization (RAOQ): Quantization-aware training for in-memory computing systems. In *Proc. International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 58739–58762, Vienna, Austria, 2024.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Supplementary Material

Table of Contents

A Experimental Setup and Statistical Reporting 16
A.1 Tasks and training 16
A.2 Analog implementation and compute 16
A.3 Evaluation and statistical reporting 16

B Architecture Survey and Related Work 17
B.1 Converter and normalization context 18
B.2 Dynamic scaling shared by digital and analog quantization 18

C Training Damage and Layer Localization 18
C.1 Endpoint replay and raw training trajectories 18
C.2 Affine effects and conversion damage across normalization layers 19

D Conversion Error and Readout Loss 22
D.1 Readout-model definitions 22
D.2 Frozen-checkpoint scale intervention 23
D.3 Why NA-∅ changes logit scale and input direction 23
D.4 Checkpoint measurements of scale and input direction 25
D.5 Absolute logit error across readout configurations 25
D.6 Derivation of the squared-margin damage estimate 25
D.7 First-order and Taylor decomposition 28
D.8 Scale-damage response across forward-ADC precision 28
D.9 Input-scaling policy robustness 29

E Supplied Gradients and Gradient Interventions 32
E.1 Loss slopes and optimizer updates 32
E.2 Checkpoint diagnostics and probe-radius robustness 33
E.3 Digital scaling and the supplied backward signal 33
E.4 Scalar digital-scale control 35
E.5 Channelwise digital-scale backward control 37
E.6 Radial gain-gradient intervention 37
E.7 Full-vector gain-gradient intervention 38
E.8 Bias-gradient intervention 39
E.9 Separating gain and bias trainability 40

F Backward Precision and Optimizer Controls 41
F.1 Backward precision and weight-update controls 41
F.2 Gradient mismatch and endpoint loss across optimizers 44
F.3 Exact-update transfer is optimizer-dependent 46

G Structural Readout Repairs 47
G.1 Post-ADC normalization removes the scale pathway 47
G.2 Post-ADC affine capacity 47
G.3 Arithmetic and parameter cost 48
G.4 Readout comparisons at a tuned head learning rate 48
G.5 Training-time gain migration 50
G.6 Architectural scope of final normalization 50

H Model, Task, and Converter Extensions 52
H.1 Width and depth 52
H.2 The 39.4M scale-up 52
H.3 Vocabulary and normalization-family controls 54
H.4 Vision tasks 55
H.5 Quantization-aware training 57
H.6 Extension to local cross-entropy training 58
H.7 Training-time output-resolution sweep 59
H.8 Output-noise details 60

I Frozen-Head Precision Replay and Post-ADC Repairs 60
I.1 Frozen-head precision replay 60
I.2 Post-ADC repair ladder 61

A Experimental Setup and Statistical Reporting

A.1 Tasks and training

The primary task is byte-level language modeling on enwik8 (Mahoney, 2011) under its standard 90M/5M/5M split. Tokens are raw bytes, so the vocabulary is the $V = 205$ distinct byte values the corpus contains. The primary model is a 4-block, width-128 Transformer with 4 attention heads, context length 64, batch size 128, 5,000 steps, and a cosine learning-rate schedule, abbreviated d128-L4 (width $d = 128$, $L = 4$ blocks; larger configurations analogously). The optimizer's learning rate rises to 3×10^{-4} over 400 warmup steps and decays by the cosine schedule to 3×10^{-5}, with $\beta = (0.9, 0.95)$, no weight decay, and gradient clipping at 1.0. Every parameter, including the analog head, shares this rate; Appendix G.4 raises the head's rate alone. Further comparisons cover a $50\times$ scale-up to 39.4M parameters (the byte-level d512-L12 model with $V = 205$, abbreviated 39M), a 32k vocabulary constructed by byte-pair encoding (BPE) (Sennrich et al., 2016) on the same corpus, and image classification on CIFAR-100 (Krizhevsky, 2009) and TinyImageNet-200 (Le & Yang, 2015) with compact vision Transformers (ViTs) (Dosovitskiy et al., 2021). A CIFAR-100 ResNet-18 (He et al., 2016) control tests the architectural scope of final normalization (Appendix G.6). Where noted, comparisons are replicated under local cross-entropy (LCE) (Nøkland & Eidnes, 2019), whose per-block readout heads share the identical readout path (Appendix H.6).

The NA-$\varnothing + \boldsymbol{\gamma}$-clamp configuration enforces $\boldsymbol{\gamma} \leq \mathbf{1}$ elementwise during training. This constraint changes optimization, not the underlying readout structure.

A.2 Analog implementation and compute

AdamW is the standard digital optimizer (Loshchilov & Hutter, 2019); *MpAdamW* accumulates its updates digitally and transfers them to the tiles as device pulses ("pulsed"), following the mixed-precision update approach of Nandakumar et al. (2020); and *exact update* denotes a hardware-aware forward with floating-point backward MVMs and exact weight updates. Analog training is pulsed unless noted otherwise. Here, "ideal backward" specifies MVM precision, not the derivative of the complete quantized forward computation; the inference backend can also apply a peripheral gradient mask (Appendix E.3 describes the mask).

Throughout the appendices, out$b$ denotes b-bit output-ADC resolution, not b-bit weights or input conversion. The training-time sweep also varies backward output precision; its input converters remain fixed, as specified in Appendix H.7. Frozen precision replays instead change only the final linear layer's forward ADC. In the primary Transformer configuration, we convert every linear layer with AIHWKit `convert_to_analog` and use `SoftBoundsReferenceDevice` with `dw_min`$=0.01$ and `up_down`$=0.1$. The main forward configuration has 6-bit input DAC and output ADC, zero read noise, `noise_management=ABS_MAX`, `bound_management=ITERATIVE`, and `out_bound`$=12$. Weight-scaling remapping is left at its AIHWKit default, off (`weight_scaling_omega`$=0$). The backward MVM reads at the same 6-bit input and output widths, so the main runs pass gradients through the converter as well; the backward-precision experiments of Appendix F.1 vary the two backward widths independently of the forward ones. Thus, `ABS_MAX` supplies per-input max-absolute scaling, while iterative bound management rescales an input whose analog output would otherwise exceed the ADC bound. LayerNorm and all explicit Q-side operators remain in digital floating point. Each stress test states its deviations from this configuration; for example, the compact-ViT placement comparisons use 6-bit forward reads and 9-bit backward reads (Appendix H.4).

Software. Experiments use Python 3.11.15, PyTorch 2.6.0 with CUDA 12.6, NumPy 2.4.6, and AIHWKit 1.1.0.

Compute environment. Experiments use a single NVIDIA H100 NVL.

A.3 Evaluation and statistical reporting

Cells are mean $\pm$ sample standard deviation (SD) unless noted otherwise, paired differences are mean $\pm$ standard error (SE), and brackets give two-sided 95% confidence intervals (CIs). Perplexity

is $\mathrm{PPL}_{\mathrm{analog}} := e^{\mathcal{L}(\tilde{z})}$ for analog evaluation and $\mathrm{PPL}_{\mathrm{ideal}} := e^{\mathcal{L}(z^\star)}$ for ideal-head replay. The readout PPL cost is $\mathrm{PPL}_{\mathrm{analog}} - \mathrm{PPL}_{\mathrm{ideal}}$ over the matched prediction items; Section 4.3 defines the corresponding additive loss penalty. Within each training study, readout configurations use the same evaluation stream, and statistical comparisons remain within that study.

Analyses selected after inspecting their outcomes are labeled exploratory. Prespecified analyses are identified where this distinction affects inference. Five-seed paired comparisons use the same seed/chip identities and a two-sided paired t statistic with four degrees of freedom ($|t| > 2.776$). Here, "ns" denotes an interval that includes zero.

The primary d128-L4 readout-fidelity, uniform-scale, input-scaling-policy, and gain-landscape replays reuse the five trained checkpoints for each readout configuration they test. Within each replay, comparisons hold the checkpoint, device realization, and evaluation batches fixed. Training interventions and extensions use the paired designs specified in their corresponding sections.

Reported intervals are nominal and per contrast. We report Holm-adjusted p values for the eight primary PPL comparisons in the 39M placement study; other p values are unadjusted.

B Architecture Survey and Related Work

Table 3 separates software prevalence from hardware exposure. We reviewed the forward paths of the PyTorch implementations in Hugging Face Transformers v5.14.1 (Wolf et al., 2020) and the official DeepSeek-V3 implementation, then checked the model papers cited below. In the table, the head is the final vocabulary or classification linear layer. LayerNorm has trainable gain and bias unless marked parameter-free; RMSNorm and RMSNorm-like layers have trainable gain without bias. A LayerNorm directly before the head gives the `NA-`$\varnothing$ software interface; the RMSNorm variants share its pre-head trainable gain. In Swin, average pooling commutes with the channelwise affine, so the trainable gain and bias still act directly before the head. OLMo-1 already uses the `N-`$\varnothing$ interface recommended here. These are software structures, with AIMC exposure conditional on mapping the head to a finite-ADC tile. ViT denotes a vision Transformer and DeiT a data-efficient image Transformer.

Table 3: Final-readout structures in representative model families.

Model families	Final-readout structure
GPT-2, OPT, BLOOM, GPT-NeoX/Pythia (Radford et al., 2019; Zhang et al., 2022; BigScience Workshop, 2024; Black et al., 2022; Biderman et al., 2023)	LayerNorm → head
BERT, RoBERTa, ALBERT (Devlin et al., 2019; Liu et al., 2019; Lan et al., 2020)	LayerNorm → vocabulary decoder
ViT/DeiT, ConvNeXt (Dosovitskiy et al., 2021; Touvron et al., 2021; Liu et al., 2022)	LayerNorm → head
Swin (Liu et al., 2021)	LayerNorm → average pooling → head
Llama, Mistral, Qwen2, Gemma, DeepSeek-V3 (Touvron et al., 2023; Jiang et al., 2023; Yang et al., 2024; Gemma Team, 2024; DeepSeek-AI, 2024)	RMSNorm → head
T5, Mamba (Raffel et al., 2020; Gu & Dao, 2024)	RMSNorm-like → head
ResNet (He et al., 2016)	Global average pooling → head
OLMo-1 (Groeneveld et al., 2024)	Parameter-free LayerNorm → head
CLIP (Radford et al., 2021)	Projection → ℓ_2 normalization

Released analog implementations do map this head. The Analog Foundation Models loader converts the full causal LM, and its default linear-module map includes the LM head (Büchel et al., 2025). Li et al. (2026a) map all MobileBERT linear layers, explicitly including the final output layer, to 8-bit-DAC/ADC AIMC tiles.

B.1 Converter and normalization context

Converter precision and its cost. Removing the pre-DAC affine avoids the error amplification it introduces without additional converter precision, which generally comes at a hardware cost (Le Gallo et al., 2023; Ali et al., 2021; Hong et al., 2025). Section 4.3 relates the resulting conversion error to readout loss under its approximation conditions.

Normalization placement and quantization. Prior work asks whether and where a normalization belongs in a digital network; we keep the final normalization and ask where its trainable affine should sit relative to the analog readout's converters. Pre-LayerNorm and Post-LayerNorm concern normalization within the residual stream (Ba et al., 2016; Xiong et al., 2020), and fine-tuning can remove all LayerNorms from GPT-2-family models at inference (Baroni et al., 2026). LayerNorm scaling factors also affect digital quantization (Li et al., 2023), and large scaling factors can emerge during pre-training (Kovaleva et al., 2021). In our setting the gain itself stays digital and unquantized; we study how it changes analog conversion error during training and the gradient feedback it receives.

B.2 Dynamic scaling shared by digital and analog quantization

The max-absolute scaling identity is also shared with digital quantization. For a fixed normalized quantizer $\mathrm{Quant}_{\mathrm{norm}}$, exact scale restoration, and $a(x) = \|x\|_\infty$, define $\mathrm{Quant}_{\mathrm{dyn}}(x) := a(x)\,\mathrm{Quant}_{\mathrm{norm}}(x/a(x))$. For nonzero x and $c > 0$, $a(cx) = ca(x)$ and $(cx)/a(cx) = x/a(x)$, so $\mathrm{Quant}_{\mathrm{dyn}}(cx) = c\,\mathrm{Quant}_{\mathrm{dyn}}(x)$. The codes are invariant, while the restored output is positively homogeneous, as in Eq. (4). This property depends on dynamic scaling and scale restoration, not on whether the computation is analog or digital; it need not hold with a fixed calibration scale. We use it to distinguish uniform scaling from channelwise direction changes and analyze the resulting loss cost.

C Training Damage and Layer Localization

C.1 Endpoint replay and raw training trajectories

Figure 2(a) uses frozen endpoints and fixed evaluation batches. Figure 2(b) uses the ideal-head training control, in which only the trunk blocks are converted to analog tiles and the head remains a digital floating-point linear layer in the forward pass, backward pass, and weight update from initialization. Because the head stays digital, it also keeps its weight tying to the token embedding. The `NA-∅` and `N-∅` controls otherwise follow the primary d128-L4 settings with five seeds and 5,000 steps, and Figure 2(b) compares their endpoint CE. Both report trainable-minus-fixed affine differences and two-sided 95% paired t_4 intervals. Pairing is within each; the two evaluation protocols are not pooled or used for a paired test between (a) and (b).

Let Δ denote the paired `NA-∅`-minus-`N-∅` difference, averaged across seeds. The share of the BP CE gap removed by ideal-head replay is $(\Delta\mathcal{L}(\tilde{z}) - \Delta\mathcal{L}(z^\star))/\Delta\mathcal{L}(\tilde{z}) = 59.9\%$, computed from mean paired CE differences rather than from mean perplexities.

Figure 6 reports the unsmoothed checkpoint histories behind Figure 1. The moving averages are applied only for visualization. Each seed is averaged in CE over a checkpoint and its available immediate neighbors, then converted to PPL where needed. Every endpoint value, confidence interval, table entry, and hypothesis test uses the unsmoothed checkpoint measurements.

All three analog configurations show an early PPL rebound, so it is not specific to the exposed readout. The early paired gap components vary in sign across seeds, whereas later checkpoints show sustained separation. The backward-precision controls in Appendix F.1 also show that the rebound depends on the training configuration.

Figure 7 extends the same raw analysis to d256-L4 over 15,000 steps. In these trajectories, the readout-penalty gap rises earlier than the ideal-head gap, and the two become comparable late in training. The exposed readout retains higher PPL, while `N-A` achieves lower mean PPL than `N-∅` at this operating point (Table 2).

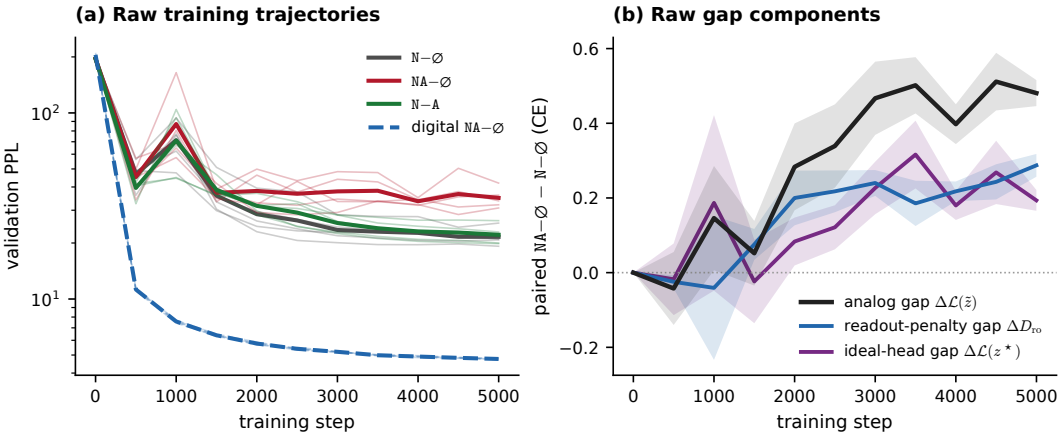


Figure 6: **Raw d128-L4 trajectories underlying the readout comparisons** (five seeds; no smoothing). Thin lines in (a) are individual seeds and thick lines are means; bands in (b) are ± SE. The two component curves sum to the total gap at each checkpoint.

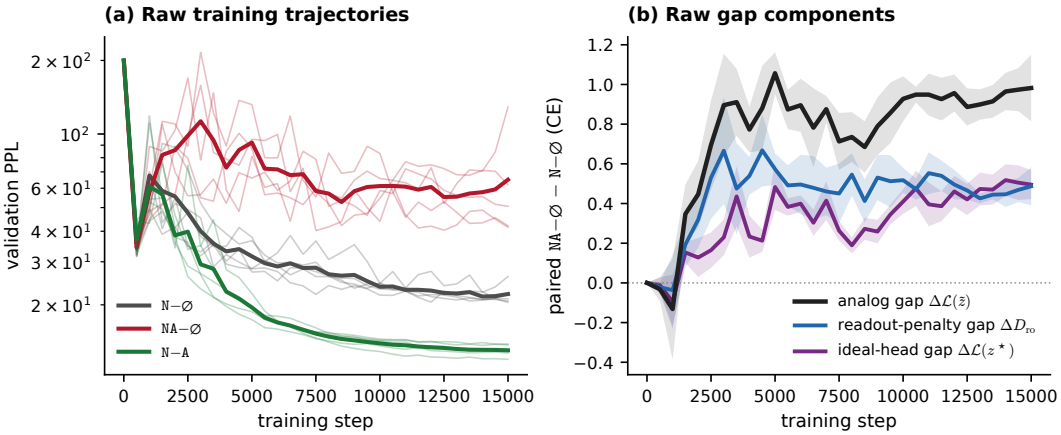


Figure 7: **Raw readout trajectories and gap components at d256-L4** over 15,000 steps (five seeds; no smoothing). Thin lines in (a) are individual seeds and thick lines are means; bands in (b) are ± SE.

C.2 Affine effects and conversion damage across normalization layers

Pulsed updates and 6-bit backward. Among the three normalization groups, training the final-readout affine causes the strongest affine-related damage. Figure 2(c) compares three groups of LayerNorm affines: the final readout (F), all four attention pre-normalizations (A), and all four MLP pre-normalizations (M). Each group is either trainable or fixed at $\gamma = 1, \beta = 0$, giving eight combinations. The architecture is identical across combinations; only the selected affine parameters are excluded from optimization. All other parameters remain trainable. Every trainable affine tensor receives nonzero gradients and changes during training, while all frozen affine parameters remain unchanged.

Each combination uses the enwik8 d128-L4 model and learning-rate schedule in Appendix A, with zero dropout, seeds 0–4, and 5,000 training steps. The 17 mapped linear layers use 6-bit input and output converters in both forward and backward MVMs. MpAdamW transfers device pulses to the SoftBoundsReference device with $\Delta w_{\min} = 0.01$ and asymmetry 0.1. Input and read noise are zero. Attention score and value products, embeddings, normalization, and activations remain digital.

Within each seed, the eight runs share the initial model and chip realization, initial CE, and training batches. Evaluation uses the same 16 fixed batches for every run. At each frozen endpoint, exact multiplication with unchanged weights replaces only the layers immediately receiving a group's

normalized activations: the final head for F, the four QKV projections for A, and the four first MLP matrices for M. The attention and MLP output projections remain analog. A group's *local conversion cost* is full analog CE minus CE after this replacement. These costs can be negative and are not additive shares of total network damage.

For each group and seed, we average the four trainable-minus-fixed contrasts over the other two groups' settings. We also compare the readout effect directly with each internal group's effect within seed. The seed is the replication unit; all intervals are two-sided 95% paired t_4 intervals without multiplicity adjustment. Table 5 reports all eight endpoints, and Tables 4 and 6 report the averaged and conditional effects.

Training the readout affine increases both task CE and local conversion cost, with effects larger than those of attention or MLP in direct paired comparisons. Both readout effects remain positive in all four attention/MLP backgrounds, with each interval excluding zero. The averaged internal-group effects are not distinguished from zero, although individual backgrounds show increases in the attention conversion cost or in MLP task CE and conversion cost, so these results do not imply that internal normalization affines have no effect.

Table 4: **Normalization location under pulsed updates and 6-bit backward.** Effects and direct location differences, with 95% paired intervals.

Location or comparison	Task CE effect	Local conversion-cost effect
Readout	$+0.4427\,[+0.3373, +0.5480]$	$+0.2853\,[+0.2396, +0.3309]$
Attention	$-0.0035\,[-0.0813, +0.0743]$	$-0.0007\,[-0.0097, +0.0082]$
MLP	$+0.0282\,[-0.0770, +0.1333]$	$+0.0111\,[-0.0095, +0.0316]$
Readout minus attention	$+0.4461\,[+0.2751, +0.6172]$	$+0.2860\,[+0.2404, +0.3317]$
Readout minus MLP	$+0.4145\,[+0.2068, +0.6222]$	$+0.2742\,[+0.2099, +0.3385]$

Table 5: **Endpoint losses under pulsed updates and 6-bit backward.** Mean $\pm$ SD over five seeds; F, A, and M denote readout, attention, and MLP normalization groups.

F	A	M	Task CE	F cost	A cost	M cost
Fixed	Fixed	Fixed	3.082 ± 0.023	0.186 ± 0.015	0.003 ± 0.004	-0.009 ± 0.007
Fixed	Fixed	Trainable	3.145 ± 0.057	0.173 ± 0.027	-0.004 ± 0.009	0.005 ± 0.007
Fixed	Trainable	Fixed	3.140 ± 0.070	0.184 ± 0.036	0.001 ± 0.003	-0.010 ± 0.009
Fixed	Trainable	Trainable	3.200 ± 0.236	0.175 ± 0.046	-0.001 ± 0.005	-0.010 ± 0.040
Trainable	Fixed	Fixed	3.619 ± 0.114	0.513 ± 0.067	-0.004 ± 0.013	-0.013 ± 0.020
Trainable	Fixed	Trainable	3.613 ± 0.153	0.495 ± 0.109	-0.002 ± 0.008	-0.006 ± 0.047
Trainable	Trainable	Fixed	3.555 ± 0.061	0.437 ± 0.078	0.006 ± 0.011	-0.013 ± 0.021
Trainable	Trainable	Trainable	3.550 ± 0.047	0.414 ± 0.065	-0.017 ± 0.016	0.010 ± 0.016

Table 6: **Conditional affine effects under pulsed updates and 6-bit backward.** Trainable-minus-fixed differences with 95% paired intervals.

Changed group	Other groups	Task CE effect	Local cost effect
Readout	A fixed, M fixed	$+0.5374\,[+0.3938, +0.6810]$	$+0.3270\,[+0.2549, +0.3991]$
Readout	A fixed, M trainable	$+0.4683\,[+0.2747, +0.6619]$	$+0.3218\,[+0.1953, +0.4484]$
Readout	A trainable, M fixed	$+0.4144\,[+0.2921, +0.5366]$	$+0.2531\,[+0.1488, +0.3573]$
Readout	A trainable, M trainable	$+0.3506\,[+0.0287, +0.6724]$	$+0.2393\,[+0.1319, +0.3467]$
Attention	F fixed, M fixed	$+0.0587\,[-0.0372, +0.1546]$	$-0.0016\,[-0.0061, +0.0029]$
Attention	F fixed, M trainable	$+0.0547\,[-0.1895, +0.2989]$	$+0.0032\,[-0.0102, +0.0165]$
Attention	F trainable, M fixed	$-0.0643\,[-0.2532, +0.1246]$	$+0.0103\,[+0.0021, +0.0185]$
Attention	F trainable, M trainable	$-0.0630\,[-0.3077, +0.1816]$	$-0.0148\,[-0.0380, +0.0083]$
MLP	F fixed, A fixed	$+0.0634\,[+0.0121, +0.1146]$	$+0.0138\,[+0.0018, +0.0257]$
MLP	F fixed, A trainable	$+0.0594\,[-0.2781, +0.3969]$	$+0.0004\,[-0.0493, +0.0501]$
MLP	F trainable, A fixed	$-0.0057\,[-0.1663, +0.1549]$	$+0.0071\,[-0.0323, +0.0465]$
MLP	F trainable, A trainable	$-0.0044\,[-0.1010, +0.0921]$	$+0.0230\,[-0.0089, +0.0549]$

Loss remaining after exact-multiplication replay. Table 7 reports the readout trainable-minus-fixed CE contrast after idealizing either the final head or all F/A/M consumers jointly. Each contrast is averaged over the four attention/MLP settings within seed. With pulsed updates and 6-bit backward, a positive CE difference remains under both replay choices. Thus, removing final-head conversion error does not remove all of the damage associated with training its affine. Joint replay gives the same qualitative result, without identifying which learned parameters retain that damage. The floating-point-backward, exact-update configuration detects no remaining CE difference under either replay choice. The configurations differ in both backward precision and weight updates, so this comparison does not isolate either factor.

Table 7: **Readout-affine CE effects after exact-multiplication replay.** Trainable-minus-fixed differences with 95% paired intervals.

Training configuration	Exact multiplication at	Remaining CE effect
Pulsed updates, 6-bit backward	Final head	$+0.1574\,[+0.0676, +0.2472]$
Pulsed updates, 6-bit backward	F/A/M consumers	$+0.1529\,[+0.0636, +0.2422]$
Exact updates, floating-point backward	Final head	$-0.0093\,[-0.0387, +0.0201]$
Exact updates, floating-point backward	F/A/M consumers	$-0.0088\,[-0.0386, +0.0210]$

Floating-point backward and exact updates. With floating-point backward and exact updates, training the readout affine still increases its local conversion cost, but no normalization group has a detected effect on analog CE. We also train all eight affine settings with floating-point backward MVMs and exact AdamW updates, retaining d128-L4, 6-bit forward reads, seeds 0–4, and the same training and evaluation schedules. The affine interventions and local conversion-cost measurements follow the protocol above. Effects are paired within seed, with unadjusted two-sided 95% t_4 intervals. Table 8 reports averaged effects and the attention-by-MLP interaction. The interaction is a difference-of-differences, averaged over the readout settings; its conversion-cost effect uses exact replay of all three consumer groups jointly.

Table 8: **Normalization-group effects with floating-point backward and exact updates** ($n = 5$ paired seeds). Mean effects with two-sided 95% paired intervals.

effect	analog CE Δ	all-groups-idealized CE Δ	local conversion cost Δ
final readout	0.02306 [−0.00486, 0.05098]	−0.00882 [−0.03864, 0.02099]	0.03238 [0.03053, 0.03423]
attention group	−0.01405 [−0.03364, 0.00553]	−0.01799 [−0.03906, 0.00307]	0.00454 [0.00357, 0.00551]
MLP group	0.00123 [−0.00689, 0.00935]	−0.00089 [−0.00972, 0.00793]	0.00265 [−0.00004, 0.00534]
attention × MLP	−0.04040 [−0.06401, −0.01679]	−0.04375 [−0.07089, −0.01661]	0.00335 [−0.00316, 0.00985]

Figure 8 resolves the conditional effects. Training the final-readout affine increases its local conversion cost in all four attention/MLP backgrounds, with every interval excluding zero. The attention group also increases its local conversion cost in all four readout/MLP backgrounds. The MLP-group effect is weaker and context-dependent, since its local conversion-cost interval excludes zero only in the two backgrounds with the attention group trainable. None of the three normalization-group averaged analog-CE main-effect intervals, nor any of the 12 conditional analog-CE intervals, excludes zero. The detected negative attention-by-MLP analog interaction is almost entirely mirrored in all-groups-idealized CE, while its conversion-cost interval crosses zero.

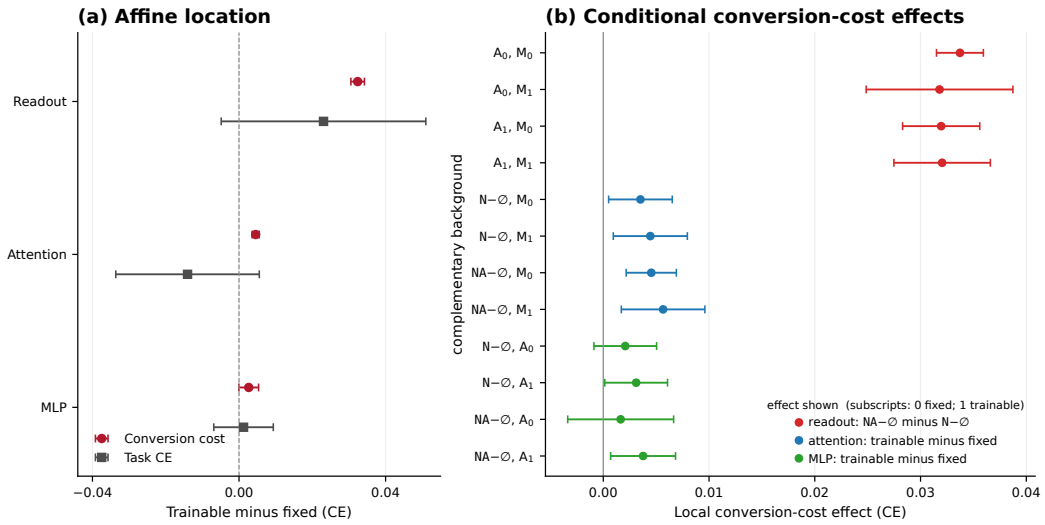


Figure 8: **Affine effects with floating-point backward and exact updates.** (a) Averaged local conversion-cost and task-CE effects. (b) Conditional local conversion-cost effects, with paired 95% intervals. F, A, and M denote readout, attention, and MLP; 0 denotes fixed and 1 trainable.

D Conversion Error and Readout Loss

D.1 Readout-model definitions

This subsection expands the operators and metrics introduced in Sections 3.1 and 4 and states the tiled-computation assumptions needed for the scale argument. For one LM token, or one pooled vision example, let $h \in \mathbb{R}^d$ be the hidden state produced by the trunk, that is, by all layers below the readout. The parameter-free centered LayerNorm over the d feature channels is

$$\mu_h := \frac{1}{d}\sum_{j=1}^{d} h_j, \qquad s_h^2 := \frac{1}{d}\sum_{j=1}^{d}(h_j - \mu_h)^2, \qquad \hat{h}_j := \frac{h_j - \mu_h}{\sqrt{s_h^2 + \varepsilon_{\mathrm{LN}}}}, \quad \varepsilon_{\mathrm{LN}} = 10^{-5}.$$

Here, the subscript LN denotes LayerNorm. Statistics are computed independently per token or pooled example, so $\hat{h}$ carries normalization only and no trainable affine. For $P = \texttt{NA}$, the trainable affine parameters are the gain $\boldsymbol{\gamma} \in \mathbb{R}^d$ and bias $\boldsymbol{\beta} \in \mathbb{R}^d$. For $P = \texttt{N}$, these are fixed to $\mathbf{1}$ and $\mathbf{0}$. In experiments explicitly labeled RMSNorm, normalization instead omits centering and divides by $\sqrt{d^{-1}\sum_j h_j^2 + \varepsilon_{\mathrm{LN}}}$. There, N and NA denote RMSNorm with fixed and trainable gain, respectively, and A is a trainable gain without bias. Under LCE the same definitions apply separately to each block's local readout state.

For an affine-only operator, we write $\texttt{A}_{\boldsymbol{s},\boldsymbol{u}}(z) = \boldsymbol{s} \odot z + \boldsymbol{u}$; $\varnothing$ denotes no operation. In the parameter-free readout control, *fixed* means $\boldsymbol{\gamma} = \mathbf{1}, \boldsymbol{\beta} = \mathbf{0}$; in checkpoint replay, parameters are instead held at their learned values. Writing $h_P = P(h)$, its per-input RMS is $s(h_P) = \mathrm{RMS}(h_P) = \|h_P\|_2/\sqrt{d}$. In the training diagnostics, h_{rms} is the arithmetic mean of the input RMS values computed within each diagnostic batch; each batch RMS pools its prediction items and hidden coordinates. Frozen-checkpoint replays instead compute input RMS after pooling all evaluation items and hidden coordinates.

To define scale over an evaluation set, index tokens or pooled examples by n, hidden coordinates by $j \in \{1, \ldots, d\}$, and output coordinates by $v \in \{1, \ldots, V\}$. We use the ordinary RMS formula from Section 4; Eq. (3) pools all evaluation items and logit coordinates. Gain and bias RMS values pool their parameter coordinates, $\gamma_{\mathrm{rms}} := \mathrm{RMS}(\boldsymbol{\gamma})$ and $\beta_{\mathrm{rms}} := \mathrm{RMS}(\boldsymbol{\beta})$. These diagnostics are computed per trained run, and configuration-level values reported in the tables summarize those run-level measurements across seeds. Where configurations must be distinguished, we write $z_{\mathrm{rms}}(M)$ for the logit scale evaluated on configuration M.

Uniform-scale invariance. Under per-input max-absolute scaling with fixed head weights, a bias-free head, and zero read noise, rescaling a nonzero pre-DAC vector by any $c > 0$ leaves the normalized DAC input, the analog product, and the returned ADC codes unchanged. The restored output and its conversion error scale by c, yielding Eq. (4) whenever the relative error is defined. With $a(h_P) = \|h_P\|_\infty$, Eq. (2) gives

$$a(ch_P) = c\,a(h_P), \qquad \frac{ch_P}{a(ch_P)} = \frac{h_P}{a(h_P)}, \qquad \tilde{z}(ch_P) = c\,\tilde{z}(h_P). \tag{14}$$

The converter-facing input stays fixed, but the restored output does not. Since the ideal product also scales by c, their difference satisfies $\epsilon(ch_P) = c\,\epsilon(h_P)$.

Scope of the scale identity. The scale factor must track the current input. This holds for per-input max-absolute scaling, including when iterative bound management is active. It need not hold for a calibrated static range or an average of earlier maxima. A channelwise affine can also change input direction, so Eq. (4) does not imply invariance under changes in individual gains or biases.

D.2 Frozen-checkpoint scale intervention

The scale-sweep variable c is an evaluation-only digital multiplier applied to a frozen BP N-∅ model immediately before the final head's DAC:

$$\text{trunk} \to \texttt{N} \to \boxed{h_P \mapsto c\,h_P} \to \text{DAC} \to \text{RPU matmul} \to \text{ADC} \to \text{logits}.$$

The trunk, head weights, device realization, and evaluation batches remain fixed across scales. Both analog and ideal-head replay receive the same scaled representation, for each token in language models or pooled example in vision models. All c sweeps use this final-head intervention. For N-∅, it is equivalent to temporarily setting $\boldsymbol{\gamma} = c\mathbf{1}$; it does not reproduce a trained NA-∅ model's channelwise gain, bias, or co-adapted weights. Table 9 reports the resulting scale sweep.

Table 9: **Causal scale intervention on trained BP N-∅ checkpoints.** Readout PPL cost is mean ± SE over five checkpoint/device units.

c	δ	analog ppl	ideal ppl	readout PPL cost
0.50	0.303	28.28	27.06	$+1.22 \pm 0.20$
1.00	0.606	21.64	18.47	$+3.17 \pm 0.46$
1.25	0.757	24.14	19.24	$+4.90 \pm 0.78$
1.35	0.818	25.84	20.03	$+5.81 \pm 0.95$
1.50	0.909	29.16	21.66	$+7.49 \pm 1.28$
2.00	1.212	48.69	31.60	$+17.09 \pm 3.11$
2.50	1.515	88.99	51.44	$+37.56 \pm 6.91$
3.00	1.817	168.79	89.13	$+79.67 \pm 14.59$

D.3 Why NA-∅ changes logit scale and input direction

The configurations N-∅ and NA-∅ differ in whether the P-side affine is trainable. For example n, write the v-th ideal logit as

$$z^\star_{n,v} = \sum_j W_{vj}\gamma_j \hat{h}_{n,j} + \sum_j W_{vj}\beta_j,$$

with $\mu_v := \mathbb{E}_n[z^\star_{n,v}]$ and $\Sigma_{jk} := \mathrm{Cov}_n(\hat{h}_{n,j}, \hat{h}_{n,k})$. The squared RMS equals the average variance plus squared mean across output coordinates:

$$z^2_{\mathrm{rms}} = \frac{1}{V}\sum_{v=1}^{V}\sum_{j,k=1}^{d} W_{vj}W_{vk}\gamma_j\gamma_k\Sigma_{jk} + \frac{1}{V}\sum_{v=1}^{V}\mu_v^2. \tag{15}$$

LayerNorm centering does not eliminate the covariance terms. Channelwise gain affects their weighted sum, so increasing an individual gain need not increase the total logit scale. The bias contributes through the squared means, since its offset is constant across examples. Together with changes in the trunk and W, these dependencies prevent γ_{rms} alone from determining z_{rms}, which we therefore measure directly.

Peak-to-RMS statistic. For evaluated token-row head inputs $h_P^{(n)} \in \mathbb{R}^d$, the peak-to-RMS ratio and its mean are

$$r_{\mathrm{pk}}\left(h_P^{(n)}\right) = \frac{\left\|h_P^{(n)}\right\|_\infty}{\left\|h_P^{(n)}\right\|_2/\sqrt{d}} = \frac{\max_j \left|h_{P,j}^{(n)}\right|}{\sqrt{d^{-1}\sum_{j=1}^d \left(h_{P,j}^{(n)}\right)^2}}, \qquad \overline{r}_{\mathrm{pk}} := \mathbb{E}_n\left[r_{\mathrm{pk}}\left(h_P^{(n)}\right)\right]. \tag{16}$$

Here, n indexes the N token rows taken over all evaluated batches and sequence positions. We report the mean rowwise ratio $\overline{r}_{\mathrm{pk}}$.

How learned channelwise gain changes peak-to-RMS. For row n, the learned affine gives $h_P^{(n)} = \boldsymbol{\gamma} \odot \hat{h}^{(n)} + \boldsymbol{\beta}$, and hence

$$r_{\mathrm{pk}}\left(h_P^{(n)}\right) = \frac{\max_j |\gamma_j \hat{h}_j^{(n)} + \beta_j|}{\sqrt{d^{-1}\sum_j (\gamma_j \hat{h}_j^{(n)} + \beta_j)^2}}. \tag{17}$$

If $\boldsymbol{\beta} = 0$ and $\boldsymbol{\gamma} = c\mathbf{1}$, the common c cancels. A learned $\boldsymbol{\gamma}$ is instead channelwise. With $\boldsymbol{\beta} = 0$ and current peak index k, while k remains the peak, $r_{\mathrm{pk}}^2 = d\gamma_k^2(\hat{h}_k^{(n)})^2 / \sum_j \gamma_j^2(\hat{h}_j^{(n)})^2$. Increasing the peak gain relative to gains on other active coordinates raises the ratio. Increasing a nonpeak gain can instead enlarge the denominator without changing the peak. The bias and co-adapted trunk can reinforce or oppose these effects, so a larger γ_{rms} alone does not imply a larger peak-to-RMS.

Why peak-to-RMS controls effective resolution. Because the maximum sets the shared DAC range, a larger peak-to-RMS coarsens the input grid relative to the row's RMS. If the normalized DAC spacing is δ_{DAC}, its spacing in the original input units is $a\delta_{\mathrm{DAC}}$. Relative to the input RMS, this is

$$\frac{a\delta_{\mathrm{DAC}}}{s(h_P)} = r_{\mathrm{pk}}(h_P)\,\delta_{\mathrm{DAC}}. \tag{18}$$

For a b-bit DAC, let $a_n = \|h_P^{(n)}\|_\infty$ and $s_n = \|h_P^{(n)}\|_2/\sqrt{d}$. Our AIHWKit convention represents a b-bit normalized DAC step as $2/q_b$, where $q_b := 2^b - 2$, so the resolution parameter $1/q_b$ is relative to the full symmetric range $[-1, 1]$. Define the componentwise normalized quantizer $\mathrm{Quant}_b(u) := (2/q_b)\,\mathrm{round}(q_b u/2)$. Let $h_{P,q}^{(n)}$ denote the scaled-and-restored DAC-quantized version of $h_P^{(n)}$; the lowercase subscript q marks quantization and is unrelated to the post-ADC interface operator Q. Before any additional bound-management rescaling, $h_{P,q}^{(n)}$, its error, and its nominal step are

$$h_{P,q}^{(n)} := a_n\,\mathrm{Quant}_b\left(\frac{h_P^{(n)}}{a_n}\right), \qquad e_h^{(n)} := h_{P,q}^{(n)} - h_P^{(n)}, \qquad \Delta_{h,n} := \frac{2a_n}{q_b}. \tag{19}$$

Under the usual unsaturated high-resolution approximation, in which quantization error is spread within a bin,

$$\frac{\mathrm{RMS}(e_h^{(n)})}{s_n} \simeq \frac{\Delta_{h,n}}{\sqrt{12}\,s_n} = \frac{2r_{\mathrm{pk}}(h_P^{(n)})}{\sqrt{12}\,q_b}, \qquad N_{\mathrm{RMS}}^{(n)} := \frac{s_n}{\Delta_{h,n}} = \frac{q_b}{2r_{\mathrm{pk}}(h_P^{(n)})}. \tag{20}$$

Here, RMS is computed over the hidden coordinates of item n. The count $N_{\mathrm{RMS}}^{(n)}$ gives the nominal DAC intervals spanning one input RMS. Uniform scaling preserves this count; channelwise gain and bias can change it.

From input resolution to relative corruption. Equation (20) describes input resolution. A proportional relation $\rho \propto r_{\mathrm{pk}}$ would require further assumptions about how input error propagates. At the full readout, ϵ contains the projected input error $We_h^{(n)}$ together with output-conversion error, and the programmed matrix, activation distribution, bound-management response, and error direction decide how much of the input-space resolution loss appears in logit space. We therefore measure the link. The following checkpoint comparisons measure peak-to-RMS and ρ, then test whether matching logit scale removes the difference in conversion damage.

D.4 CHECKPOINT MEASUREMENTS OF SCALE AND INPUT DIRECTION

Figure 3 plots five-seed means as bars and joins the two configurations within each seed. The identity $\delta = \rho z_{\mathrm{rms}}$ holds within each seed, so the plotted means need not multiply exactly. Each diagnostic is higher in NA-∅ in all five paired seeds.

Matching logit scale. Since the head is bias-free, scaling a frozen N-∅ input multiplies its ideal logits and their RMS by c. Within paired seed/chip unit s, we match the trained NA-∅ model's ideal-logit RMS using

$$c_{\mathrm{eff}}^{(s)} := \frac{z_{\mathrm{rms}}(\texttt{NA-}\varnothing, s)}{z_{\mathrm{rms}}(\texttt{N-}\varnothing, s)}. \tag{21}$$

Table 10 gives these factors and the residual differences. The wider scale sweeps use their rounded mean, $c_{\mathrm{rep}} = 1.35$, as a common plotting anchor. At the matched scale, the trained NA-∅ models still have greater absolute logit perturbation and readout CE penalty, with both paired intervals above zero. Uniform scaling alone therefore does not explain the gap. Input direction, trunk, and head differ between the trained models, so their contributions remain combined in the residual.

Table 10: **Seed-specific scale matching and residual readout damage.** The top panel lists the factors defined in Eq. (21). The bottom panel summarizes these factors and the paired residuals, trained NA-∅ minus frozen N-∅ replayed at its matched scale. All measurements use the same sixteen fixed evaluation batches. Intervals are unadjusted two-sided 95% t_4 intervals over five seeds.

seed 0	seed 1	seed 2	seed 3	seed 4
1.1188	1.6302	1.3698	1.2632	1.3468

Quantity	Mean	SE	95% CI
Scale factor c_{eff}	1.3458	0.0836	$[1.1136, 1.5779]$
Residual absolute perturbation δ	$+0.3393$	0.0804	$[0.1161, 0.5625]$
Residual readout CE penalty D_{ro}	$+0.1971$	0.0337	$[0.1035, 0.2908]$

D.5 ABSOLUTE LOGIT ERROR ACROSS READOUT CONFIGURATIONS

Table 11 lists the three fidelity diagnostics next to the endpoint perplexity for six readout variants spanning low to high δ, and Figure 9 plots their association. Larger δ generally accompanies higher analog PPL in these configurations. This is a descriptive comparison across readout structures and training rules, rather than an isolated test of the loss approximation.

Table 11: **Logit-error diagnostics across readout variants** (five frozen checkpoint/device units). Fidelity columns are measured on the same sixteen fixed evaluation batches as the checkpoint-paired measurements of Section 4.2, and analog perplexity is replayed on those same batches.

variant	δ	ρ	z_{rms}	analog ppl
BP N-∅	0.606	0.139	4.367	21.64
LCE N-∅	0.616	0.157	4.045	23.64
BP N-A	0.618	0.177	3.579	22.29
BP NA-A	1.032	0.201	5.327	26.40
LCE NA-∅	1.090	0.181	6.160	32.99
BP NA-∅	1.158	0.200	5.878	34.96

D.6 DERIVATION OF THE SQUARED-MARGIN DAMAGE ESTIMATE

The coefficient in Eq. (5) follows from the ideal predictions without fitting. These predictions change under rescaling, so D_{ro} need not grow by c^2.

Empirical losses and margin expansion. We first consider a bare ADC output, $Q = \varnothing$, so the logits in Eq. (2) enter the loss directly. Matched evaluation holds $\mathcal{E}$, the trunk states, and the

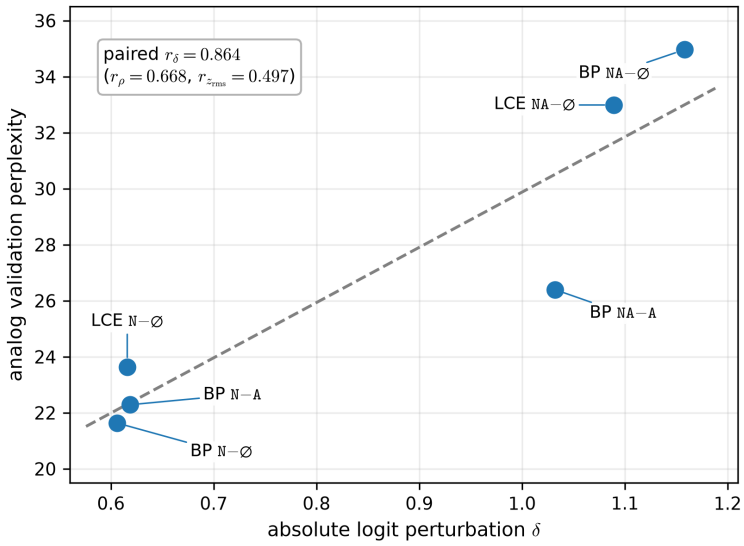


Figure 9: Absolute logit perturbation tracks analog perplexity across readout variants. Points are per-variant 5-seed means, and the dashed line is a least-squares fit. The displayed correlations are averaged over the five within-seed comparisons.

programmed head weights fixed. With $Q = \varnothing$, the single-item loss is $\ell(z, y) = -\log \mathrm{softmax}(z)_y$. As in the main text, $\mathbb{E}_n$ denotes the uniform empirical average over the N prediction items in $\mathcal{E}$. The two mean losses and their itemwise difference satisfy

$$\begin{aligned} \mathcal{L}(z^\star) &= \mathbb{E}_n[\ell(z_n^\star, y_n)], & \mathcal{L}(\tilde{z}) &= \mathbb{E}_n[\ell(\tilde{z}_n, y_n)], \\ \Delta\,\mathrm{CE}_n &:= \ell(\tilde{z}_n, y_n) - \ell(z_n^\star, y_n), & D_{\mathrm{ro}} &= \mathcal{L}(\tilde{z}) - \mathcal{L}(z^\star) = \mathbb{E}_n[\Delta\,\mathrm{CE}_n]. \end{aligned} \tag{22}$$

Run-level statistics are subsequently aggregated across seeds.

Cross-entropy responds to errors in class margins, not to every logit displacement equally. A common shift leaves all probabilities unchanged; perturbing one class relative to another does not. For item n, let $p_n = \mathrm{softmax}(z_n^\star)$ and let e_{y_n} be the one-hot target. The Taylor expansion makes this distinction explicit:

$$\Delta\,\mathrm{CE}_n = (p_n - e_{y_n})^\top \epsilon_n + \frac{1}{2} \sum_{i<j} p_{n,i} p_{n,j} (\epsilon_{n,i} - \epsilon_{n,j})^2 + R_n, \tag{23}$$

where R_n is the third-order remainder. The first term is task-aligned and can have either sign. The second is nonnegative and depends only on errors in pairwise margins. It equals $\frac{1}{2}\epsilon_n^\top H_n \epsilon_n$, with $H_n := \mathrm{diag}(p_n) - p_n p_n^\top$ the softmax Hessian. The formal statement below makes explicit the correction terms omitted by the informal Proposition 1 in Section 4.3.

Proposition 2 (Squared-margin estimate of readout damage, formal)**.** *For $Q = \varnothing$ under the matched evaluation above, let $T_1 := \mathbb{E}_n[(p_n - e_{y_n})^\top \epsilon_n]$ be the first-order task projection, $T_{2,\mathrm{exact}} := \frac{1}{2}\mathbb{E}_n[\mathrm{Var}_{v\sim p_n}(\epsilon_{n,v})]$ the exact second-order term, and $T_{2,\mathrm{global}} := \frac{1}{2}\delta^2\,\mathbb{E}_n[1 - \|p_n\|^2]$ its scalar estimate. With $R_{\geq 3} := \mathbb{E}_n[R_n]$ the mean Taylor remainder in Eq. (23),*

$$D_{\mathrm{ro}} = T_{2,\mathrm{global}} + T_1 + (T_{2,\mathrm{exact}} - T_{2,\mathrm{global}}) + R_{\geq 3}. \tag{24}$$

The error of the squared-margin estimate in Eq. (5) is therefore exactly $T_1 + (T_{2,\mathrm{exact}} - T_{2,\mathrm{global}}) + R_{\geq 3}$. The estimate is accurate when these correction terms are small relative to $T_{2,\mathrm{global}}$.

Proof. Average Eq. (23) over evaluation items. Its quadratic term equals $\frac{1}{2}\,\mathrm{Var}_{v\sim p_n}(\epsilon_{n,v})$, so the average loss difference is $T_1 + T_{2,\mathrm{exact}} + R_{\geq 3}$. Adding and subtracting $T_{2,\mathrm{global}}$ gives Eq. (24). □

For finite errors, the same decomposition can be evaluated directly from the logits:

$$D_{\mathrm{ro}} = T_1 + \mathbb{E}_n\left[\log\left(\sum_v p_{n,v} e^{\epsilon_{n,v}}\right) - p_n^\top \epsilon_n\right]. \tag{25}$$

The bracketed term is nonnegative by Jensen's inequality. The quadratic term and remainder can be evaluated without constructing the Hessian:

$$\begin{aligned} T_{2,\mathrm{exact}} &= \tfrac{1}{2}\mathbb{E}_n\left[\sum_v p_{n,v}\epsilon_{n,v}^2 - \left(\sum_v p_{n,v}\epsilon_{n,v}\right)^2\right], \\ R_{\geq 3} &= D_{\mathrm{ro}} - T_1 - T_{2,\mathrm{exact}}. \end{aligned} \tag{26}$$

Moment conditions. The first aggregate condition, *task-orthogonality*, requires the mean first-order projection to be small relative to the quadratic term:

$$\left|\mathbb{E}_n[(p_n - e_{y_n})^\top \epsilon_n]\right| \;\ll\; \tfrac{1}{2}\mathbb{E}_n[\epsilon_n^\top H_n \epsilon_n]. \tag{27}$$

This is a separate condition on error alignment with the task gradient, not a consequence of small error magnitude. With ideal logits and error directions fixed, rescaling every error by $t > 0$ gives $T_1(t) = tT_1(1)$ and $T_{2,\mathrm{exact}}(t) = t^2 T_{2,\mathrm{exact}}(1)$. If $T_1(1) \neq 0$ and $T_{2,\mathrm{exact}}(1) > 0$, their magnitude ratio grows as $1/t$. Small errors control the higher-order remainder but do not by themselves make the linear term negligible. Task-orthogonality concerns the aggregate projection; it does not require each item's error to be orthogonal to its task gradient.

Common shifts are absent from both the exact loss change and its first two Taylor terms. Define the class-mean error $m_n := V^{-1}\mathbf{1}^\top \epsilon_n$ and the centered error $\epsilon_n^\perp := \epsilon_n - m_n\mathbf{1}$. Softmax shift invariance gives

$$\ell(z_n^\star + \epsilon_n, y_n) - \ell(z_n^\star, y_n) = \ell(z_n^\star + \epsilon_n^\perp, y_n) - \ell(z_n^\star, y_n).$$

Also, $(p_n - e_{y_n})^\top \mathbf{1} = 0$ and $H_n\mathbf{1} = 0$, so replacing ϵ_n by $\epsilon_n^\perp$ leaves both T_1 and $T_{2,\mathrm{exact}}$ unchanged. The uncentered RMS nevertheless includes the common component. With $\delta_c^2 := (NV)^{-1}\sum_{n=1}^N \|\epsilon_n^\perp\|^2$, its exact decomposition is

$$\delta^2 = \delta_c^2 + \mathbb{E}_n[m_n^2].$$

Thus, additional common-shift energy can increase δ^2 without changing the loss. Equation (3) retains $\delta^2 = (NV)^{-1}\sum_{n=1}^N \|\epsilon_n\|^2$. Reducing the Hessian-weighted term to this scalar RMS requires *curvature-neutrality*:

$$\mathbb{E}_n[\epsilon_n^\top H_n \epsilon_n] \;\approx\; \delta^2\,\mathbb{E}_n[\mathrm{tr}(H_n)] = \delta^2\,\mathbb{E}_n[1 - \|p_n\|^2]. \tag{28}$$

This approximation depends on error orientation relative to H_n and on how error energy varies with $\mathrm{tr}(H_n)$ across prediction items.

An isotropic second-moment model explains the uncentered coefficient without requiring negligible common-shift energy. For this illustrative model, let $\mathbb{E}_\epsilon$ average over a hypothetical error distribution at fixed ideal logits and labels, and suppose $\mathbb{E}_\epsilon[\epsilon_n\epsilon_n^\top] = \sigma^2 I$ with the same σ^2 for every item. Then

$$\begin{aligned} \mathbb{E}_{n,\epsilon}[\|\epsilon_n\|^2/V] &= \sigma^2, \\ \tfrac{1}{2}\mathbb{E}_{n,\epsilon}[\epsilon_n^\top H_n \epsilon_n] &= \tfrac{1}{2}\sigma^2\mathbb{E}_n[\mathrm{tr}(H_n)], \\ \mathbb{E}_{n,\epsilon}[m_n^2] &= \sigma^2/V. \end{aligned}$$

The population second-order term therefore matches the uncentered scalar estimate even when the common component accounts for $1/V$ of the error energy. Conditional zero mean, $\mathbb{E}_\epsilon[\epsilon_n] = 0$, would additionally make the population first-order term zero; the second-moment condition alone does not imply this.

Centering does not establish task-orthogonality or curvature-neutrality. Even under isotropy in the $(V-1)$-dimensional centered subspace with equal error energy across items, using centered RMS would require an additional coefficient $V/(V-1)$ in the population quadratic estimate. We retain the uncentered empirical RMS. The isotropic model above illustrates sufficient moment conditions without asserting a distribution for deterministic converter errors. Together with a small remainder in Eq. (23), Eqs. (27) and (28) give Eq. (5). Appendix D.7 tests the Taylor remainder and both aggregate moment conditions on the frozen replay.

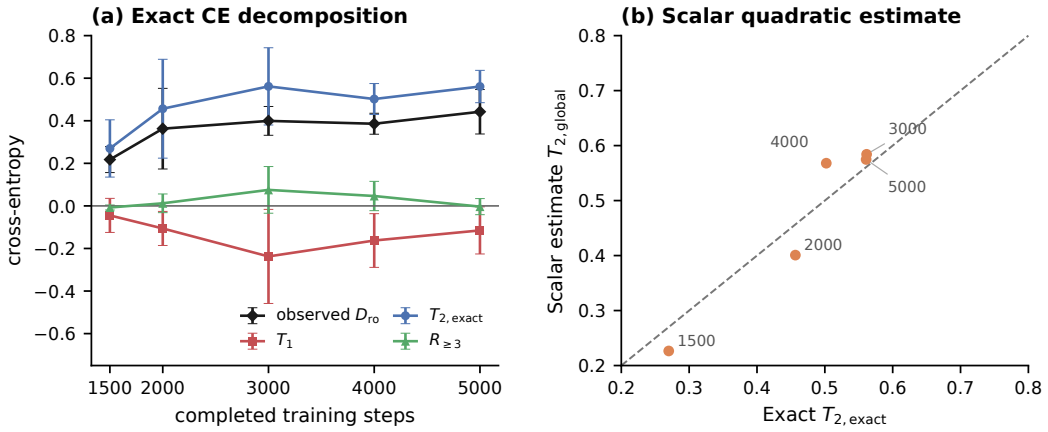


Figure 10: **Frozen replay Taylor decomposition** (five predeclared paired seed/chip units; pointwise two-sided 95% t intervals). (a) Observed penalty, first-order task projection T_1, exact Hessian-weighted second-order term, and higher-order remainder. (b) Scalar quadratic estimate versus the exact second-order term; labels give training steps, and the dashed line is equality.

Precision and scale. Since $\delta = \rho z_{\mathrm{rms}}$, two models with equal ρ but different z_{rms} incur quadratic loss in the ratio z_{rms}^2 whenever their softmax spread and moment conditions are comparable. Equal-z_{rms} models at different precision differ analogously through ρ. It is their product, not either factor, that controls Eq. (5). Raising ADC precision and removing exposed pre-DAC scale act on its two factors.

The following replay measures the correction terms that limit this scalar estimate.

D.7 First-order and Taylor decomposition

Frozen-checkpoint replay evaluates the terms of Eq. (24) on matched analog and ideal-head batches. Table 12 reports the loss penalty, first-order term, exact and approximated quadratic terms, and remainder. The ratio $|T_1|/T_{2,\mathrm{global}}$ measures the first-order contribution relative to the scalar estimate, while $T_{2,\mathrm{global}}/T_{2,\mathrm{exact}}$ assesses the curvature-neutrality approximation.

Table 12: **Frozen replay test of the assumptions behind the pure quadratic estimate.** Entries are five-seed means on matched analog/ideal batches. Ratios compare the first-order projection and the scalar quadratic estimate with the corresponding second-order terms.

step	δ	D_{ro}	T_1	$\lvert T_1\rvert/T_{2,\mathrm{global}}$	$T_{2,\mathrm{exact}}$	$T_{2,\mathrm{global}}$	$T_{2,\mathrm{global}}/T_{2,\mathrm{exact}}$	$R_{\geq 3}$
1500	0.693	0.2177	−0.0447	20%	0.2699	0.2265	84%	−0.0075
2000	0.937	0.3628	−0.1060	26%	0.4566	0.4008	88%	0.0122
3000	1.190	0.3994	−0.2374	41%	0.5616	0.5841	104%	0.0753
4000	1.162	0.3859	−0.1626	29%	0.5021	0.5680	113%	0.0465
5000	1.158	0.4423	−0.1151	20%	0.5609	0.5748	102%	−0.0034

Figure 10 plots the decomposition and the accuracy of the scalar quadratic estimate. At the earliest checkpoint, the interval for T_1 crosses zero. From step 2000 onward, T_1 is detectably negative and partially offsets the quadratic estimate. The scalar estimate remains close to the exact second-order term across these checkpoints. The late net-loss discrepancy comes chiefly from the first-order task projection under signal-correlated deterministic quantization.

D.8 Scale-damage response across forward-ADC precision

Appendix D.2 measures the scale-to-damage response at one operating point, which leaves open how much scale a different converter can absorb. This experiment replays the *same* frozen BP N-∅

checkpoints at four forward-ADC precisions, changing only that resolution. The out6 sweep is therefore the same reference condition as in Table 9.

All four precisions already show a detected cost at $c = 0.5$, so we instead report an iso-damage analysis against the prior prediction. For ADC bound B_{out}, the nominal restored ADC step is $\Delta := 2aB_{\text{out}}/(2^b - 2)$. When ADC error dominates and is spread within quantization bins, $\delta \propto c/(2^b - 2)$. Under Eq. (5) with comparable loss curvature, this gives a coarse-to-fine cost ratio of $\big((2^b - 2)/(2^{b-1} - 2)\big)^2 \approx 4$ and a scale budget that approximately doubles per bit.

Table 13: **Readout CE penalty and estimate accuracy across precision.** Same five frozen BP N-∅ checkpoints and matched evaluation batches at every cell; only the output ADC resolution differs. *frac* is the observed damage coefficient divided by the value Eq. (5) predicts for it, so 1.00 means the estimate prices the cost exactly. The out6 column reports CE and coefficient accuracy for the same replay experiment that Table 9 reports in PPL units.

	out5		out6		out7		out8	
c	CE	frac	CE	frac	CE	frac	CE	frac
0.50	0.0819	0.63	0.0437	0.96	0.0121	1.09	0.0030	0.96
1.00	0.4030	0.83	0.1563	0.95	0.0414	1.03	0.0104	0.91
1.35	0.6406	0.79	0.2507	0.92	0.0669	1.02	0.0172	0.91
2.00	0.9692	0.66	0.4288	0.88	0.1173	1.02	0.0314	0.94
3.00	1.2218	0.50	0.6546	0.79	0.1928	1.02	0.0532	0.94
mean frac over the low-error grid ($c \leq 1.35$):	0.765 (out5),		0.942 (out6),		1.042 (out7),		0.924 (out8)	

Over the low-error grid, Eq. (5) closely predicts the mean damage coefficient at out6–out8 (Table 13). The measured per-bit cost ratios and scale budgets also approximately follow the predicted scaling (Table 14). We estimate the scale budget for a 0.10 CE penalty by linear interpolation in c^2 between adjacent seed-mean measurements; the out8 crossing lies beyond the tested grid.

At out5, the coefficient departs further from the approximation, and the measured cost reduction from adding one bit is smaller than predicted. The converter response remains homogeneous under uniform scaling, but the scalar loss estimate and its predicted per-bit benefit are less accurate. This discrepancy alone does not distinguish first-order effects, curvature mismatch, higher-order terms, or failure of the ADC-dominated error assumption.

Table 14: Per-bit readout CE cost ratios and interpolated scale budgets. Measured ratios are medians over eight matched scales; budgets use a 0.10 CE penalty.

Precision pair	Measured cost ratio	Predicted cost ratio
out5 / out6	2.38	4.27
out6 / out7	3.69	4.13
out7 / out8	3.87	4.06

Precision	Scale budget c	Budget relative to out6
out5	0.541	0.68
out6	0.791	1.00
out7	1.795	2.27
out8	> 3	–

D.9 Input-scaling policy robustness

This replay tests whether the scale mechanism survives a change in the simulator's input-scaling rule. We evaluate P0 (`ABS_MAX+ITERATIVE`), P1 (`AVERAGE_ABS_MAX+ITERATIVE`), P2 (`ABS_MAX+NONE`), and P3 (`CONSTANT+NONE`). `CONSTANT+NONE` uses a per-checkpoint static input range calibrated before evaluation. Figure 11 shows the complete scale sweep, and Table 16 gives the two prespecified anchor points.

Replay protocol. The grid contains five checkpoint-frozen N-∅ sources, four policies, and eight scales $c \in \{0.5, 1, 1.25, 1.35, 1.5, 2, 2.5, 3\}$, for 160 cells. Every cell uses 16 matched evaluation batches. Four separate calibration batches are used only for CONSTANT+NONE, whose threshold is the 99.9th percentile of row maxima over a fixed 4×8 sample. Within a seed, all policies share the ideal logits, effective head weights, evaluation batches, and replay random state. The calibration and evaluation batches are disjoint. To prevent cycle-to-cycle update variation from entering the policy comparison, all four policies use the same frozen checkpoint state within a seed, and no retraining occurs during replay. The checkpoint/device seed is the unit of inference.

Throughout this subsection, $D_{\pi,s}(c)$ is analog-readout CE under policy π minus ideal-float CE for paired unit s, using the same scale c, checkpoint, effective head weights, and evaluation batches; only the analog term depends on π. Thus, $D > 0$ means that analog is worse than its matched ideal replay. At $c_{\text{rep}} = 1.35$, we compute the relative-corruption response $R_\pi := \rho_\pi(c_{\text{rep}})/\rho_\pi(1) - 1$ and the penalty change $\Delta D_\pi := D_\pi(c_{\text{rep}}) - D_\pi(1)$ within each seed before aggregation. For alternative policy π, $J_R(\pi) := R_\pi - R_{\text{P0}}$ compares the corruption responses, and $I(\pi) := \Delta D_\pi - \Delta D_{\text{P0}}$ compares the penalty changes. Table 15 reports these contrasts; seed subscripts are suppressed for readability.

P0 and P2 are identical at all eight scales, consistent with inactive iterative bound management at this operating point. Reconstructed bound-management activity and ADC clip fractions are zero in all 160 cells. P1 already has large error at the unscaled anchor. Its increasingly negative first-order term offsets the quadratic contribution, so the readout penalty decreases even as ρ rises. For P3, both policy contrasts against P0 are detected, but its own penalty change at the common anchor is not. Its static range initially benefits from improved range use and later clips. These results limit transfer of the scalar loss approximation across input-scaling policies.

Table 15: **Input-scaling policy changes the scale response.** Means over five paired checkpoint/device units; brackets are paired 95% CIs.

policy	R	$\Delta D = D(1.35) - D(1)$ (CE)	J_R [95% CI]	I [95% CI] (CE)
P0: per-input maximum + iterative	+0.000	+0.094	–	–
P1: mean input maximum + iterative	+0.158	−0.098	+0.158 [+0.124, +0.193]	−0.193 [−0.255, −0.131]
P2: per-input maximum + none	+0.000	+0.094	0.000 [0.000, 0.000]	0.000 [0.000, 0.000]
P3: calibrated static + none	−0.234	−0.029	−0.234 [−0.268, −0.201]	−0.123 [−0.175, −0.071]

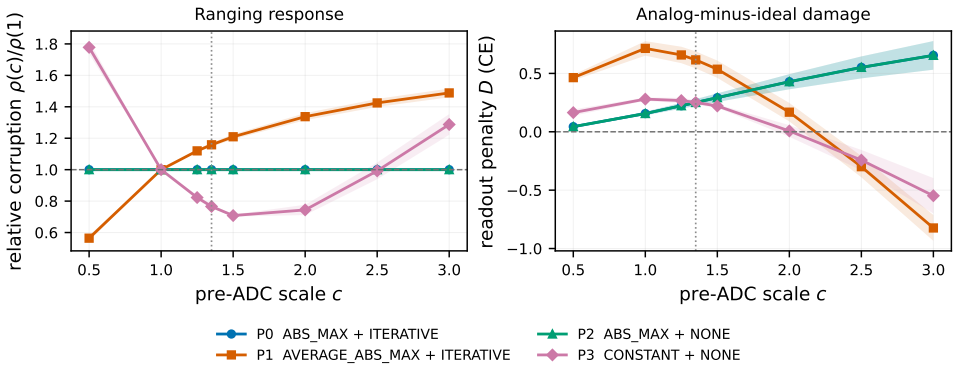


Figure 11: Relative corruption and readout CE penalty across scales under four input-scaling policies. Curves are five-seed paired means and bands are ± SE. The vertical line marks the common anchor $c_{\text{rep}} = 1.35$.

Training under AVERAGE_ABS_MAX+ITERATIVE. A separate experiment trains N-∅ and NA-∅ from scratch under AVERAGE_ABS_MAX+ITERATIVE on five fresh seed/chip pairs (seeds 5–9). The final NA-∅ analog CE is higher in all five pairs, with a paired interval above zero (Table 17). Both ideal CE and the readout penalty contribute to the mean increase. This result supports fixing the pre-DAC affine in the tested AVERAGE_ABS_MAX+ITERATIVE configuration.

Table 16: **Input-scaling policy anchors.** Values are mean ± SE over five paired checkpoint/device units, using the same fixed evaluation batches as the checkpoint-paired measurements.

policy	c	ρ	δ	D (CE)
P0: ABS_MAX+ITERATIVE	1.00	0.139 ± 0.006	0.606 ± 0.022	0.156 ± 0.016
P0: ABS_MAX+ITERATIVE	1.35	0.139 ± 0.006	0.818 ± 0.030	0.251 ± 0.029
P1: AVERAGE_ABS_MAX+ITERATIVE	1.00	0.538 ± 0.018	2.351 ± 0.085	0.714 ± 0.062
P1: AVERAGE_ABS_MAX+ITERATIVE	1.35	0.623 ± 0.021	3.676 ± 0.140	0.616 ± 0.072
P2: ABS_MAX+NONE	1.00	0.139 ± 0.006	0.606 ± 0.022	0.156 ± 0.016
P2: ABS_MAX+NONE	1.35	0.139 ± 0.006	0.818 ± 0.030	0.251 ± 0.029
P3: CONSTANT+NONE	1.00	0.186 ± 0.007	0.813 ± 0.023	0.280 ± 0.014
P3: CONSTANT+NONE	1.35	0.143 ± 0.006	0.840 ± 0.026	0.251 ± 0.020

Training with CONSTANT+NONE at the readout. The CONSTANT+NONE experiment changes only the final vocabulary MVM. For each seed/chip pair (seeds 10–14), four disjoint step-0 calibration batches set the constant input range to the 99.9th percentile of per-row absolute maxima. The threshold is then held fixed for all 5,000 updates and evaluations. The trunk retains ABS_MAX+ITERATIVE. The N-∅ and NA-∅ branches reproduce a common initial state and initial logits and use identical calibration and evaluation batches within each pair.

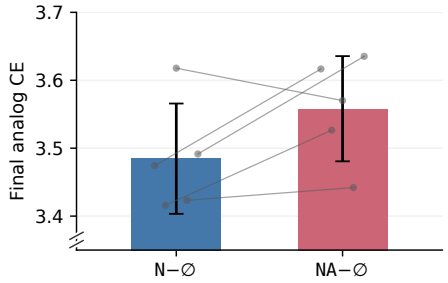


Figure 12: **Final analog CE under a calibrated static input range.** Bars show five-seed means, error bars show SD, and connected points show paired runs. The vertical axis is truncated.

Figure 12 compares the endpoints, and Table 17 separates the paired loss components. Neither the total nor either component interval excludes zero. Input clipping is higher for NA-∅ in all five pairs. The final-CE ordering under CONSTANT+NONE is unresolved. With the range held fixed, input growth changes range utilization and can increase clipping, rather than preserving converter codes under the uniform scaling of Section 4.1.

Table 17: Paired training differences, NA-∅ minus N-∅, under alternative input-scaling policies. Each policy uses five separate seed/chip pairs. Intervals are two-sided 95% t_4 intervals.

Policy	Quantity	Mean	SE	95% CI
P1	Analog CE	$+0.4002$	0.0506	$[+0.2597, +0.5407]$
P1	Ideal CE	$+0.1698$	0.0433	$[+0.0496, +0.2901]$
P1	Readout CE penalty	$+0.2304$	0.0259	$[+0.1584, +0.3023]$
P3	Analog CE	$+0.0735$	0.0379	$[-0.0317, +0.1788]$
P3	Ideal CE	$+0.1015$	0.0452	$[-0.0239, +0.2269]$
P3	Readout CE penalty	-0.0279	0.0339	$[-0.1222, +0.0663]$
P3	Input-clipping difference	0.0026–0.0099 (range across seeds)		

E SUPPLIED GRADIENTS AND GRADIENT INTERVENTIONS

E.1 LOSS SLOPES AND OPTIMIZER UPDATES

Loss slopes and their penalty components. At a frozen checkpoint t, we vary the gain along

$$\boldsymbol{\gamma}_t(s) := e^s \boldsymbol{\gamma}_t, \qquad \left.\frac{\mathrm{d}\boldsymbol{\gamma}_t(s)}{\mathrm{d}s}\right|_{s=0} = \boldsymbol{\gamma}_t, \tag{29}$$

with the trunk, head weights, bias, evaluation items, and hardware settings fixed. With a fixed bias, this path generally differs from uniform input scaling. Writing $\boldsymbol{\gamma} = \boldsymbol{\gamma}_t$, the ideal-loss slope and two penalty components are

$$\begin{aligned} \bar{D}_{\mathrm{ideal}} &= \frac{1}{d}\left\langle \nabla_{\boldsymbol{\gamma}} \mathcal{L}(z^\star), \boldsymbol{\gamma} \right\rangle, \\ \bar{D}^{\mathrm{vis}}_{\mathrm{pen}} &:= \frac{1}{d}\left\langle \widetilde{\nabla}_{\boldsymbol{\gamma}} \mathcal{L}(\tilde{z}) - \nabla_{\boldsymbol{\gamma}} \mathcal{L}(z^\star), \boldsymbol{\gamma} \right\rangle, \\ \bar{D}^{\mathrm{FD}}_{\mathrm{pen}} &:= \frac{D_{\mathrm{ro}}(e^{\eta}\boldsymbol{\gamma}) - D_{\mathrm{ro}}(e^{-\eta}\boldsymbol{\gamma})}{2\eta d}. \end{aligned} \tag{30}$$

The bar denotes division by the number of gain channels d; unbarred quantities omit this normalization. The predicted slope is $\bar{D}_{\mathrm{predicted}} = \bar{D}_{\mathrm{ideal}} + \bar{D}^{\mathrm{vis}}_{\mathrm{pen}}$, while the measured-response estimate is $\bar{D}_{\mathrm{measured}} = \bar{D}_{\mathrm{ideal}} + \bar{D}^{\mathrm{FD}}_{\mathrm{pen}}$. Negative slopes favor gain growth and positive slopes favor shrinkage along this path. The central secant at $\eta = 0.03$ measures a finite-scale response, which can include quantizer threshold crossings. It need not equal the local derivative; the scalar mismatch in Eq. (11) does not require such crossings.

We obtain the supplied and ideal gradients from two matched backward passes in evaluation mode, using the same readout weights, validation minibatches, and random states. The analog readout's backward signal is propagated through the digital affine to obtain $\widetilde{\nabla}_{\boldsymbol{\gamma}} \mathcal{L}(\tilde{z})$; replacing only the final-head multiplication with the exact operation gives $\nabla_{\boldsymbol{\gamma}} \mathcal{L}(z^\star)$. We average their projections across the fixed minibatches before clipping or parameter updates. The finite-difference response instead uses paired forward evaluations on these same items.

The ideal-loss slope. For one item, write $q := W(\boldsymbol{\gamma} \odot \mathtt{N}(h))$ and $b := W\boldsymbol{\beta}$, so $z^\star(e^s\boldsymbol{\gamma}) = e^s q + b$. With $p = \mathrm{softmax}(q + b)$,

$$\left.\frac{\mathrm{d}\ell(z^\star(e^s\boldsymbol{\gamma}), y)}{\mathrm{d}s}\right|_{s=0} = (p - e_y)^\top q = -\sum_{v \neq y} p_v (q_y - q_v). \tag{31}$$

The derivative is negative when $q_y > \sum_v p_v q_v$; $q_y > q_v$ for every $v \neq y$ is sufficient. With $\boldsymbol{\beta} = 0$, $q = z^\star$, but with nonzero bias, correct ranking by the full logits alone does not ensure this condition. The batch average, divided by d, gives $\bar{D}_{\mathrm{ideal}}$ and determines whether the ideal loss favors gain growth at the checkpoint.

Clipping and optimizer updates. Figure 13(c) records a related but distinct training-batch quantity. If $g^{\mathrm{raw}}_{\mathrm{train}}$ is the gradient of the training loss before global gradient clipping and $g^{\mathrm{clip}}_{\mathrm{train}}$ is the vector supplied to AdamW after clipping, then

$$\bar{D}^{\mathrm{raw}}_{\mathrm{train}} := \frac{\langle g^{\mathrm{raw}}_{\mathrm{train}}, \boldsymbol{\gamma}_t \rangle}{d}, \qquad \bar{D}^{\mathrm{clip}}_{\mathrm{train}} := \frac{\langle g^{\mathrm{clip}}_{\mathrm{train}}, \boldsymbol{\gamma}_t \rangle}{d}. \tag{32}$$

Global norm clipping at 1.0 multiplies all gradients by a common positive factor, so the raw and clipped projections have the same sign. Figure 13(c) shows the clipped projection supplied to AdamW (Loshchilov & Hutter, 2019).

AdamW preconditions this gradient before updating the gain. Let $\boldsymbol{\gamma}^{\mathrm{AdamW}}_{t+1}$ be the gain immediately after the optimizer update and before any clamp or other post-step control. The actual per-channel radial AdamW motion is

$$U_{\mathrm{AdamW}}(t) := \frac{1}{d}\left\langle \boldsymbol{\gamma}^{\mathrm{AdamW}}_{t+1} - \boldsymbol{\gamma}_t, \boldsymbol{\gamma}_t \right\rangle. \tag{33}$$

Thus, $U_{\mathrm{AdamW}} > 0$ means that the realized optimizer step increases the radial gain component; $U_{\mathrm{AdamW}} < 0$ means the opposite. It carries units of $\boldsymbol{\gamma}^2$ rather than of loss, so it is comparable with

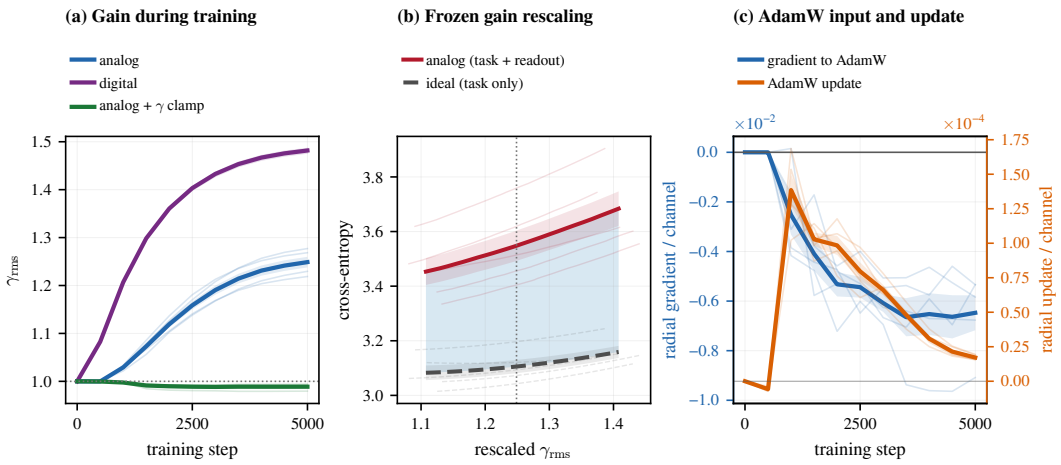


Figure 13: **Gain trajectories, frozen loss response, and AdamW updates.** (a) Gain magnitude during training. (b) Analog and ideal CE under gain rescaling at the frozen step-5000 checkpoint; blue shading marks the readout penalty. (c) Clipped gain-gradient and actual AdamW-update projections, each divided by the number of gain channels. Thin lines show five seeds, thick lines their means, and bands ± SE.

the radial direction quantities above in sign but not in magnitude; Figure 13(c) accordingly plots it on its own axis. The observed $\gamma_{\text{rms}}(t)$ trajectory separately shows the accumulated parameter motion.

E.2 Checkpoint diagnostics and probe-radius robustness

The measured and predicted CE changes in Figure 4(a) use the frozen step-5000 checkpoint and matched batches within each seed, with trunk, head, and bias fixed. We include all evaluated points within $|s| \leq 0.03$ and center each seed's loss at its own $s = 0$ value before aggregation. These CE changes are not divided by d. Figure 13 relates this loss response to gain trajectories and actual AdamW updates.

Figure 14 shows the final-checkpoint penalty components $\bar{D}^{\text{vis}}_{\text{pen}}$ and $\bar{D}^{\text{FD}}_{\text{pen}}$ for each seed. The training-time probes in Figure 4(b), this frozen-endpoint comparison, and the radius sweep below use different evaluation batches. Within each comparison, the supplied and finite-difference estimates use matched batches; the radius sweep also holds these batches fixed across radii and checkpoints.

Probe-radius robustness. We test the radius dependence of the central secant with an eight-scale replay at $\eta \in \{0.0025, 0.005, 0.01, 0.02, 0.03, 0.05, 0.08, 0.10\}$. Five paired seed/chip units replay the same matched batches at steps 1500, 2000, 3000, 4000, and 5000. The supplied penalty component is negative and the finite-difference response positive for every seed at every tested checkpoint and radius (Figure 15). The same replay's first-order and Taylor decomposition appears in Appendix D.7.

E.3 Digital scaling and the supplied backward signal

The max-absolute scale in Eq. (2) is input dependent. Its forward computation and its backward treatment are separate choices. In the AIHWKit implementation used here, `AnalogFunction` defines a custom autograd operation. Its forward calls the tile's forward routine without recording the internal operations in an autograd graph; its backward calls a separate tile backward routine. The latter applies the configured backward MVM, including backward-side input scaling when applicable. That backward-side scaling acts on the incoming error signal; it is not the chain-rule derivative of the forward input's max-absolute scale. Fixed digital mapping scales are also distinct from this per-input scale. In the primary `AnalogTile` perfect-backward control, the returned input signal is $W^\top e$ in effective-weight coordinates.

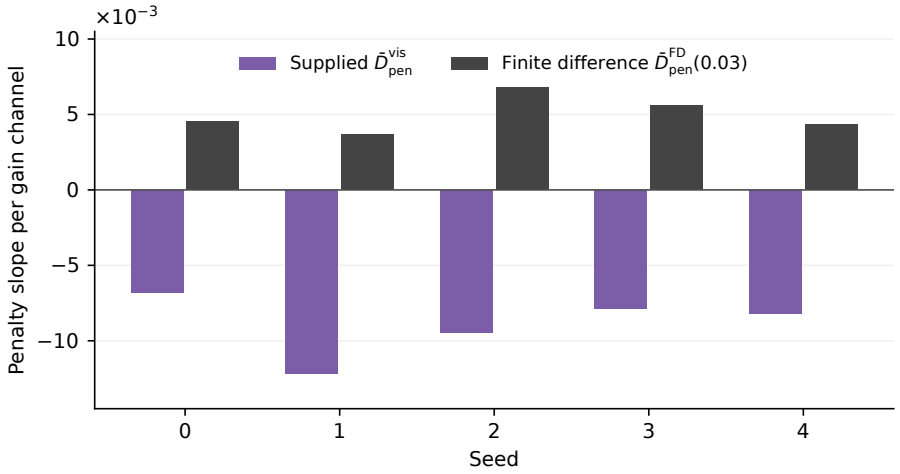


Figure 14: **Supplied and finite-difference penalty slopes at the final checkpoint.** Paired values for analog `NA-∅` at $\eta = 0.03$, divided by the number of gain channels.

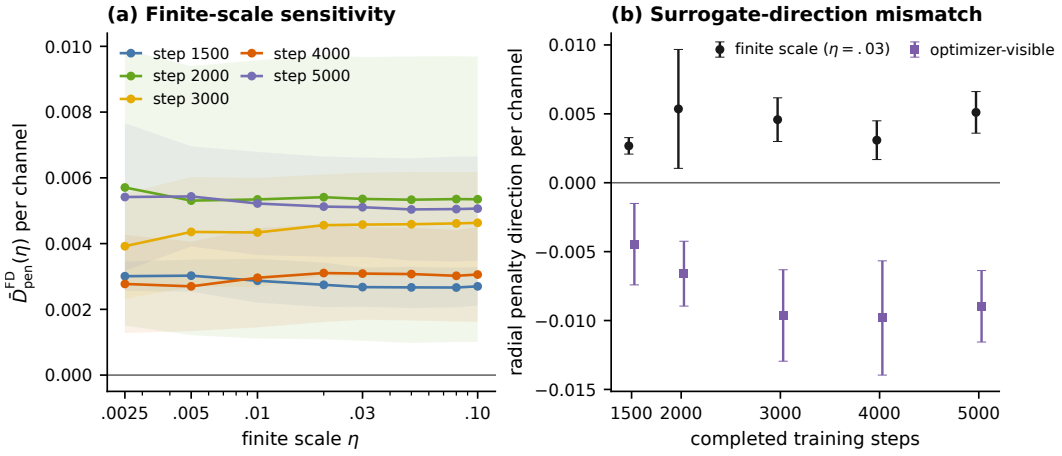


Figure 15: **Probe-radius robustness.** (a) Finite-difference penalty slopes over eight radii. (b) Supplied and finite-difference penalty slopes at $\eta = 0.03$. Means and pointwise 95% t intervals over five seeds, normalized by $d = 128$.

The `InferenceTile` implementation additionally invokes peripheral forward/backward routines. Its peripheral backward mask compares the restored tile output with 0.999 times the configured output bound and output scale, and can zero incoming error coordinates before the perfect MVM. With that mask written as M, its supplied signal is then $W^\top Me$. This mask is separate from both backward quantization and the digital-scale chain rule; `AnalogTile` overrides these peripheral routines. An unmodified inference backend therefore cannot be assumed to supply $W^\top e$ merely because its backward MVM is perfect.

To expose the scaling term, write the forward computation as

$$x := h_P, \qquad a = a(x) = \|x\|_\infty, \qquad u := x/a, \qquad b := F(u), \qquad \tilde{z} = ab,$$

where F contains the fixed-range DAC, analog multiplication, and ADC. Consider a nonzero input with a unique largest-magnitude coordinate, away from any change in the bound-management branch. Let J_F denote a chosen surrogate Jacobian for the inner operator. Differentiating the two digital scaling operations while retaining that inner surrogate gives

$$J_{\text{explicit}} = J_F + (b - J_F u)(\nabla_x a)^\top. \tag{34}$$

With the perfect linear inner surrogate $J_F = W$, the supplied input-gradient signal becomes

$$g_x^{\text{explicit}} = W^\top e + \left[e^\top (b - Wu)\right] \nabla_x a. \tag{35}$$

This expression differentiates the digital scaling exactly and retains the chosen surrogate for the converter. Its accuracy in a given input direction therefore depends on that surrogate. For a unique peak $k := \arg\max_j |x_j|$, we have $\nabla_x a = \operatorname{sgn}(x_k)\mathbf{e}_k$, where $\mathbf{e}_k$ is the coordinate basis vector, so the added input-gradient term acts only on the peak coordinate. Nonpeak coordinates retain the chosen inner surrogate signal. The peak can vary across examples and training steps, so this pointwise sparsity does not imply that only one gain coordinate changes during training. Appendix E.5 tests the correction during training with channelwise gain and bias.

Along the scalar path $x = gr$ with fixed r and $g > 0$, the normalized input u is constant and $(\nabla_x a)^\top x = a$. Hence Eq. (34) gives

$$J_{\text{explicit}} x = \tilde{z}, \qquad (g_x^{\text{explicit}})^\top x = e^\top \tilde{z} = g \, \frac{\mathrm{d}\ell(\tilde{z}(g), y)}{\mathrm{d}g}.$$

Differentiating the digital scale therefore recovers the exact scalar response even when the inner operator uses a surrogate. Placing the gain after conversion gives the identical forward function $g \operatorname{Analog}(W, r)$ and the same gain derivative. The native perfect-MVM backward instead supplies $e^\top W x$, omitting the conversion-error contribution identified in Eq. (12).

For the scalar setting $\boldsymbol{\gamma} = g\mathbf{1}$ and $\boldsymbol{\beta} = 0$, averaging that identity over M prediction items and dividing by the number of gain channels gives

$$\frac{1}{d} \left.\frac{\mathrm{d}\mathcal{L}(\tilde{z}(e^s g))}{\mathrm{d}s}\right|_{s=0} - \bar{D}_{\text{predicted}} = \frac{1}{Md} \sum_{n=1}^{M} e_n(g)^\top \epsilon_n(g).$$

Along this path, converter codes remain fixed and the losses are smooth in s. Consequently,

$$\bar{D}_{\text{measured}} = \frac{1}{d} \left.\frac{\mathrm{d}\mathcal{L}(\tilde{z}(e^s g))}{\mathrm{d}s}\right|_{s=0} + O(\eta^2).$$

This finite-difference approximation applies to the scalar path; channelwise gain changes with fixed bias can cross converter thresholds. Finite-precision rounding can also affect the normalized input, so the scalar control below checks forward and converter-code agreement.

E.4 Scalar digital-scale control

This control isolates the backward convention without changing forward geometry or parameter count, testing the scalar-gain argument in Eq. (11). It uses the five primary BP `N-∅` step-5000 checkpoints, with the analog trunk and parameter-free final normalization retained. We cache 32,768 prediction items from the enwik8 training split and 8,192 from its validation split, using separate fixed sampling seeds; all three branches within a source unit use the same representations, labels, and training minibatch sequence. This design holds the learned representation fixed while testing how the backward convention affects scalar readout adaptation.

Three matched readouts. Each branch has one positive scalar gain $g = \exp(\theta)$, initialized to one, and the same bias-free effective head matrix W. The branches compute

$$\begin{aligned} \text{pre, native:} \quad & \operatorname{Analog}_{\text{ABS_MAX}}(W, gr), \\ \text{post:} \quad & g \operatorname{Analog}_{\text{ABS_MAX}}(W, r), \\ \text{pre, explicit:} \quad & a \operatorname{Analog}_{\text{NONE}}(W, gr/a), \qquad a = \|gr\|_\infty. \end{aligned}$$

The explicit branch retains both digital scale operations in autograd and disables the inner tile's max-absolute scaling to avoid applying it twice. All branches retain six-bit DAC and ADC, bounds 1 and 12, zero read noise, deterministic rounding, and iterative bound management. The inner backward uses the perfect linear surrogate. We implement exact weight updates with `InferenceTile` and disable its additional peripheral output-bound gradient mask in all three branches, matching the primary `AnalogTile`'s pure-MVM backward convention (Appendix E.3 defines this convention).

Forward equivalence and gradient accuracy. At gains $0.5, 0.75, 1, 1.25, 1.5, 2$, we compare forward outputs, supplied gain signals, analytic derivatives, and central finite differences with log-gain radii $0.001, 0.01, 0.03$. Across all five source checkpoints, the DAC and ADC codes remain unchanged and iterative bound rescaling is inactive. Branch outputs agree within 1.91×10^{-6}, and the native, post, and explicit gradient identities hold within 2.10×10^{-6}. At $g = 1$, the actual derivative is positive and the native supplied signal negative in every checkpoint, demonstrating the mismatch without converter-code changes.

Training protocol and scope. We either freeze both trunk and head and train only θ, or freeze the trunk and train W together with θ. Both start from the same source weights and $g = 1$. Both stages use 1,000 updates, batches of 512 prediction items, AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.95$, no weight decay, constant gain learning rate 0.003, and head learning rate 0.0003 when applicable. There is no gradient clipping. We evaluate the same validation items every 50 updates. The primary contrast is the paired native-minus-post final CE with frozen head weights; the trainable-head contrast is secondary. Source checkpoints are the units for the five-pair t_4 intervals.

Table 18: **Forward-equivalent scalar-gain training controls.** Five paired source checkpoints, 1,000 updates. Entries are mean validation CE; brackets give the paired native-minus-post 95% t_4 interval. The trunk is frozen in both rows.

head weights	pre, native	post	pre, explicit	native − post [95% CI]
fixed	3.0477	3.0194	3.0194	$+0.0284\ [-0.0006, +0.0574]$
trainable	3.1099	2.8803	2.8803	$+0.2296\ [+0.1145, +0.3447]$

Table 18 shows no statistically detected native-minus-post CE gap with frozen head weights, but a positive gap when the head is trainable. Explicit digital-scale differentiation yields nearly identical endpoint CE to the post-ADC gain in both cases. Figure 16 shows that the native branch increases gain, whereas the explicit and post branches reduce it and follow nearly overlapping gain and CE trajectories throughout training. When the head is trainable, the explicit and post branches can still diverge slightly, because rounding differences lead to different quantizer crossings and training trajectories even when the exact-arithmetic forward functions coincide.

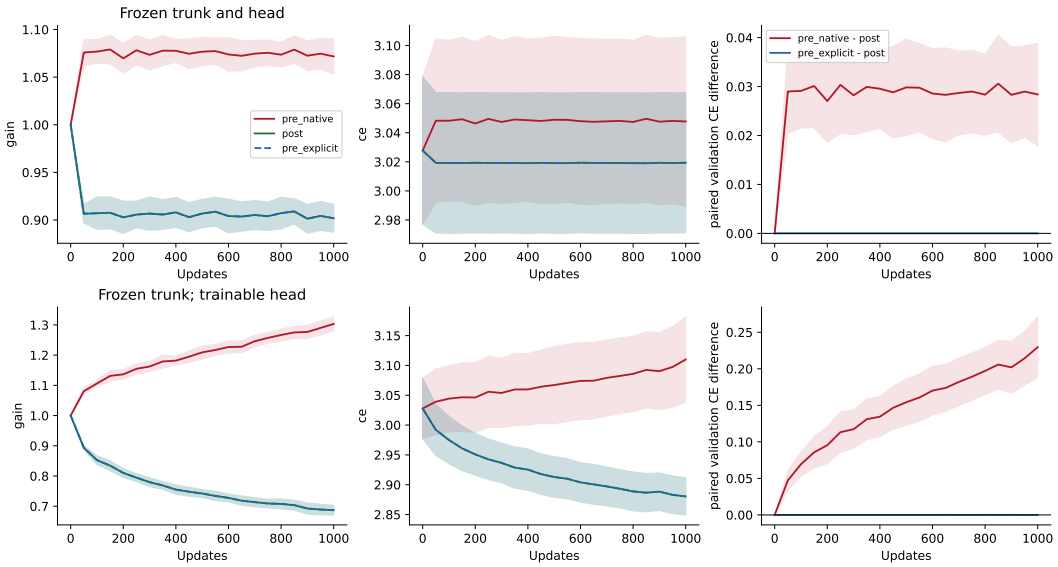


Figure 16: **Scalar-gain training with identical forward parameterizations.** Five-source means with ± SE bands. The top row freezes the trunk and head; the bottom row updates the head while keeping the trunk frozen. Columns show gain, validation CE, and paired CE differences from the post-ADC scalar parameterization.

E.5 CHANNELWISE DIGITAL-SCALE BACKWARD CONTROL

We next evaluate explicit digital-scale differentiation with channelwise gain and bias, a trainable trunk, and pulsed head updates. Within each of five fresh source units, native-backward NA-∅, explicit-scale NA-∅, and N-∅ share the initial trunk/head weights and chip parameters. All branches train the enwik8 byte-level d128-L4 model for 5,000 steps under the primary MpAdamW recipe, with six-bit forward and backward converters. The same fixed validation batches evaluate every branch, and evaluation preserves the training data RNG state.

Digital-scale gradient correction. Let v be the native head-input gradient, e the incoming output gradient, and $a = \|x\|_\infty$. We apply the digital-scale correction to the head-input gradient as

$$v_{\text{explicit}} := v + \frac{e^\top \tilde{z} - v^\top x}{a} \nabla_x a.$$

This retains the native inner backward convention while differentiating the two digital scale operations in Appendix E.3. The correction affects the gain, bias, and trunk gradients. It preserves the native head forward and the raw signals supplied to the head's weight-update rule; subsequent global gradient clipping can still affect all updates. No ideal-head loss or gradient enters training. The corrected input gradient agrees numerically with explicit differentiation of the digital scaling operations under both floating-point and six-bit inner backward. It satisfies the radial identity while retaining the raw weight-update signals. The correction is applied once per training step.

Table 19: **Explicit digital-scale differentiation in channelwise training.** Top entries are final five-seed means ± SD; lower CE and PPL are better. Native and explicit use NA-∅; fixed uses N-∅. Bottom entries are paired CE differences with 95% t_4 intervals. The first contrast is primary; the other two are unadjusted secondary comparisons.

Metric	Native	Explicit	Fixed
Analog CE	3.595 ± 0.131	3.629 ± 0.162	3.095 ± 0.063
Analog PPL	36.67 ± 4.83	38.08 ± 6.22	22.13 ± 1.39
Ideal-readout CE	3.168 ± 0.171	3.519 ± 0.125	2.905 ± 0.033
Readout penalty (CE)	0.427 ± 0.121	0.111 ± 0.049	0.191 ± 0.036
Gain RMS	1.258 ± 0.021	0.897 ± 0.026	1.000 ± 0.000
Bias RMS	0.253 ± 0.055	0.067 ± 0.028	0.000 ± 0.000
Absolute logit error δ	1.225 ± 0.073	0.525 ± 0.086	0.668 ± 0.066
Ideal-logit RMS σ_z	6.325 ± 1.299	2.952 ± 0.628	4.069 ± 0.538
Relative corruption ρ	0.202 ± 0.054	0.190 ± 0.076	0.166 ± 0.024

Contrast	Mean ΔCE	95% interval
Explicit − native (primary)	$+0.034$	$[-0.229, +0.298]$
Fixed − native	-0.500	$[-0.680, -0.320]$
Explicit − fixed	$+0.534$	$[+0.370, +0.698]$

Effect on training. Table 19 shows no detected improvement in final analog CE from explicit digital-scale differentiation. The primary interval includes zero, while the corrected configuration retains a positive gap to N-∅ under the secondary comparison. The explicit branch has lower mean gain, absolute logit perturbation, and readout penalty. Its higher ideal-readout replay CE offsets the reduction in readout penalty. This replay replaces only the final head at each model's own learned state and retains the analog trunk. Thus, the correction suppresses gain growth and readout damage in this setting, but does not eliminate the endpoint performance gap.

E.6 RADIAL GAIN-GRADIENT INTERVENTION

We test a radial-gradient intervention in the same d128-L4/out6 from-scratch setting. On each training batch, the correction adjusts only the penalty projection along the current gain vector. Write

$g_{\mathrm{vis}} := \widetilde{\nabla}_{\boldsymbol{\gamma}}\mathcal{L}(\tilde{z})$ for the supplied gain gradient and let $\varepsilon > 0$ be a fixed numerical stabilizer. Then

$$g_{\mathrm{corr}} := g_{\mathrm{vis}} + \frac{D_{\mathrm{pen}}^{\mathrm{FD}} - D_{\mathrm{pen}}^{\mathrm{vis}}}{\|\boldsymbol{\gamma}\|_2^2 + \varepsilon}\boldsymbol{\gamma},$$
$$\langle g_{\mathrm{corr}}, \boldsymbol{\gamma}\rangle = D_{\mathrm{ideal}} + D_{\mathrm{pen}}^{\mathrm{FD}} - \frac{\varepsilon(D_{\mathrm{pen}}^{\mathrm{FD}} - D_{\mathrm{pen}}^{\mathrm{vis}})}{\|\boldsymbol{\gamma}\|_2^2 + \varepsilon}. \tag{36}$$

The correction leaves every non-radial gradient component unchanged before global clipping and optimization. Five paired seed/chip units compare the ordinary surrogate with correction every step and every ten steps. We also include three structural controls: capping gain at one (NA-$\varnothing$ + $\boldsymbol{\gamma}$-clamp), fixed gain and bias (N-$\varnothing$), and the deferred readout N-A.

Table 20: **Radial gain-gradient interventions and structural controls.** Five-seed endpoint means after 5,000 steps at d128-L4/out6. Differences are paired analog CE contrasts against the supplied-gradient baseline, with 95% t_4 intervals.

Condition	Analog CE	Ideal CE	D_{ro}	γ_{rms}	Δ analog CE [95% CI]
Supplied gain gradient	3.6552	3.1396	0.5155	1.2189	reference
Radial every step	3.5211	3.2114	0.3097	0.6209	-0.1340 $[-0.2796, 0.0115]$
Radial every 10 steps	3.6873	3.3148	0.3725	1.0045	$+0.0321$ $[-0.0587, 0.1230]$
Cap gain at one	3.0882	2.9058	0.1824	0.9920	-0.5670 $[-0.6420, -0.4919]$
Fixed gain and bias	3.0917	2.9051	0.1866	1.0000	-0.5635 $[-0.6979, -0.4291]$
Deferred affine (N-A)	3.0688	2.9503	0.1184	1.0000	-0.5864 $[-0.7145, -0.4582]$

The correction satisfies Eq. (36) before global gradient clipping. Every-step correction lowers mean gain and readout penalty but raises ideal CE, leaving the net-CE improvement statistically unresolved (Table 20). Correction every ten steps also has no detected net effect, whereas all three structural controls lower CE. These results show that better readout fidelity can be offset by worse task fit; channelwise gain allocation and model co-adaptation may contribute to this tradeoff. The radial-only intervention leaves non-radial components unchanged, unlike the full-vector intervention below.

E.7 Full-vector gain-gradient intervention

We next replace the complete gain-penalty vector in a six-arm intervention. Ten source units (seeds 0–9, each with its device realization locked to the seed) supply one common step-0 state to all branches. Every branch trains for 5,000 steps under the primary pulsed MpAdamW recipe and d128-L4/out6 forward configuration, and the same fixed evaluation batches. Within a unit the branches begin from an identical state and identical initial logits. The intervention targets the unquantized digital gain and estimates the readout penalty's response to its perturbation. Additional paired forward evaluations supply the penalty response, and an ideal-head gradient retains the task component. Holding the other gradient components fixed before global clipping and optimization lets us test the effect of gain feedback on training. The structural remedies avoid these extra evaluations. The primary contrast is penalty replacement minus the supplied-gradient baseline, using a two-sided paired t_9 interval; replacement minus fixed gain and bias is secondary.

The supplied gain gradient $\widetilde{\nabla}_{\boldsymbol{\gamma}}\mathcal{L}(\tilde{z})$, ideal-head gradient $\nabla_{\boldsymbol{\gamma}}\mathcal{L}(z^\star)$, and penalty component $\boldsymbol{p}_{\mathrm{vis}}$ follow Eq. (13). The branches below specify the gradient used to update $\boldsymbol{\gamma}$ or fix its parameters.

branch	gradient used to update $\boldsymbol{\gamma}$	reading
Supplied gain gradient	$\widetilde{\nabla}_{\boldsymbol{\gamma}}\mathcal{L}(\tilde{z})$	exposed readout, NA-$\varnothing$
Remove penalty	$\nabla_{\boldsymbol{\gamma}}\mathcal{L}(z^\star)$	$\boldsymbol{p}_{\mathrm{vis}} \to \mathbf{0}$
Negate penalty	$\nabla_{\boldsymbol{\gamma}}\mathcal{L}(z^\star) - \boldsymbol{p}_{\mathrm{vis}}$	$\boldsymbol{p}_{\mathrm{vis}} \to -\boldsymbol{p}_{\mathrm{vis}}$
Replace penalty	$\boldsymbol{G}_{\mathrm{repair}}$	$\boldsymbol{p}_{\mathrm{vis}} \to \boldsymbol{p}_{\mathrm{FD}}$
Cap gain at one	$\widetilde{\nabla}_{\boldsymbol{\gamma}}\mathcal{L}(\tilde{z})$, then $\boldsymbol{\gamma} \le \mathbf{1}$	bound gain after each step
Fixed gain and bias	none	$\boldsymbol{\gamma} = \mathbf{1}, \boldsymbol{\beta} = \mathbf{0}$, giving N-$\varnothing$

Here $\boldsymbol{p}_{\mathrm{FD}}$ is obtained from the mean of two reproducible antithetic Rademacher estimates of the full penalty response in log-gain coordinates at $\eta = 0.03$, then converted to gain coordinates. Writing m

Table 21: **Full-vector gain-gradient interventions.** Endpoint means over ten paired seeds after 5,000 steps at d128-L4/out6. Differences are analog CE contrasts with 95% t_9 intervals; the top panel uses the supplied-gradient baseline.

branch	analog CE	ideal CE	D_{ro}	γ_{rms}	Δ analog CE [95% CI]
Supplied gain gradient (NA-∅)	3.5631	3.1928	0.3703	1.2548	reference
Remove penalty	3.2989	3.0337	0.2652	1.0675	-0.2642 $[-0.3897, -0.1388]$
Negate penalty	3.1736	2.9827	0.1909	1.0041	-0.3895 $[-0.5346, -0.2444]$
Replace penalty	**3.1039**	2.9319	**0.1720**	1.0048	$\mathbf{-0.4593}$ $[\mathbf{-0.5947}, \mathbf{-0.3238}]$
Cap gain at one	3.1932	2.9756	0.2176	0.9886	-0.3700 $[-0.4640, -0.2759]$
Fixed gain and bias (N-∅)	**3.1018**	**2.9264**	0.1754	1.0000	-0.4613 $[-0.5792, -0.3435]$

Secondary contrast	Analog CE difference [95% CI]
Replacement minus fixed gain and bias	$+0.0021$ $[-0.0754, +0.0796]$
Replacement minus removal	-0.195 $[-0.289, -0.101]$
Replacement minus negation	-0.070 $[-0.191, +0.052]$

for the number of directions and $v_k \in \{-1, +1\}^d$ for each direction, the estimator is

$$\widehat{q} := \frac{1}{m} \sum_{k=1}^{m} \frac{D_{\mathrm{ro}}(\boldsymbol{\gamma} \odot e^{\eta v_k}) - D_{\mathrm{ro}}(\boldsymbol{\gamma} \odot e^{-\eta v_k})}{2\eta} v_k, \qquad (\boldsymbol{p}_{\mathrm{FD}})_j = \widehat{q}_j / \gamma_j^{\mathrm{safe}}. \tag{37}$$

The two directions are resampled at every training step using a dedicated pseudorandom generator keyed by source unit, step, and direction index. The exponential acts coordinatewise. The denominator γ_j^{safe} preserves the gain's sign while flooring its magnitude at 10^{-6} (zero uses the positive floor). This implements the chain-rule conversion from multiplicative log-gain perturbations to the additive gain-gradient coordinates expected by the optimizer, with a safeguard near zero. Each penalty evaluation is the analog-minus-ideal loss on the same batch, with the other weights fixed and replay random states matched. With $m = 2$, the estimator requires four penalty evaluations, each comprising an analog and an ideal-head forward pass, in addition to the ideal-head forward and gradient computation used to obtain $\nabla_{\boldsymbol{\gamma}} \mathcal{L}(z^\star)$. These replacements occur before global clipping and AdamW. At that stage, the ideal task gradient and all other parameter gradients are unchanged; the replacement can subsequently change the global clipping factor and hence other parameters' updates. The supplied-gradient branch is the NA-∅ cell trained as usual, and fixing gain and bias supplies the target endpoint.

The primary finite-scale branch improves final analog CE relative to the ordinary surrogate and nearly closes the gap to fixed normalization in mean (Table 21). Removing or negating the penalty also recovers part of the gap. The paired intervals show a benefit of full-vector replacement over removal but no detected benefit over negation; the latter comparison does not establish that estimating the penalty magnitude is necessary. These results support a causal role for harmful feedback to the gain vector at this operating point. In the separate radial intervention of Eq. (36), lower gain magnitude and readout penalty accompany worse ideal CE, leaving the net improvement undetected. This contrast links the endpoint gap to gain feedback while leaving the relative roles of overall scale, channelwise gain allocation, and trunk/head co-adaptation unresolved.

E.8 Bias-gradient intervention

We test the bias counterpart of Eq. (13) in a separate experiment. Ten paired source units (seeds 0–9) each supply an identical step-0 state to ordinary training and bias penalty replacement. Both branches train the exposed NA-∅ readout for 5,000 steps at d128-L4 with six-bit forward and backward converters, zero read noise, and pulsed updates. The learning-rate schedule and device settings follow the full-vector gain experiment above. Evaluation uses sixteen fixed validation batches of 128 examples. Final analog CE is the primary contrast; all pairs are included without pooling with the gain experiment. There is no fixed-normalization arm in this experiment.

For the bias branch, we retain the ideal-head bias gradient on the current training batch and replace only its penalty residual. Unlike the multiplicative gain probes, additive bias probes remain defined

at the initial $\boldsymbol{\beta} = 0$. With two independent Rademacher directions $u_k \in \{-1, +1\}^d$ resampled at every step, the replacement is

$$\boldsymbol{p}_{\mathrm{FD},\beta} = \frac{1}{m} \sum_{k=1}^{m} \frac{D_{\mathrm{ro}}(\boldsymbol{\beta} + h u_k) - D_{\mathrm{ro}}(\boldsymbol{\beta} - h u_k)}{2h} u_k, \qquad m = 2, \quad h = 0.03. \tag{38}$$

The radius is absolute in normalized-feature units and fixed before training. Each penalty evaluation uses analog and ideal-head losses on the same batch with all other parameters fixed. After each probe, the bias, random states, and backward state at the unperturbed parameters are restored. The intervention changes only the bias gradient before global clipping; the gain, trunk, and head continue to learn, and clipping and subsequent co-adaptation can affect their updates. This finite-scale, two-direction estimate is not an exact penalty gradient.

Table 22: **Bias-only penalty-gradient replacement.** Endpoint means over ten paired runs. Differences are replacement minus ordinary, with paired 95% t_9 intervals; analog CE is primary and the remaining rows are descriptive.

metric	ordinary	bias replacement	paired difference [95% interval]
Analog CE	3.5934	3.5762	-0.0172 $[-0.1163, +0.0819]$
Ideal-head CE	3.1982	3.1831	-0.0151 $[-0.1488, +0.1186]$
Readout penalty	0.3952	0.3932	-0.0021 $[-0.0879, +0.0838]$
Gain RMS	1.2458	1.2817	$+0.0359$ $[+0.0013, +0.0705]$
Bias RMS	0.2601	0.1209	-0.1392 $[-0.1776, -0.1008]$

The intervention lowers bias RMS but yields no statistically detected endpoint improvement or readout-penalty reduction (Table 22). This experiment tests the effect of the finite-scale bias-gradient correction; the effect of bias trainability is not isolated here.

E.9 Separating gain and bias trainability

We independently enable training of the final LayerNorm gain and bias to test which parameter can reproduce the degradation. Ten paired seeds (0–9) each supply the same initial state and training batches to four arms: both parameters fixed, trainable gain only, trainable bias only, and both trainable. Fixed gain remains exactly $\boldsymbol{\gamma} = \mathbf{1}$ and fixed bias remains exactly $\boldsymbol{\beta} = \mathbf{0}$ throughout training. All arms retain the same affine LayerNorm architecture and trainable trunk and head; only the final affine parameters' trainability differs. These runs use ordinary backpropagation without gradient replacement.

Each arm trains for 5,000 steps on enwik8 at d128-L4 with six-bit forward and backward converters, zero read noise, and pulsed updates. The learning-rate schedule and optimizer settings follow Appendix A. Evaluation uses sixteen fixed validation batches of 128 examples and context length 64. All four arms are trained together in this experiment; paired comparisons use its own controls, rather than the gradient-intervention baselines in Figure 5.

Table 23: **Gain and bias trainability.** Endpoint mean $\pm$ sample SD over ten paired seeds.

Trainable parameters	Analog CE	Ideal-head CE	Readout penalty
Neither	3.1208 ± 0.0630	2.9407 ± 0.0450	0.1801 ± 0.0277
Gain only	3.5622 ± 0.0885	3.1773 ± 0.0993	0.3849 ± 0.0534
Bias only	3.1375 ± 0.1038	2.9348 ± 0.0931	0.2027 ± 0.0516
Gain and bias	3.6165 ± 0.1863	3.1214 ± 0.1227	0.4951 ± 0.1005

Table 23 gives the endpoints. The primary contrasts compare gain-only and bias-only training with the fixed-affine arm on final analog CE. Table 24 also reports the secondary contrasts that enable one parameter while the other is trainable. The interaction is computed within each seed as both trainable minus gain only minus bias only plus neither trainable. Intervals are nominal paired 95% t_9 intervals without multiplicity adjustment.

Trainable gain alone increases analog CE, ideal-head CE, and the readout penalty, reproducing the degradation with bias fixed at zero. Trainable bias alone yields no statistically detected increase in

Table 24: **Paired effects of affine trainability.** Mean difference [95% interval].

Contrast	Analog CE	Ideal-head CE	Readout penalty
Gain only − neither	$+0.441\ [0.358, 0.525]$	$+0.237\ [0.151, 0.322]$	$+0.205\ [0.163, 0.247]$
Bias only − neither	$+0.017\ [-0.054, 0.087]$	$-0.006\ [-0.072, 0.060]$	$+0.023\ [-0.021, 0.066]$
Both − gain only	$+0.054\ [-0.117, 0.226]$	$-0.056\ [-0.192, 0.080]$	$+0.110\ [0.043, 0.178]$
Both − bias only	$+0.479\ [0.305, 0.653]$	$+0.187\ [0.061, 0.312]$	$+0.292\ [0.205, 0.380]$
Interaction	$+0.038\ [-0.189, 0.264]$	$-0.050\ [-0.216, 0.116]$	$+0.088\ [-0.007, 0.183]$

these metrics. When gain is also trainable, enabling bias increases the readout penalty, although the net analog-CE change remains unresolved. Thus, gain alone is sufficient for degradation at this setting, but the results do not establish that bias is harmless. The interaction intervals include zero for all three metrics.

F Backward Precision and Optimizer Controls

F.1 Backward precision and weight-update controls

We compare N-∅, NA-∅, and N-A over five paired seeds under three conditions: pulsed MpAdamW with 6-bit or floating-point backward MVMs, and exact updates with floating-point backward MVMs. The exact-update inference backend uses floating-point backward MVMs irrespective of its backward converter settings; the cross-optimizer study and the exact-update localization control use the same exact-update configuration. All runs share the d128-L4 out6 model, learning-rate schedule and optimizer hyperparameters, forward precision, chip identity, matched evaluation, and trainable affine parameters in every normalization other than the readout. Pairing matches initialization, chip, and data stream, but not individual stochastic pulse updates. Figure 17 shows endpoint gaps and gradient diagnostics; Table 25 gives paired contrasts for all three readouts. The six N-A endpoint-CE comparisons are exploratory, with no multiple-comparison adjustment; ideal CE and readout penalty give their decomposition.

Table 25: **Readout contrasts across backward precisions and update backends.** Paired CE differences with 95% t_4 intervals over five seeds. Forward reads remain 6-bit; backward precision applies to both MVM input and output.

Contrast	Analog CE Δ	Ideal CE Δ	ΔD_{ro}
Pulsed MpAdamW, 6-bit backward			
NA-∅ − N-∅	$0.5251\ [0.4584, 0.5917]$	$0.2218\ [0.0986, 0.3449]$	$0.3033\ [0.1899, 0.4167]$
N-A − NA-∅	$-0.5519\ [-0.6283, -0.4755]$	$-0.1827\ [-0.3312, -0.0343]$	$-0.3692\ [-0.4696, -0.2687]$
N-A − N-∅	$-0.0269\ [-0.1134, 0.0597]$	$0.0390\ [-0.0369, 0.1150]$	$-0.0659\ [-0.0874, -0.0443]$
Pulsed MpAdamW, float backward			
NA-∅ − N-∅	$-0.0579\ [-0.1539, 0.0382]$	$-0.0841\ [-0.1796, 0.0113]$	$0.0262\ [0.0223, 0.0302]$
N-A − NA-∅	$0.0591\ [-0.0920, 0.2101]$	$0.0896\ [-0.0641, 0.2432]$	$-0.0305\ [-0.0349, -0.0261]$
N-A − N-∅	$0.0012\ [-0.2105, 0.2128]$	$0.0055\ [-0.2060, 0.2169]$	$-0.0043\ [-0.0059, -0.0027]$
Exact-update AdamW, float backward			
NA-∅ − N-∅	$0.0238\ [-0.0290, 0.0767]$	$-0.0082\ [-0.0587, 0.0423]$	$0.0320\ [0.0275, 0.0366]$
N-A − NA-∅	$-0.0304\ [-0.0528, -0.0081]$	$0.0048\ [-0.0168, 0.0265]$	$-0.0353\ [-0.0392, -0.0313]$
N-A − N-∅	$-0.0066\ [-0.0470, 0.0338]$	$-0.0034\ [-0.0422, 0.0355]$	$-0.0032\ [-0.0061, -0.0004]$

The early PPL rebound, shared by all three readouts in Figure 6, is absent for N-∅ with floating-point backward under either update backend, which traces it to backward quantization rather than to the exposed affine.

Endpoint gaps and gradient mismatch. Replacing 6-bit pulsed backward reads with floating-point backward reads reduces the analog gap. Both the ideal-replay gap, which reflects the training trajectory, and the direct readout-penalty gap decrease (Table 26).

Under both pulsed backward precisions, the finite-scale penalty response is positive and the supplied penalty slope negative at every late measurement (Figure 17(b)). Training with 6-bit backward produces a larger mismatch than training with floating-point backward (Table 26). These measurements compare the checkpoints reached under each training condition; they do not isolate an immediate effect of switching backward precision at fixed weights. Endpoint gain is nearly unchanged, while floating-point backward reduces logit perturbation and uses more ADC levels. The smaller mismatch persists without a detected NA-∅-minus-N-∅ endpoint CE gap, so mismatch alone does not determine accumulated training damage.

Across all three conditions, N-A lowers the readout penalty relative to NA-∅ (Table 25). Its endpoint CE improvement is detected with 6-bit pulsed backward and with exact updates and floating-point backward, but remains unresolved with pulsed updates and floating-point backward. No endpoint CE improvement over N-∅ is detected in any condition.

Table 26: **Changes in the training gap and readout diagnostics with backward precision.** (a) Pulsed 6-bit-minus-float differences with paired 95% t_4 intervals. (b) Exposed-readout endpoint means; sign counts use five late checkpoints per seed.

(a) Paired changes with backward precision

Quantity	Mean [95% CI]
Analog CE gap	$0.5829\ [0.4262, 0.7397]$
Ideal-replay CE gap	$0.3059\ [0.1259, 0.4858]$
Readout-penalty gap	$0.2771\ [0.1632, 0.3909]$
Exposed-readout radial mismatch	$1.9003\ [1.7581, 2.0426]$

(b) Exposed-readout diagnostics

Quantity	6-bit backward in/out	Float backward in/out
Opposed signs	25/25	25/25
Endpoint γ_{rms}	1.271	1.275
Relative corruption ρ	0.209	0.067
ADC levels per row	5.19	14.98

Forward-precision control. With backward held float, 6-bit forward reads retain small direct penalty gaps under both update backends, while neither total analog gap is detected (Table 25). Disabling forward quantization makes both penalty gaps numerically negligible and both total gaps small (Table 27). The exact-update total gap is slightly negative, with an interval excluding zero. The backend-by-forward interaction on total CE is not detected. Forward quantization is therefore necessary for a direct readout penalty in these controls. With floating-point backward, it does not reproduce the large pulsed endpoint gap. The backward-precision comparison supports trajectory amplification from backward quantization at this operating point.

Table 27: **Forward precision with floating-point backward.** Values are paired exposed-minus-fixed CE differences [95% t_4 CI] over five seeds. The interaction is the pulsed-minus-exact difference in the 6-bit-minus-unquantized effect. Numerically negligible values are rounded to zero.

Condition	Analog CE gap	Readout-penalty gap
Pulsed, unquantized forward	$-0.0035\ [-0.0081, 0.0012]$	$0.0000\ [0.0000, 0.0000]$
Exact, unquantized forward	$-0.0020\ [-0.0036, -0.0003]$	$0.0000\ [0.0000, 0.0000]$
Backend × forward	$-0.0802\ [-0.1656, 0.0053]$	$-0.0058\ [-0.0094, -0.0022]$

Backward I/O localization. A final frozen pulsed 2×2 separates 6-bit versus float quantization on the backward MVM input and output while holding the forward read at 6-bit. Table 28 localizes the detected amplification to backward-output quantization. The input main effect and interaction are unresolved. Late radial penalty signals oppose the finite-scale response at nearly every measurement in all four conditions.

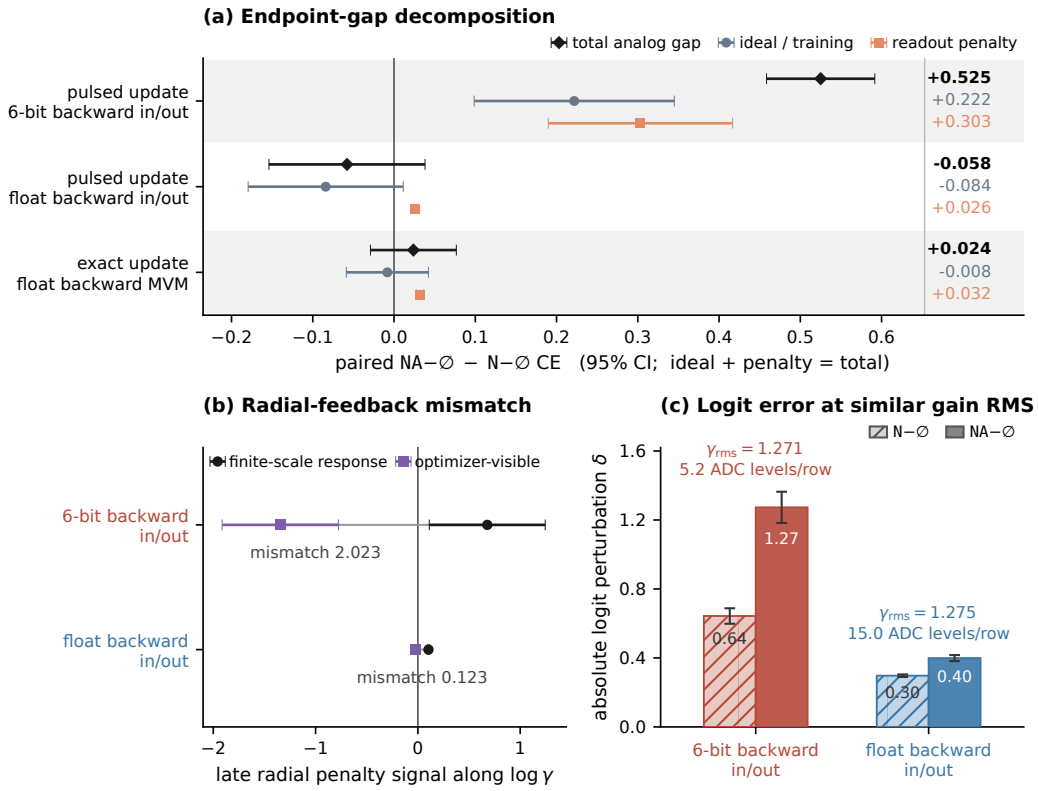


Figure 17: **Backward precision and readout damage.** (a) Paired NA-∅-minus-N-∅ endpoint CE and its ideal-replay and readout-penalty components, with 95% t intervals. Some penalty intervals are narrower than the markers. (b) Late finite-scale and surrogate penalty directions; connectors span their mismatch. (c) Endpoint absolute logit perturbation. Forward reads remain 6-bit throughout.

Table 28: **Backward input/output precision factorial.** Analog-CE gaps are NA-∅ minus N-∅, with paired 95% t_4 intervals over five seeds. Main effects enable 6-bit quantization; counts use five late checkpoints per seed.

Backward input / output	Analog CE gap [95% CI]	Opposed signs
6-bit / 6-bit	$+0.5251$ $[0.4584, 0.5917]$	25/25
6-bit / float	-0.0014 $[-0.1164, 0.1136]$	24/25
Float / 6-bit	$+0.5113$ $[0.4011, 0.6215]$	23/25
Float / float	-0.0579 $[-0.1539, 0.0382]$	25/25
Input main effect	$+0.0351$ $[-0.1096, 0.1798]$	–
Output main effect	$+0.5478$ $[0.4746, 0.6210]$	–
Input × output	-0.0427 $[-0.2159, 0.1305]$	–

Pulsed-update nonideality controls. The two factors here are the device properties that analog-training theory identifies as sources of inexact convergence (Wu et al., 2025; Li et al., 2026b). A comparison crossing pulse asymmetry (`up_down`) with pulse granularity (`dw_min`) uses the levels in Table 29, with all other settings unchanged from the pulsed 6-bit-backward comparison above. It retains a large gap even with zero asymmetry and fine-grained pulses (Table 30). Neither main effect nor their interaction is detected. An exploratory residual-device factorial likewise does not detect a cycle-to-cycle-noise main effect; device-to-device variation increases the total gap, primarily through the ideal trajectory. A deterministic ideal-pulsed cell still retains a positive gap, primarily through the readout penalty. The residual-device comparison fixes zero mean asymmetry and the finer pulse size,

then crosses cycle-to-cycle noise with device-to-device variation. The device-to-device parameters in Table 29 are switched together; the deterministic cell sets both noise and variation to zero.

Table 29: Device-parameter levels in the pulsed-update controls.

Factor	AIHWKit parameter	Levels
Mean pulse asymmetry	`up_down`	0, 0.1
Pulse granularity	`dw_min`	0.001, 0.01
Cycle-to-cycle noise	`dw_min_std`	0, 0.3
Device-to-device variation	`dw_min_dtod`	0, 0.3
	`w_min_dtod`, `w_max_dtod`	0, 0.3
	`up_down_dtod`	0, 0.01

Table 30: **Pulsed-update nonideality controls.** Values are paired mean effects [95% t_4 CI] over five seeds. Cell gaps are exposed minus fixed; factorial effects measure changes in that gap. Main effects add asymmetry, cycle-to-cycle noise, or device-to-device variation, or reduce pulse size.

Cell or effect	Analog CE	Ideal CE	Readout penalty
Zero-asymmetry, fine-grain cell	$+0.6813$ $[0.2251, 1.1374]$	$+0.3795$ $[0.2656, 0.4934]$	$+0.3018$ $[-0.0935, 0.6970]$
Pulse-asymmetry main effect	-0.0664 $[-0.4035, 0.2708]$	-0.0745 $[-0.3355, 0.1866]$	$+0.0081$ $[-0.2497, 0.2659]$
Pulse-granularity main effect	$+0.0898$ $[-0.0795, 0.2591]$	$+0.0833$ $[-0.0651, 0.2317]$	$+0.0065$ $[-0.1652, 0.1783]$
Asymmetry $\times$ granularity	-0.0450 $[-0.6672, 0.5772]$	$+0.0365$ $[-0.2002, 0.2731]$	-0.0815 $[-0.6099, 0.4469]$
Cycle-to-cycle noise main effect	$+0.0625$ $[-0.3779, 0.5028]$	$+0.0566$ $[-0.1792, 0.2924]$	$+0.0059$ $[-0.3009, 0.3126]$
Device-to-device variation main effect	$+0.2814$ $[0.0193, 0.5434]$	$+0.2682$ $[0.1652, 0.3711]$	$+0.0132$ $[-0.2480, 0.2744]$
Cycle $\times$ device interaction	$+0.2825$ $[-0.5175, 1.0825]$	$+0.1762$ $[-0.2855, 0.6380]$	$+0.1063$ $[-0.2890, 0.5015]$
Deterministic ideal-pulsed cell	$+0.3374$ $[0.1341, 0.5408]$	$+0.0548$ $[-0.1690, 0.2785]$	$+0.2827$ $[0.0952, 0.4701]$

F.2 Gradient mismatch and endpoint loss across optimizers

The study uses enwik8 d128-L4 with 6-bit forward reads, floating-point backward MVMs, exact updates, and matched analog/ideal evaluation over five paired seeds (0–4). In the *final-affine-only* scope, AdamW, SGD with momentum, Lion (Chen et al., 2023), or full-matrix AdaGrad (Duchi et al., 2011) updates the final gain and bias; all other parameters retain the background AdamW schedule. AdaGrad's preconditioner spans the d gain coordinates.

In the *training-recipe* scope, AdamW, SGD-momentum, or Lion updates the whole model. For Muon+AuxAdamW (Liu et al., 2025) and Shampoo+AuxAdamW (Gupta et al., 2018), Muon or Shampoo updates eligible hidden attention/MLP matrices, while AuxAdamW updates the final affine, other normalization parameters, embeddings, heads, and biases. Within each scope, we compare `NA-∅` and `N-∅` under identical parameter routing.

All runs use 5,000 steps, 400 warmup steps, cosine decay to one tenth of the peak learning rate, gradient clipping at norm one, and zero weight decay. Table 31 gives settings selected on digital-only pilot seeds 5–7 before inspecting analog outcomes. The scopes can differ in both parameter routing and selected learning rate, so their results are reported separately.

For each method, the paired estimand Δ_{opt} is final analog validation CE for `NA-∅` minus that for `N-∅`. Table 32 decomposes the contrast into ideal CE and the realized readout loss penalty D_{ro} of Eq. (22).

Table 32 shows detected analog-CE increases for final-affine-only AdamW and Lion, and for training-recipe AdamW, SGD-momentum, Muon+AuxAdamW, and Shampoo+AuxAdamW. Training-recipe AdamW uses peak learning rate 10^{-3}, above the primary pulsed setting's 3×10^{-4}. The remaining settings have no detected endpoint gap. In the settings with detected damage, the mean readout-penalty component is larger than the ideal-CE component; this decomposition does not rule out a contribution from the learned representation and head weights. Final-affine-only SGD-momentum and AdaGrad have smaller analog-CE gaps than AdamW (Table 33).

The penalty-slope mismatch is more consistent across settings than endpoint damage. At five late checkpoints (steps 3000–4999), the finite-scale penalty response is positive and opposes the supplied penalty slope at nearly every measurement, including settings with near-unit gain and no detected endpoint gap (Table 33).

Table 31: **Optimizer routing and frozen settings.** "LR" is the learning rate; the peak rate applies to the parameters targeted by each scope.

method	peak LR	all other parameters	method-specific settings
final-affine-only scope: the named optimizer updates $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$			
AdamW	10^{-3}	AdamW 3×10^{-4}	$\beta = (0.9, 0.95)$
SGD-momentum	10^{-3}	AdamW 3×10^{-4}	momentum 0.9
Lion	3×10^{-4}	AdamW 3×10^{-4}	$\beta = (0.9, 0.99)$
full-matrix AdaGrad	10^{-4}	AdamW 3×10^{-4}	$\epsilon = 10^{-6}$
training-recipe scope: the named recipe updates the whole network			
AdamW	10^{-3}	same, 10^{-3}	$\beta = (0.9, 0.95)$
SGD-momentum	10^{-1}	same, 10^{-1}	momentum 0.9
Lion	3×10^{-4}	same, 3×10^{-4}	$\beta = (0.9, 0.99)$
Muon+AuxAdamW	10^{-2}	AuxAdamW 3×10^{-4}	momentum 0.95; five Newton–Schulz steps; Nesterov interpolation; rectangular scaling
Shampoo+AuxAdamW	10^{-2}	AuxAdamW 3×10^{-4}	block 128; inverse fourth roots every 50 steps; $\epsilon = 10^{-6}$; momentum 0.9; SGD-norm graft

Table 32: **Readout contrasts across optimizers.** Differences are `NA-∅` minus `N-∅` over five paired seeds. Analog CE includes 95% t_4 intervals; other columns give means.

optimizer scope	method	exposed γ_{rms}	analog CE Δ [95% CI]	ideal CE Δ	penalty Δ
final-affine-only	AdamW	1.835	$0.1076\ [0.0654, 0.1498]$	0.0164	0.0912
final-affine-only	SGD-momentum	1.014	$-0.0097\ [-0.0361, 0.0167]$	-0.0099	0.0001
final-affine-only	Lion	1.759	$0.0815\ [0.0470, 0.1159]$	-0.0062	0.0877
final-affine-only	full-matrix AdaGrad	1.001	$0.0107\ [-0.0081, 0.0296]$	0.0110	-0.0003
training-recipe	AdamW	1.478	$0.0523\ [0.0060, 0.0987]$	0.0051	0.0472
training-recipe	SGD-momentum	2.409	$4.5633\ [3.7951, 5.3315]$	1.1248	3.4385
training-recipe	Lion	1.016	$0.0353\ [-0.0449, 0.1154]$	0.0358	-0.0005
training-recipe	Muon+AuxAdamW	1.347	$0.0243\ [0.0192, 0.0295]$	0.0010	0.0233
training-recipe	Shampoo+AuxAdamW	1.250	$0.0509\ [0.0125, 0.0893]$	0.0131	0.0378

Table 33: **Optimizer contrasts and late penalty directions.** Contrasts compare each method's exposed-minus-fixed analog-CE gap with AdamW in the same scope (five paired seeds; 95% t_4 intervals). Counts pool five late checkpoints per seed.

Final-affine-only contrast	Mean CE difference [95% CI]
SGD-momentum minus AdamW	$-0.1173\ [-0.1674, -0.0672]$
Full-matrix AdaGrad minus AdamW	$-0.0968\ [-0.1455, -0.0482]$

Scope	Positive finite-scale response	Opposed surrogate sign
Training-recipe Lion	25/25	23/25
Each other setting	25/25	25/25

Why the sign mismatch need not become endpoint damage. The sign comparison concerns the penalty component of the local direction. Equation (30) gives $\bar{D}_{\mathrm{predicted}} = \bar{D}_{\mathrm{ideal}} + \bar{D}^{\mathrm{vis}}_{\mathrm{pen}}$, so the ideal task term can offset a wrong-signed penalty term. The resulting gradient projection is transformed by the optimizer's learning rate and state, while the other parameters change concurrently. Replacing only the radial penalty projection reduces gain, but its endpoint-CE change is not detected (Table 20). The optimizer sweep and this intervention show why endpoint behavior depends on the task gradient and optimizer as well as the penalty signal. Gain magnitude, channelwise allocation, and model co-adaptation change together in these experiments.

F.3 EXACT-UPDATE TRANSFER IS OPTIMIZER-DEPENDENT

We test gain-gradient replacement with exact updates, floating-point backward MVMs, and different optimizers. For each of six optimizer configurations, five source units supply a common step-0 state to the same six gain-intervention branches defined in Appendix E.7. The AdamW anchor uses learning rate 3×10^{-4}, matching the pulsed intervention's learning-rate schedule, model, forward precision, and intervention settings. The inference backend changes both the update rule and backward treatment; this comparison tests transfer across training configurations rather than isolating update precision. The other five rows use the training-recipe configurations of Table 31; Muon and Shampoo update eligible hidden matrices while AuxAdamW updates the final gain. The design comprises 180 training branches, each trained for 5,000 steps at d128-L4/out6. The primary contrast within each row is supplied-gradient minus penalty-replacement analog CE, with a two-sided paired t_4 interval. The six rows are reported separately without a multiplicity correction.

Table 34: **Gain-gradient replacement under exact updates and floating-point backward.** Means over five paired seeds per recipe. (a) Analog CE and gain RMS. (b) Ideal-CE and penalty components. (c) Control endpoints. Contrasts are supplied-gradient minus replacement, with 95% t_4 intervals.

(a) Primary transfer contrast

recipe	Supplied gain gradient	Replace penalty	Δ CE [95% CI]	γ_{rms} supplied → replace
AdamW anchor	2.2392	2.2195	$+0.01968$ $[+0.00007, +0.03928]$	$1.3386 \rightarrow 1.0008$
AdamW transfer	1.9785	1.9142	$+0.06432$ $[+0.03329, +0.09535]$	$1.4785 \rightarrow 0.9867$
SGD+momentum	6.9414	2.7297	$+4.21163$ $[+3.45164, +4.97161]$	$2.4091 \rightarrow 1.5582$
Lion	3.0815	3.1665	-0.08495 $[-0.14490, -0.02500]$	$1.0164 \rightarrow 0.9948$
Muon+AuxAdamW	1.5942	1.5710	$+0.02319$ $[+0.01453, +0.03185]$	$1.3471 \rightarrow 1.0033$
Shampoo+AuxAdamW	2.8078	2.7829	$+0.02498$ $[-0.01069, +0.06066]$	$1.2501 \rightarrow 0.9919$

(b) Components of the primary contrast

recipe	ideal CE Δ [95% CI]	readout-penalty Δ [95% CI]
AdamW anchor	-0.00849 $[-0.02641, +0.00943]$	$+0.02817$ $[+0.02489, +0.03145]$
AdamW transfer	$+0.01242$ $[-0.01308, +0.03792]$	$+0.05190$ $[+0.04484, +0.05896]$
SGD+momentum	$+0.94718$ $[+0.64029, +1.25407]$	$+3.26445$ $[+2.24134, +4.28755]$
Lion	-0.10988 $[-0.18020, -0.03955]$	$+0.02493$ $[-0.00117, +0.05102]$
Muon+AuxAdamW	$+0.00318$ $[-0.00473, +0.01108]$	$+0.02002$ $[+0.01834, +0.02169]$
Shampoo+AuxAdamW	-0.00261 $[-0.03586, +0.03064]$	$+0.02760$ $[+0.02107, +0.03412]$

(c) Control endpoints (mean analog CE)

recipe	Remove penalty	Negate penalty	Cap gain at one	Fixed gain and bias (N-∅)
AdamW anchor	2.2478	2.2398	2.2304	2.2149
AdamW transfer	1.9492	1.9154	1.9504	1.9251
SGD+momentum	2.7908	2.3989	2.5762	2.3645
Lion	3.0626	3.0337	3.0040	3.0480
Muon+AuxAdamW	1.5870	1.5822	1.5727	1.5684
Shampoo+AuxAdamW	2.7888	2.8070	2.7707	2.7587

Gain-gradient replacement lowers analog CE for the AdamW anchor, AdamW transfer, SGD-momentum, and Muon+AuxAdamW (Table 34). The anchor's interval narrowly excludes zero, and its mean benefit is smaller than in the pulsed configuration. Shampoo remains unresolved, while replacement raises CE under Lion.

The loss decomposition explains this variation. Replacement lowers mean readout penalty in all six configurations, with a detected reduction in five; Lion's penalty contrast is unresolved. Under Lion, worse ideal CE outweighs the smaller mean penalty reduction. Under Shampoo, the detected penalty reduction does not produce a detected net-CE improvement. Thus, correcting gain feedback can improve readout fidelity while changing the training trajectory unfavorably. The control endpoints also show that full-vector replacement does not consistently attain the lowest mean CE across recipes.

G STRUCTURAL READOUT REPAIRS

This appendix explains why post-ADC normalization removes the scale pathway, what the post-ADC affine costs, and whether the design rule holds at a tuned head learning rate and in other architectures.

G.1 POST-ADC NORMALIZATION REMOVES THE SCALE PATHWAY

With a post-ADC operation, Eq. (2) sends $Q(\tilde{z})$ to the loss, and its matched ideal replay sends $Q(z^\star)$. Thus, the relevant perturbation is $\epsilon_Q := Q(z^\star + \epsilon) - Q(z^\star)$, not the pre-$Q$ error alone. For an affine $Q(z) = \boldsymbol{s} \odot z + \boldsymbol{u}$,

$$\epsilon_Q = \boldsymbol{s} \odot \epsilon. \tag{39}$$

The shared bias cancels, and fixed output gains do not remove the dependence on the pre-DAC scale.

Parameter-free normalization instead satisfies $\mathtt{N}(cz) = \mathtt{N}(z)$ for $c > 0$, up to its fixed numerical stabilizer. Under the uniform rescaling in Eq. (14), both ideal and digitized logits acquire the same factor c, so normalizing either removes it. Any affine after this normalization preserves that input-scale invariance. Ignoring the numerical stabilizer, let $C := I - \mathbf{1}\mathbf{1}^\top/V$, $v := Cz^\star$, and $\sigma := (v^\top v/V)^{1/2} > 0$. Since $\mathtt{N}(z^\star) = v/\sigma$, a perturbation ϵ gives

$$\mathrm{d}v = C\epsilon, \qquad \mathrm{d}\sigma = \frac{v^\top \epsilon}{V\sigma}.$$

Differentiating the quotient therefore yields

$$\epsilon_{\mathtt{N}} \approx J_{\mathtt{N}}(z^\star)\epsilon = \frac{1}{\sigma} \underbrace{\left(C - \frac{vv^\top}{V\sigma^2}\right)}_{\Pi} \epsilon, \tag{40}$$

where σ is the ideal logits' standard deviation. The projection removes the common-shift and centered-logit directions, leaving an error measured relative to the ideal-logit standard deviation.

A normalizing Q removes the uniform-scale pathway for either P row; an affine-only Q does not.

This invariance also constrains task fit. Without an output affine, the normalized logits have unit standard deviation (ignoring the stabilizer), so uniform rescaling of the incoming logits cannot adjust softmax temperature. A learned affine after normalization restores adjustable output scale and bias while preserving input-scale invariance. This constraint provides a possible explanation for the higher ideal-head CE of `NA-N` on our vision benchmarks (Appendix H.4), although those comparisons do not isolate temperature restriction as the cause.

G.2 POST-ADC AFFINE CAPACITY

The P-Q design rule determines which operations act before and after the converters. Within the tested $P = \mathtt{N}$, $Q = \mathtt{A}$ configuration, a separate training-time question is whether the digitized logits need offsets only or both per-logit scale and offset. The experiments below train these alternatives from scratch.

For $P = \mathtt{N}$, output scale and offset define three nested readouts:

$$Q_0(z) := z, \qquad Q_{\mathrm{unit}}(z) := \mathtt{A}_{\mathbf{1},\boldsymbol{u}}(z) = z + \boldsymbol{u}, \qquad Q_{\mathrm{full}}(z) := \mathtt{A}_{\boldsymbol{s},\boldsymbol{u}}(z) = \boldsymbol{s} \odot z + \boldsymbol{u}. \tag{41}$$

The improvement criterion requires the paired 95% CE-difference interval to lie below zero; non-inferiority requires its upper endpoint to be at most $+0.01$ CE. A five-seed byte-level d128-L4/out6 experiment trains these readouts together with two $P = \mathtt{NA}$ controls (Table 35). The paired unit-minus-no-affine comparison establishes neither improvement nor non-inferiority.

At the shared learning rate, neither offsets alone nor a full affine yield a detected analog-CE benefit over Q_0. At a tenfold head rate, however, the full affine lowers CE relative to `N`-$\varnothing$, with a further reduction when normalization precedes it (Appendix G.4).

We also train the three nested $P = \mathtt{N}$ readouts at V32k for 15,000 updates with five paired seeds. Table 36 reports their endpoints and contrasts. The full affine raises analog CE relative to Q_0. Its smaller readout penalty is outweighed by higher ideal CE. Neither the full-versus-unit comparison nor the unit-versus-none comparison has a detected analog-CE difference. Thus, the added post-ADC capacity does not consistently lower task loss across the tested settings.

Table 35: **Post-ADC affine capacity at the primary setting.** Endpoint means and paired CE differences with 95% t_4 intervals over five seeds. Unit and full refer to Eq. (41).

condition	analog CE	ideal CE	penalty
NA-∅ control	3.67875	3.23988	0.43887
N-∅	3.09279	2.93385	0.15894
N-A (unit)	3.07247	2.90013	0.17235
N-A (full)	3.07195	2.93661	0.13534
NA-A (unit)	3.59127	3.15332	0.43795

contrast	analog CE	ideal CE	readout penalty
unit affine − no affine ($P = \mathtt{N}$)	-0.02032 $[-0.09713, 0.05650]$	-0.03373 $[-0.08316, 0.01571]$	$+0.01341$ $[-0.02812, 0.05494]$
full affine − no affine ($P = \mathtt{N}$)	-0.02084 $[-0.11884, 0.07717]$	$+0.00276$ $[-0.07917, 0.08470]$	-0.02360 $[-0.05800, 0.01081]$
unit − full affine ($P = \mathtt{N}$)	$+0.00052$ $[-0.10729, 0.10833]$	-0.03649 $[-0.14147, 0.06850]$	$+0.03701$ $[0.00394, 0.07007]$
unit affine − no affine ($P = \mathtt{NA}$)	-0.08749 $[-0.21365, 0.03868]$	-0.08657 $[-0.20678, 0.03364]$	-0.00092 $[-0.18700, 0.18517]$
NA-∅ − N-∅	$+0.58596$ $[0.42242, 0.74951]$	$+0.30603$ $[0.14207, 0.47000]$	$+0.27993$ $[0.10287, 0.45699]$

Table 36: **Nested post-ADC affine capacity at V32k.** Five-seed mean endpoints and paired CE differences (left minus right) with 95% t_4 intervals, using the readouts of Eq. (41).

Setting	Q_0 analog CE	Q_{unit} analog CE	Q_{full} analog CE
V32k	5.74838	5.78799	5.87101

setting	contrast	analog CE	ideal CE	readout penalty
V32k	unit − none	$+0.03961$ $[-0.02894, 0.10817]$	$+0.04202$ $[-0.02666, 0.11070]$	-0.00241 $[-0.00390, -0.00091]$
	full − unit	$+0.08302$ $[-0.01578, 0.18181]$	$+0.08607$ $[-0.01302, 0.18516]$	-0.00305 $[-0.00651, 0.00040]$
	full − none	$+0.12263$ $[0.03443, 0.21083]$	$+0.12809$ $[0.04194, 0.21424]$	-0.00546 $[-0.00952, -0.00140]$

G.3 Arithmetic and parameter cost

Table 37 gives arithmetic and parameter counts for a bias-free $V \times d$ head and its post-ADC affine. The full affine adds $1/d$ as many multiplications and $2/d$ as many parameters as the head, independently of vocabulary size. These counts do not estimate energy or latency overhead, which also depends on digital circuitry and parameter traffic.

Table 37: **Readout arithmetic and parameter cost.** Operations are per prediction item; head additions count dot-product sums. Post-ADC operations are digital.

Component	Parameters	Multiplications	Additions
Head	Vd	Vd	$V(d-1)$
Offset-only post-ADC map	V	0	V
Full post-ADC affine	$2V$	V	V

G.4 Readout comparisons at a tuned head learning rate

The main comparisons give every parameter, including the analog head, one shared learning rate (Appendix A). This section tests whether that choice produces the failure or the design rule. Each run multiplies only the head tile's learning rate by a fixed factor at every step, keeping the warmup and cosine shape, while the normalization gain and bias and all other parameters keep the base rate. All runs use the primary d128-L4 setting with 6-bit converters, 5,000 steps, and ten paired seeds with chip identity equal to seed. The sweep reports all tested multipliers. The grid uses the tenfold rate, selected for its lowest mean CE in the sweep. Intervals are unadjusted two-sided 95% paired t intervals.

A larger head step improves ideal-head performance and readout fidelity. Raising the head's learning rate lowers both ideal-head CE and the readout penalty (Table 38). Most of the total CE improvement remains under ideal-head replay. The benefit levels off between tenfold and twentyfold.

Table 38: N-∅ **across head learning-rate multipliers.** Ten-seed endpoint means; changes from the shared rate have paired 95% t_9 intervals.

family	head rate	analog CE	ideal CE	penalty	change from 1× [95% CI]
LayerNorm	1×	3.1431	2.9637	0.1795	–
	2×	2.9066	2.7917	0.1149	-0.2365 $[-0.3075, -0.1655]$
	5×	2.7128	2.6456	0.0672	-0.4303 $[-0.5088, -0.3518]$
	10×	2.6591	2.6053	0.0538	-0.4840 $[-0.5520, -0.4161]$
	20×	2.6884	2.6340	0.0544	-0.4547 $[-0.5396, -0.3698]$
RMSNorm	1×	3.1684	2.9807	0.1876	–
	2×	2.9173	2.8037	0.1136	-0.2511 $[-0.3606, -0.1416]$
	5×	2.7128	2.6481	0.0647	-0.4556 $[-0.5479, -0.3633]$
	10×	2.6392	2.5835	0.0558	-0.5292 $[-0.6361, -0.4223]$
	20×	2.6782	2.6217	0.0565	-0.4902 $[-0.5859, -0.3944]$

The exposed affine still degrades training at a tuned head step. The gap between NA-∅ and N-∅ stays positive and is detected at every rate in both normalization families (Table 39). The mean gap is smaller at the raised rates. Most of the remaining gap is readout penalty. The penalty of N-∅ falls with the head's rate, whereas the exposed readout keeps a large penalty and its gain grows further.

Table 39: **Exposed-affine gap across head learning rates.** Endpoint means over ten seeds; the analog-CE gap is NA-∅ minus N-∅ with a paired 95% t_9 interval.

family	head rate	N-∅ CE	NA-∅ CE	gap [95% CI]	N-∅ penalty	NA-∅ penalty	γ_{rms}
LayerNorm	1×	3.1431	3.6406	$+0.4975$ $[+0.3320, +0.6630]$	0.1795	0.4816	1.258
	5×	2.7128	3.1802	$+0.4673$ $[+0.4078, +0.5269]$	0.0672	0.3798	1.557
	10×	2.6591	3.0543	$+0.3952$ $[+0.3238, +0.4666]$	0.0538	0.3050	1.533
RMSNorm	1×	3.1684	3.6357	$+0.4673$ $[+0.2886, +0.6459]$	0.1876	0.5071	1.258
	5×	2.7128	3.1311	$+0.4183$ $[+0.3715, +0.4652]$	0.0647	0.3447	1.585
	10×	2.6392	2.9557	$+0.3164$ $[+0.2605, +0.3723]$	0.0558	0.2470	1.583

Readout placements at a tenfold head rate. We train the eight BP LayerNorm configurations of Table 1 at the tenfold rate (Table 40). Every post-ADC operation still lowers analog CE relative to NA-∅. At this rate, NA-NA and N-NA have the lowest mean CE, a post-ADC affine improves on N-∅, N-N has no detected difference from N-∅, and normalization alone after an exposed affine (NA-N) stays above N-∅. Comparisons pair seed/chip units, and each endpoint's analog and ideal losses use matched evaluation batches. Ten independent held-out seeds (10–19) confirm the exploratory N-NA-minus-N-A contrast from the grid (seeds 0–9). At the shared head rate, the main grid detects no difference between these readouts, so the additional benefit is established at the tuned rate in this setting.

Scope. The additional post-ADC benefits reported here are supported at the tested tenfold head rate. These comparisons cover BP at the primary operating point, the LayerNorm grid, and the simulated device of Appendix A.2, whose weight-scaling remapping is off. The local cross-entropy, vision, large-vocabulary, and 39.4M comparisons use the shared learning rate; the effect of raising the head rate in those settings remains untested.

Table 40: **The BP LayerNorm readout grid at a tenfold head rate.** Ten-seed endpoint means and paired CE differences with 95% t_9 intervals. The held-out row uses ten independent seeds.

configuration	analog CE	ideal CE	penalty
N-∅	2.6714	2.6151	0.0563
N-N	2.6744	2.6548	0.0196
N-A	2.6311	2.5918	0.0393
N-NA	2.5731	2.5557	0.0174
NA-∅	3.0543	2.7492	0.3050
NA-N	2.7144	2.6887	0.0257
NA-A	2.8169	2.6721	0.1448
NA-NA	2.5675	2.5472	0.0203

contrast	mean difference [95% CI]
NA-N − NA-∅	-0.3399 $[-0.4197, -0.2600]$
NA-NA − NA-∅	-0.4868 $[-0.5609, -0.4126]$
NA-A − NA-∅	-0.2374 $[-0.3222, -0.1526]$
NA-N − N-∅	$+0.0429$ $[+0.0187, +0.0672]$
NA-NA − N-∅	-0.1039 $[-0.1162, -0.0916]$
NA-A − N-∅	$+0.1454$ $[+0.0996, +0.1913]$
N-A − N-∅	-0.0403 $[-0.0697, -0.0110]$
N-NA − N-∅	-0.0983 $[-0.1295, -0.0672]$
N-N − N-∅	$+0.0029$ $[-0.0225, +0.0283]$
NA-∅ − N-∅	$+0.3828$ $[+0.3127, +0.4529]$
N-NA − N-A (exploratory)	-0.0580 $[-0.0892, -0.0268]$
N-NA − N-A (held-out)	-0.0645 $[-0.0951, -0.0339]$

G.5 Training-time gain migration

This experiment uses the primary d128-L4 setting, 6-bit converters, 5,000 training steps, and ten paired seeds at the shared head learning rate. Outlier Suppression migrates a normalization gain into the following weight matrix after training (Section 2). We repeat the fold after every optimizer step or every 100 steps, resetting the gain to one and clearing its optimizer moments at each fold. Every-step resetting removes gain amplification from subsequent forward inputs; at the longer interval, the gain can grow between folds. In RMSNorm, exact readback and an exact weight write preserve the ideal readout function. In LayerNorm, the bias stays before the DAC, so folding only the gain also changes the ideal function. The device-path version instead adds the fold to the optimizer's pulse-quantized transfer buffer. Transfer delays and residuals prevent immediate function preservation after the gain reset. A further version estimates the weights through analog forward reads and least squares before computing the device-path fold.

Every folded arm lowers analog CE relative to NA-∅ in both families (Table 41). Migration changes gain exposure, head weights, and optimizer state together, so these comparisons do not separate their contributions to the benefit. The exposed LayerNorm bias does not prevent repair in these runs.

G.6 Architectural scope of final normalization

The main co-design map holds P to a normalization family because the tested Transformer natively places a final LayerNorm with trainable gain and bias before its bias-free language-model head. Two separate studies compare this normalized readout with the unnormalized ∅–∅ reference. On enwik8, ∅-∅ removes the final normalization for BP or every local-readout normalizer for LCE. All other d128-L4 out6 settings and the five chip/seed identities match the N-∅ controls. On CIFAR-100, a standard CIFAR-adapted ResNet-18 uses a 3×3 stem, BatchNorm inside its residual trunk, global average pooling, and a biased $512 \to 100$ classifier. The trunk remains digital, and only the final classifier is mapped to an out6 `InferenceRPUConfig` tile. Its native bias is programmed on the tile together with the classifier weights (`digital_bias=false`), so no post-ADC bias-only operator is present. A deterministic class-balanced 45k/5k split of official train is used for 200 fixed epochs. Official test is untouched.

Table 41: **Training-time gain migration at the primary setting.** Seed means of final analog CE over ten paired seeds, with paired differences from NA-∅ and N-∅. The fold is written exactly or through the optimizer's pulse-quantized transfer buffer (device path), and it is computed from an exact copy of the head's weights or from a measured readout. The interval gives the optimizer steps between folds.

family	fold write	fold read	interval	analog CE	minus NA-∅	minus N-∅
LayerNorm	NA-∅ (exposed)			3.6406	–	–
	N-∅ (gain fixed)			3.1431	–	–
	exact	exact	1	3.0399	-0.6007 ± 0.0605	-0.1032 ± 0.0422
	exact	exact	100	2.9594	-0.6811 ± 0.0618	-0.1837 ± 0.0380
	device path	exact	1	3.0538	-0.5868 ± 0.0574	-0.0893 ± 0.0496
	device path	exact	100	2.9760	-0.6645 ± 0.0609	-0.1671 ± 0.0445
	device path	measured	1	3.0208	-0.6197 ± 0.0650	-0.1223 ± 0.0368
	device path	measured	100	2.9595	-0.6811 ± 0.0689	-0.1836 ± 0.0511
RMSNorm	NA-∅ (exposed)			3.6357	–	–
	N-∅ (gain fixed)			3.1684	–	–
	exact	exact	1	3.0354	-0.6003 ± 0.0749	-0.1330 ± 0.0475
	exact	exact	100	2.9512	-0.6845 ± 0.0861	-0.2172 ± 0.0557
	device path	exact	1	2.9943	-0.6413 ± 0.0731	-0.1740 ± 0.0505
	device path	exact	100	2.9578	-0.6779 ± 0.0729	-0.2106 ± 0.0471
	device path	measured	1	3.0163	-0.6194 ± 0.0828	-0.1521 ± 0.0518
	device path	measured	100	2.9522	-0.6835 ± 0.0742	-0.2162 ± 0.0478

Table 42: **Final-normalization boundary.** Paired final CE differences with 95% Student-t intervals (five seeds). "Ideal" replays the trained classifier without converter error; "penalty" is analog minus ideal. Each row gives ∅-∅ minus N-∅ within one experiment.

experiment	analog CE	ideal CE	readout penalty
enwik8 BP	$+2.5675\ [1.7710, 3.3639]$	$+2.3220\ [0.3419, 4.3021]$	$+0.2455\ [-1.1420, 1.6329]$
enwik8 LCE	$+7.3832\ [-2.3637, 17.1301]$	$+6.1072\ [-4.4761, 16.6906]$	$+1.2760\ [-0.7433, 3.2953]$
CIFAR-100 ResNet-18 BP	$-0.2068\ [-0.2214, -0.1921]$	$-0.2022\ [-0.2199, -0.1845]$	$-0.0045\ [-0.0097, 0.0006]$

For Transformer BP, removing the final normalization increases analog CE, and most of the mean degradation persists under ideal-head replay (Table 42). This supports retaining parameter-free normalization when removing the final affine in this architecture. LCE has a positive mean difference but high variance, with one strongly diverging seed; its interval includes zero and does not establish non-inferiority at the $+0.01$ CE margin.

For ResNet-18, inserting parameter-free normalization raises CE relative to the native GAP–linear head (Table 43). Most of the native head's CE advantage is already present under ideal-head replay; the readout-penalty difference remains unresolved. An additional test inserts a non-native LayerNorm with trainable affine parameters before the DAC, and NA-∅ is worse than ∅-∅ in both analog CE and readout penalty. Post-ADC normalization improves analog CE relative to NA-∅, but NA-N nevertheless remains worse than the untouched GAP–linear head.

Table 43: ResNet-18 readout controls on CIFAR-100 validation. Endpoints are mean ± SD over five seeds; paired contrasts have two-sided 95% Student-t intervals.

Configuration	Analog CE	Accuracy (%)
Native ∅-∅	1.0293 ± 0.0260	75.54 ± 0.53
Inserted N-∅	1.2360 ± 0.0215	74.92 ± 0.36

Paired contrast	Mean	95% CI
NA-∅ minus ∅-∅, analog CE	$+0.9241$	$[0.8789, 0.9693]$
NA-∅ minus ∅-∅, readout penalty	$+0.1268$	$[0.1053, 0.1482]$
NA-N minus NA-∅, analog CE	-0.6377	$[-0.6683, -0.6072]$

H Model, Task, and Converter Extensions

H.1 Width and depth

The width/depth sweep trains each configuration for 5,000 steps with six-bit converters and five paired seeds (Table 44). Readout configurations share a chip within each model and seed. The exposed affine raises PPL at every tested width and depth. A 15,000-step d256-L4 comparison evaluates all six readout configurations (Table 45).

Table 44: Width and depth comparisons after 5,000 steps. Paired PPL differences are means with 95% t_4 intervals over five seeds.

Model	Parameters	NA-∅ − N-∅	NA-∅ + γ-clamp − N-∅
d64-L4	0.20M	$+5.09\ [2.34, 7.84]$	$+0.53\ [-1.24, 2.30]$
d128-L2	0.39M	$+13.62\ [9.93, 17.30]$	$+1.64\ [-0.14, 3.41]$
d128-L4	0.79M	$+14.53\ [5.87, 23.20]$	$+1.05\ [-1.78, 3.88]$
d128-L6	1.18M	$+10.91\ [7.52, 14.31]$	$-1.94\ [-5.40, 1.52]$
d256-L4	3.15M	$+26.87\ [9.46, 44.27]$	$+5.85\ [-9.13, 20.83]$

Table 45: Readout placement in d256-L4 after 15,000 steps. Analog PPL is mean ± SD over five seeds.

Readout	N-∅	NA-∅	NA-∅ + γ-clamp	N-A	NA-N	NA-NA
Analog PPL	22.14 ± 2.32	65.04 ± 36.67	23.90 ± 2.07	13.03 ± 0.76	17.20 ± 0.83	13.62 ± 1.36

H.2 The 39.4M scale-up

Protocol. The six configurations use d512-L12 with eight heads, context length 256, 39.4M parameters, byte vocabulary, and 20,000 training steps on seeds 5–14. All runs use six-bit converters, pulsed updates, and the same learning-rate schedule. Within each seed, the configurations share initial trunk/head weights, fabricated device parameters, and the training data stream; cycle-to-cycle pulse-noise draws are not shared. Analog and ideal-head evaluation use matched batches. We report the final checkpoint of every run.

The primary endpoint is analog PPL (Table 46). The four clamp/post-ADC configurations are each compared with both N-∅ and NA-∅, forming eight prespecified paired tests with Holm correction. CE and the comparisons among post-ADC configurations are secondary; confidence intervals are nominal 95% t_9 intervals. The separate NA-∅−N-∅ comparison uses a sample size chosen after inspecting interim outcomes; its intervals and tests are nominal and do not adjust for this selection or repeated inspection.

Table 46: **Readout configurations at 39.4M parameters.** Ten-seed endpoints after 20,000 steps; PPL is mean ± SD and other columns are means.

config	analog PPL	analog CE	ideal CE	penalty CE	γ_{rms}	h_{rms}
NA-∅	350.81 ± 382.50	5.4006	3.8192	1.5815	1.3477	2.0335
N-∅	39.48 ± 6.97	3.6631	3.0594	0.6036	1.0000	1.0000
NA-∅ + γ-clamp	39.15 ± 4.20	3.6621	3.1246	0.5376	0.9942	1.2549
N-A	14.18 ± 0.52	2.6511	2.6270	0.0241	1.0000	1.0000
NA-N	17.55 ± 1.01	2.8636	2.8286	0.0350	2.3270	3.0073
NA-NA	14.49 ± 0.70	2.6724	2.6493	0.0231	2.4456	3.3219

The N-A, NA-N, and NA-NA configurations improve PPL over N-∅ after Holm correction (Table 47). Thus, removing the exposed affine does not exhaust the available repair at this scale. The N-A and NA-NA endpoints are not statistically distinguished, while the secondary comparison favors NA-NA over NA-N.

The exposed NA-∅ configuration has higher PPL than N-∅ in every paired seed, with a nominal paired interval above zero. Each clamp/post-ADC configuration also improves on NA-∅ in every seed. Their PPL comparisons with NA-∅ pass nominal tests but not the eight-test Holm correction; the secondary CE intervals exclude zero. The large PPL variance of NA-∅ limits precision despite the consistent direction of these differences.

The post-ADC-normalized configurations retain larger pre-DAC gain and input RMS while achieving lower loss, consistent with normalization blocking uniform-scale amplification. Clamping the gain yields no detected difference from N-∅ in either PPL or CE.

Table 47: **Paired placement effects at 39.4M parameters.** Differences are left minus right with nominal 95% t_9 intervals. Holm correction covers the eight primary PPL comparisons above the divider.

contrast	ΔPPL [95% CI]	Holm p_{PPL}	ΔCE [95% CI]
NA-∅ + γ-clamp − N-∅	-0.33 $[-4.45, 3.79]$	0.861	-0.0009 $[-0.1019, 0.1001]$
NA-∅ + γ-clamp − NA-∅	-311.66 $[-584.65, -38.67]$	0.107	-1.7385 $[-2.4321, -1.0449]$
N-A − N-∅	-25.30 $[-30.39, -20.21]$	< 0.001	-1.0120 $[-1.1370, -0.8870]$
N-A − NA-∅	-336.63 $[-610.49, -62.78]$	0.107	-2.7495 $[-3.4570, -2.0421]$
NA-N − N-∅	-21.93 $[-26.29, -17.57]$	< 0.001	-0.7995 $[-0.8854, -0.7135]$
NA-N − NA-∅	-333.26 $[-606.62, -59.90]$	0.107	-2.5370 $[-3.2194, -1.8547]$
NA-NA − N-∅	-24.99 $[-29.98, -19.99]$	< 0.001	-0.9907 $[-1.1121, -0.8694]$
NA-NA − NA-∅	-336.32 $[-609.81, -62.84]$	0.107	-2.7283 $[-3.4155, -2.0411]$
NA-NA − NA-N	-3.06 $[-3.90, -2.22]$	–	-0.1913 $[-0.2410, -0.1415]$
NA-NA − N-A	$+0.31$ $[-0.38, 1.00]$	–	$+0.0213$ $[-0.0272, 0.0697]$
NA-∅ − N-∅	$+311.33$ $[38.59, 584.08]$	–	$+1.7376$ $[1.0358, 2.4394]$

Bias-free transfer at 39M. We compare gain control and readout placement with true RMSNorm at the same d512-L12, byte-level, out6 operating point (Table 48). All configurations use seeds 0–4 and 15,000 pulsed-update steps with the same training and evaluation recipe. In N-A, the pre-DAC RMSNorm is parameter-free and the post-ADC affine is a trainable per-logit gain without bias. In NA-N, the pre-DAC gain trains and the post-ADC RMSNorm is parameter-free. Hidden blocks retain LayerNorm. Analog and ideal-head evaluations use the same checkpoint and batches, retaining the post-ADC operation in both evaluations.

Table 48: **True-RMSNorm gain control and placement at 39M.** Final five-seed means ± SE. Analog-CE differences are matched by seed, with nominal two-sided 95% t_4 intervals.

configuration	analog CE	ideal CE	readout penalty	γ_{rms}	h_{rms}	δ
N-∅	3.7272 ± 0.1021	3.1846 ± 0.0651	0.5426 ± 0.0833	1.000 ± 0.000	1.000 ± 0.000	1.302 ± 0.099
NA-∅	4.8647 ± 0.2034	4.1164 ± 0.4393	0.7483 ± 0.3759	1.211 ± 0.043	1.630 ± 0.017	2.361 ± 0.266
NA-∅ + γ-clamp	4.1390 ± 0.1944	3.4050 ± 0.1467	0.7340 ± 0.0831	0.991 ± 0.002	0.999 ± 0.000	1.464 ± 0.051
N-A	2.8725 ± 0.0322	2.8015 ± 0.0242	0.0710 ± 0.0084	1.000 ± 0.000	1.000 ± 0.000	0.931 ± 0.049
NA-N	3.3187 ± 0.0211	3.1997 ± 0.0181	0.1189 ± 0.0060	1.819 ± 0.025	1.941 ± 0.034	2.922 ± 0.289

Paired analog-CE contrast	Mean	95% CI
NA-∅ minus N-∅	$+1.13750$	$[0.71628, 1.55873]$
NA-∅ + γ-clamp minus N-∅	$+0.41183$	$[-0.02308, 0.84675]$
N-A minus NA-∅	-1.99216	$[-2.52258, -1.46174]$
NA-N minus NA-∅	-1.54599	$[-2.14759, -0.94439]$

Trainable gain increases analog CE, input RMS, and absolute logit perturbation relative to fixed gain, extending the exposed-gain failure across both normalization family and model scale. The paired clamp-minus-fixed interval includes zero. Clamping restores unit input scale; its endpoint contrast remains unresolved. Both deferring the gain and normalizing after the ADC lower analog CE relative to the exposed trainable-gain readout, extending the two repairs to RMSNorm. Post-ADC normalization lowers the mean readout penalty despite a larger absolute logit perturbation before normalization.

H.3 VOCABULARY AND NORMALIZATION-FAMILY CONTROLS

The 32k-vocabulary comparison uses 15,000 training steps and five paired seeds. The exposed affine raises analog PPL, and gain clamping leaves a residual gap from N-∅ (Table 49). The following bias controls test whether this residual gap persists when $\boldsymbol{\beta}$ is fixed.

Table 49: Readout placement at vocabulary size 32,000. Analog PPL is mean ± SD over five seeds; paired PPL differences from N-∅ are means with 95% t_4 intervals; input RMS and logit perturbations are means.

Configuration	Analog PPL	Paired ΔPPL vs N-∅ [95% CI]
N-∅	327 ± 48	–
NA-∅	3358 ± 300	$+3030.8$ $[2644.4, 3417.2]$
NA-∅ + $\boldsymbol{\gamma}$-clamp	569 ± 41	$+242.2$ $[153.6, 330.8]$
N-A	386 ± 35	$+59.2$ $[-0.2, 118.6]$
NA-N	461 ± 42	$+134.3$ $[63.7, 204.9]$
NA-NA	518 ± 28	$+191.3$ $[117.3, 265.2]$

Configuration	N-∅	NA-∅	NA-∅ + $\boldsymbol{\gamma}$-clamp	N-A
h_{rms}	1.000	3.72	2.284	1.000
δ	0.33	1.88	0.50	0.25
ρ	0.023	0.062	0.015	0.018

Bias-free normalization controls. The V32k LayerNorm and RMSNorm controls use separate paired training cohorts. To isolate the gain contribution, we use a five-seed LayerNorm reference and fix $\beta = 0$ in both the unconstrained-gain and clamped-gain cells. Gain remains trainable in both. RMSNorm removes centering and bias. Its three states use fixed gain, unconstrained trainable gain, and clamped trainable gain, respectively. Table 50 reports only within-study contrasts.

Table 50: V32k final-normalization controls at 15000 steps (mean ± SE, five seeds). Deltas are within-family analog CE relative to that family's fixed-gain cell. LayerNorm and RMSNorm are separate training studies. Bold deltas cross the $|t| > 2.776$ threshold.

family	gain/bias mode	analog CE	ΔCE	γ_{rms}	β_{rms}	h_{rms}
LayerNorm	fixed $\boldsymbol{\gamma} = \mathbf{1}, \boldsymbol{\beta} = \mathbf{0}$	5.7304 ± 0.0175	–	–	–	1.000
LayerNorm	trainable $\boldsymbol{\gamma}, \boldsymbol{\beta}$	7.3588 ± 0.1488	$\mathbf{+1.6284 \pm 0.1390}$	2.806	1.664	3.693
LayerNorm	clamped $\boldsymbol{\gamma}$, trainable $\boldsymbol{\beta}$	6.3848 ± 0.0717	$\mathbf{+0.6544 \pm 0.0638}$	0.997	1.705	2.193
LayerNorm	trainable $\boldsymbol{\gamma}$, fixed $\boldsymbol{\beta} = \mathbf{0}$	6.9864 ± 0.0785	$\mathbf{+1.2560 \pm 0.0899}$	3.034	0	3.049
LayerNorm	clamped $\boldsymbol{\gamma}$, fixed $\boldsymbol{\beta} = \mathbf{0}$	5.7826 ± 0.0420	$+0.0522 \pm 0.0503$	1.000	0	1.000
RMSNorm	fixed $\boldsymbol{\gamma} = \mathbf{1}$	5.7971 ± 0.0233	–	–	–	1.000
RMSNorm	trainable $\boldsymbol{\gamma}$	6.8781 ± 0.0424	$\mathbf{+1.0810 \pm 0.0355}$	2.966	–	2.971
RMSNorm	clamped $\boldsymbol{\gamma}$	5.8158 ± 0.0217	$+0.0187 \pm 0.0272$	1.000	–	1.000

Additional paired LayerNorm contrast	ΔCE ± SE	t
Bias-free clamp minus trainable-bias clamp	-0.6022 ± 0.0916	-6.57

The bias-frozen LayerNorm result distinguishes gain-dependent damage from the bias-enabled gap under gain clamping. Trainable gain remains strongly harmful without bias, so β is not necessary for this failure. Clamping gain while freezing bias improves on the biased clamp and leaves no detected gap from the fixed reference. At this operating point, β provides a route around the gain constraint. RMSNorm supplies an architecture-family transfer test, with a detected trainable-minus-fixed contrast and an unresolved clamp-minus-fixed contrast.

Direct input-RMS control. To suppress scale regardless of whether $\boldsymbol{\gamma}$ or $\boldsymbol{\beta}$ carries it, we penalize the pre-DAC tensor h_P by $\lambda(\mathrm{RMS}(h_P) - 1)^2$. A seed-5 pilot, excluded from the reported comparisons, selected $\lambda = 1.0$ from $\{0.01, 0.1, 1.0\}$ by taking the largest weight that still trained stably. Across five matched-evaluation paired seeds, the regularizer reduces input RMS and analog CE

relative to the matched NA-∅ control, improving both ideal CE and the readout penalty (Table 51). It remains worse than both matched N-∅ and N-A controls. These results support controlling input scale to reduce damage, although the regularizer changes both ideal loss and readout penalty and does not isolate either contribution.

Table 51: Direct input-RMS regularization on V32k. Contrasts are paired over five seeds; brackets give two-sided 95% t_4 intervals.

Configuration	Mean input RMS
NA-∅	3.686
Input-RMS regularization	1.494
Paired contrast	**Mean [95% CI]**
Regularized minus NA-∅, analog CE	-1.3158 $[-2.1122, -0.5195]$
Regularized minus NA-∅, ideal CE	-0.6925 $[-1.1974, -0.1875]$
Regularized minus NA-∅, readout penalty	-0.6234 $[-0.9310, -0.3157]$
Regularized minus N-∅, analog CE	$+0.3492$ $[0.2207, 0.4777]$
Regularized minus N-A, analog CE	$+0.2542$ $[0.0809, 0.4274]$

H.4 VISION TASKS

The language-model evidence leaves open whether the failure and its repairs cross task family. We therefore evaluate a ViT/CIFAR-100 model with forward 6-bit reads, training-time 9-bit backward reads, six readout configurations, five chip-paired seeds, and final official-test analog CE as the primary endpoint. The higher backward precision is a training choice made on the basis of pilot runs. With 6-bit backward reads, every gradient we inspected inside the vision blocks was exactly zero, whereas 9-bit reads produce a trainable trunk. The model is a compact ViT of our own configuration: four blocks, width 128, four heads, 4×4 patches, a digital patch embedding, and analog attention/MLP blocks and classifier head. It trains for 15000 steps with MpAdamW, batch 128, a 400-step warmup, cosine $3 \times 10^{-4} \to 3 \times 10^{-5}$, and random horizontal flips.

The primary experiment trains on 45,000 images and reserves a deterministic, class-balanced 5,000-image validation split. A post-outcome exploratory comparison trains on all 50,000 images and monitors training metrics only. Both use six configurations and five paired seeds, with a single evaluation on all 10,000 official-test images at the fixed final step. Development and backward-precision pilots consulted the official test set; their runs are excluded from the reported comparisons.

Table 52: Compact-ViT full official-test CE. Cells are mean ± SD; parenthesized values are paired ΔCE ± SE versus N-∅. Bold deltas cross the frozen $|t| > 2.776$ threshold. The 45k/5k experiment is primary; the 50k experiment is post-outcome diagnostic.

configuration	45k/5k primary	50k diagnostic
N-∅	2.7503 ± 0.0456	2.6981 ± 0.0535
NA-∅	2.7748 ± 0.0524 $(+0.0245 \pm 0.0196)$	2.7864 ± 0.0228 $(+\mathbf{0.0883 \pm 0.0274})$
NA-∅ + γ-clamp	2.7278 ± 0.0510 (-0.0226 ± 0.0204)	2.7398 ± 0.0380 $(+0.0416 \pm 0.0246)$
N-A	2.7219 ± 0.0421 (-0.0284 ± 0.0146)	2.7336 ± 0.0525 $(+0.0355 \pm 0.0222)$
NA-N	2.8615 ± 0.0268 $(+\mathbf{0.1112 \pm 0.0145})$	2.8575 ± 0.0132 $(+\mathbf{0.1594 \pm 0.0202})$
NA-NA	2.7509 ± 0.0605 $(+0.0006 \pm 0.0200)$	2.7547 ± 0.0346 $(+0.0566 \pm 0.0208)$

Table 52 reports both experiments. The primary NA-∅−N-∅ contrast is positive but not statistically resolved. The exposed configuration's lower ideal CE offsets most of its extra readout penalty. The exposed affine still increases absolute logit perturbations and the readout loss penalty, alongside gain and input-scale growth (Table 53). At the matched scales of Figure 18, the damage coefficient stays close to the N-∅ softmax-spread prediction. Deferring the affine until after the ADC improves analog CE relative to the exposed readout; the clamp does not cross the threshold. Against N-∅, neither N-A nor the clamp differs detectably.

The 50k diagnostic detects the net NA-∅−N-∅ contrast, while its extra readout penalty remains similar. What changes descriptively is the ideal account, which no longer offsets the readout penalty. The paired 45k-versus-50k interaction remains unresolved (Table 54).

Post-ADC normalization changes the ideal task fit. NA-N is significantly worse than N-∅ in both experiments because its ideal CE increases. NA-NA has lower mean CE than NA-N, but its difference from N-∅ is not statistically resolved in either experiment. Figure 18 uses the primary experiment. Its scale interventions show task-dependent deviations from the softmax-spread prediction. The enwik8 curve bends below it at high error, CIFAR-100 stays close, and TinyImageNet-200 lies slightly above it.

Table 53: Compact-ViT CIFAR-100 readout diagnostics. Component differences are NA-∅ minus N-∅; coefficient fractions use the N-∅ softmax-spread prediction.

Mean diagnostic	45k/5k primary	50k diagnostic
Ideal CE difference	-0.0503	$+0.0031$
Readout CE penalty difference	$+0.0748$	$+0.0851$
Observed / predicted damage coefficient	94–104%	95–110%
Primary endpoint	**N-∅**	**NA-∅**
Readout CE penalty	0.0415	0.1163
Gain RMS	1.000	1.609
Input RMS	1.000	1.639
Absolute logit perturbation	0.310	0.547

Table 54: Additional CIFAR-100 analog-CE contrasts, mean ± SE over five paired seeds.

Contrast	Mean ± SE	t
Primary N-A minus NA-∅	-0.0529 ± 0.0174	-3.05
Diagnostic minus primary NA-∅−N-∅ effect	$+0.0637 \pm 0.0420$	1.52

A separate TinyImageNet-200 experiment uses 200 classes, 64×64 images, 8×8 patches, the same four-block width-128 ViT, five chip-paired seeds, and one full-validation-set endpoint per run. Its NA-∅−N-∅ contrast is positive and detected, whereas the clamp and N-A do not differ detectably from N-∅ (Table 55). Deferring the affine also lowers mean analog CE relative to the exposed readout on TinyImageNet-200.

Table 55: **Pre-DAC and post-ADC operations on Compact-ViT TinyImageNet-200.** Full-validation-set analog CE (five seeds; lower is better). The first numeric column is mean ± SD; the last column gives the paired ΔCE ± SE versus N-∅. Bold deltas satisfy $|t| > 2.776$.

configuration	analog CE	paired ΔCE vs N-∅
N-∅	3.4570 ± 0.0287	–
NA-∅	3.5276 ± 0.0392	$\mathbf{+0.0706 \pm 0.0074}$
NA-∅ + γ-clamp	3.4645 ± 0.0183	$+0.0074 \pm 0.0053$
N-A	3.4592 ± 0.0137	$+0.0021 \pm 0.0111$
NA-N	3.6735 ± 0.0194	$\mathbf{+0.2165 \pm 0.0084}$
NA-NA	3.5894 ± 0.0305	$\mathbf{+0.1323 \pm 0.0075}$

Cross-task scale and precision checks. The frozen scale interventions use held-out, train-derived validation, with common endpoints and eight scales across tasks. The vision precision sweeps use one full-test-set endpoint on CIFAR-100 and one full-validation-set endpoint on TinyImageNet-200 at out5, out6, and out7 (Table 56). These mechanism and placement contrasts are secondary to the 45k/5k study's exposed-minus-fixed analog-CE contrast and use unadjusted intervals.

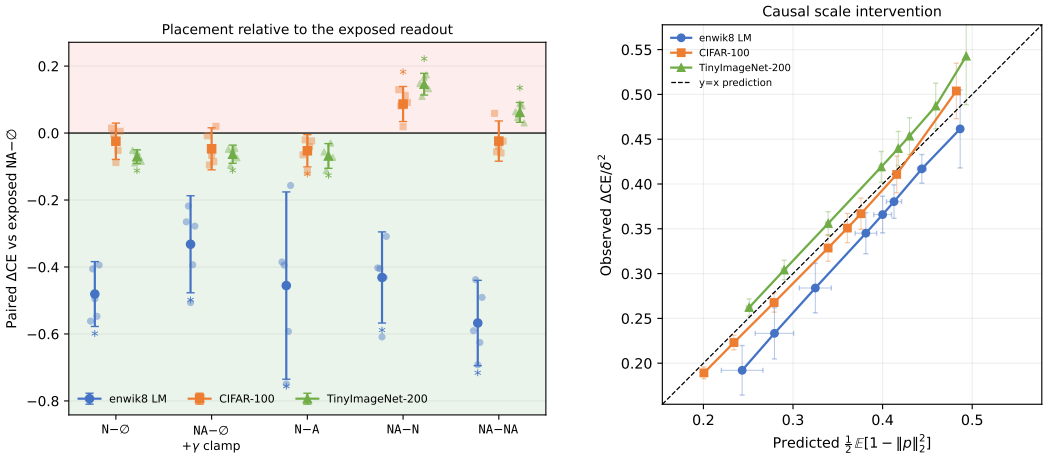


Figure 18: Readout placement and conversion damage across three benchmarks. The left panel shows paired analog-CE changes from NA-∅ to alternative configurations, with seed dots, paired means, and 95% t_4 intervals. Stars mark unadjusted intervals excluding zero. The right panel shows matched scale interventions on frozen BP N-∅ readouts and each task's softmax-spread prediction.

Table 56: Paired NA-∅−N-∅ analog CE by output resolution (mean ± SE, five seeds). Bold crosses the $|t| > 2.776$ threshold.

task	out5	out6	out7
CIFAR-100	$\mathbf{+0.4634 \pm 0.1152}$	$+0.0245 \pm 0.0196$	$+0.0095 \pm 0.0119$
TinyImageNet-200	$\mathbf{+0.2803 \pm 0.0224}$	$\mathbf{+0.0706 \pm 0.0074}$	-0.0211 ± 0.0078

H.5 Quantization-aware training

We compare readout configurations over five seeds each on enwik8 and a 45k/5k clean-split CIFAR-100 ViT, with one full-test-set evaluation per vision run. Digital pretraining is followed by AIHWKit target-bit QAT with exact autograd weight updates and an analog-forward inference RPU configuration. This setup isolates quantization-aware adaptation and is not pulsed analog-update training. All mapped linear layers use the target out6 ADC. We compare NA-∅, N-∅, N-A, and NA-N, each pretrained from scratch with its readout configuration in place. All configurations use 5,000 digital pretraining steps followed by 2,000 QAT steps on enwik8, and 15,000 pretraining steps followed by 5,000 QAT steps on CIFAR-100. Both stages use AdamW with batch size 128 and cosine learning-rate decay after warmup; Table 57 gives the schedules.

Table 57: Digital pretraining and QAT schedules. Learning rates decay from the first value to the second.

Dataset	Stage	Steps	Warmup steps	Learning rate
enwik8	Pretraining	5,000	400	$3 \times 10^{-4} \rightarrow 3 \times 10^{-5}$
enwik8	QAT	2,000	100	$10^{-4} \rightarrow 10^{-5}$
CIFAR-100	Pretraining	15,000	400	$3 \times 10^{-4} \rightarrow 3 \times 10^{-5}$
CIFAR-100	QAT	5,000	100	$10^{-4} \rightarrow 10^{-5}$

Table 58 reports the endpoints, and Table 59 gives the paired comparisons. Target-bit QAT substantially repairs the absolute NA-∅ PTQ failure on both tasks. It does not erase the effect of readout configuration. Both N-∅ and N-A have lower analog CE than NA-∅ after vanilla QAT, with lower mean ideal loss and readout penalty. In contrast, NA-N has higher analog CE on both tasks, so post-ADC normalization alone does not improve these QAT pipelines.

Table 58: Readout-QAT results (mean ± SD, five seeds). Loss is matched validation CE on enwik8 and one-shot official-test CE on CIFAR-100. IDEAL digitally replays only the programmed readout weights; penalty is quantized minus ideal loss.

dataset	pipeline / adaptation	PTQ CE	adapted CE	ideal CE	penalty
enwik8	NA-∅, vanilla QAT	7.6045 ± 0.1495	2.5173 ± 0.0248	2.4745 ± 0.0254	0.0428 ± 0.0010
enwik8	N-∅, vanilla QAT	6.4441 ± 0.0635	2.4314 ± 0.0141	2.4075 ± 0.0147	0.0238 ± 0.0006
enwik8	N-A, vanilla QAT	7.0825 ± 0.0878	2.4786 ± 0.0146	2.4461 ± 0.0138	0.0325 ± 0.0010
enwik8	NA-N, vanilla QAT	–	2.7596 ± 0.0598	2.7152 ± 0.0592	0.0444 ± 0.0015
CIFAR-100	NA-∅, vanilla QAT	6.9152 ± 0.3761	2.2241 ± 0.0247	2.1379 ± 0.0248	0.0862 ± 0.0037
CIFAR-100	N-∅, vanilla QAT	6.0508 ± 0.3281	2.1408 ± 0.0237	2.1147 ± 0.0220	0.0261 ± 0.0031
CIFAR-100	N-A, vanilla QAT	6.3417 ± 0.3928	2.1710 ± 0.0237	2.1247 ± 0.0240	0.0463 ± 0.0021
CIFAR-100	NA-N, vanilla QAT	–	2.3544 ± 0.0190	2.2261 ± 0.0181	0.1283 ± 0.0049

Table 59: Paired CE contrasts in the QAT studies. Values are mean ± SE over five seeds; intervals are two-sided 95% t_4 intervals.

Dataset	Contrast	Mean ± SE	t
enwik8	NA-∅ QAT minus PTQ	-5.0872 ± 0.0704	-72.28
enwik8	N-∅ minus NA-∅, vanilla QAT	-0.0859 ± 0.0117	-7.34
enwik8	N-A minus NA-∅, vanilla QAT	-0.0387 ± 0.0135	-2.87
enwik8	NA-N minus NA-∅, vanilla QAT	$+0.2423 \pm 0.0316$	7.68
CIFAR-100	NA-∅ QAT minus PTQ	-4.6911 ± 0.1625	-28.87
CIFAR-100	N-∅ minus NA-∅, vanilla QAT	-0.0833 ± 0.0036	-22.96
CIFAR-100	N-A minus NA-∅, vanilla QAT	-0.0531 ± 0.0013	-41.72
CIFAR-100	NA-N minus NA-∅, vanilla QAT	$+0.1303 \pm 0.0137$	9.52
NA-∅ QAT minus PTQ, enwik8, 95% CI		$[-5.2826, -4.8918]$	
NA-∅ QAT minus PTQ, CIFAR-100, 95% CI		$[-5.1422, -4.2400]$	

H.6 EXTENSION TO LOCAL CROSS-ENTROPY TRAINING

Local cross-entropy (LCE) (Nøkland & Eidnes, 2019) applies the same readout path separately to intermediate blocks, serving as a layer-local alternative to backpropagation (BP) that avoids propagating errors backward across blocks. To test whether the placement results extend to this learning rule, we repeat the 2×4 comparison of $P \in \{\mathtt{N}, \mathtt{NA}\}$ and $Q \in \{\varnothing, \mathtt{N}, \mathtt{A}, \mathtt{NA}\}$ under LCE. The model, converter settings, and statistical conventions are given in Appendix A; the eight configurations use one matched chip per seed across five seeds.

The endpoint replay ordering of Section 3.2 also holds under LCE: the exposed NA-∅ readout has higher loss than N-∅ both before and after ideal-head replay (Table 60). Thus, removing final-head conversion error does not remove the entire gap between the separately trained configurations.

Table 60: **Operations before the DAC and after the ADC on enwik8 under LCE.** Rows choose P and columns choose Q. Entries give analog validation PPL (mean ± sample SD over five seeds) / mean readout PPL cost $\mathrm{PPL}_{\mathrm{analog}} - \mathrm{PPL}_{\mathrm{ideal}}$ at the same checkpoint.

$P \backslash Q$	∅	N	A	NA
N (LCE)	23.64 ± 2.00 / $+3.33$	26.58 ± 1.13 / $+2.02$	23.04 ± 1.54 / $+2.74$	22.94 ± 1.94 / $+2.06$
NA (LCE)	32.99 ± 1.10 / $+10.93$	25.07 ± 1.51 / $+1.78$	28.83 ± 3.00 / $+5.78$	23.45 ± 1.96 / $+1.98$

Table 60 reproduces the placement pattern under BP: post-ADC normalization removes most of the exposed readout's PPL cost, whereas an affine-only output gives a smaller repair. The paired analog PPL difference between NA-NA and N-∅ is -0.18 $[-3.52, 3.16]$, so the apparent advantage is not separated from zero. Under LCE, these results separate exposed from protected readouts without ranking the protected ones. Once P is parameter-free, a normalizing Q also changes task fit: under LCE, N-N lowers the readout penalty relative to N-∅ while raising analog PPL. Reducing conversion cost therefore need not improve the final task performance.

H.7 TRAINING-TIME OUTPUT-RESOLUTION SWEEP

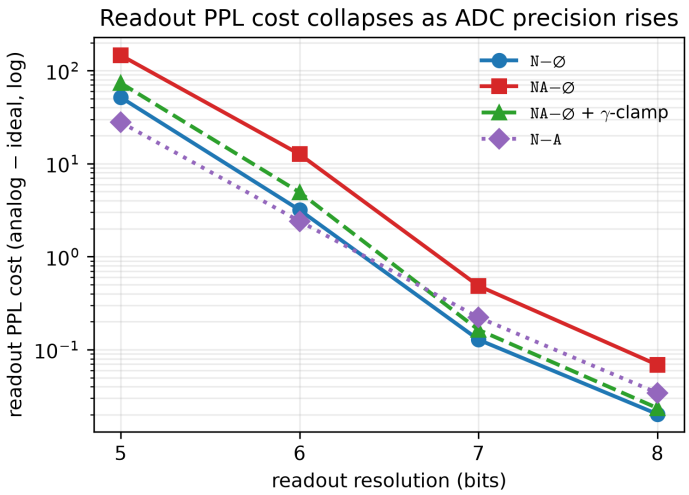


Figure 19: Readout PPL cost across training-time converter precisions for four readout configurations. The vertical axis is logarithmic.

Table 61 gives the full data behind Figure 19. The out6 rows are the primary $V = 205$ runs evaluated on the shared batches, so they match Tables 1 and 11; out5, out7, and out8 are separately trained under the same protocol. At each point of this sweep, the forward output ADC and the backward-MVM output both read at `out_res` bits, while the forward and backward inputs stay at 6, so `out_res` here is a training-time precision axis rather than a read-only one; the frozen-checkpoint replays of Appendix D.8 vary the forward ADC alone.

Readout cost falls sharply as precision increases. At out5, all configurations have large, variable costs, and their ranking is unresolved. The exposed `NA-∅` readout has the largest mean cost at out6, but higher precision substantially reduces its cost.

Table 61: **Full readout-resolution sweep.** Validation PPL, ideal-head PPL, and readout PPL cost are mean ± SD over five seeds within each precision and readout configuration.

`out_res`	mode	val ppl	ideal ppl	readout PPL cost
5	`N-∅`	79.15 ± 25.86	27.60 ± 2.98	$+51.55 \pm 23.66$
5	`NA-∅`	174.51 ± 83.99	28.23 ± 3.55	$+146.29 \pm 84.43$
5	`NA-∅` + γ-clamp	100.91 ± 32.13	27.39 ± 2.83	$+73.52 \pm 31.37$
5	`N-A`	57.83 ± 13.70	29.98 ± 6.60	$+27.84 \pm 7.40$
6	`N-∅`	21.64 ± 2.33	18.47 ± 1.34	$+3.17 \pm 1.04$
6	`NA-∅`	34.96 ± 4.51	22.35 ± 1.24	$+12.61 \pm 3.51$
6	`NA-∅` + γ-clamp	24.99 ± 1.10	20.08 ± 0.75	$+4.91 \pm 0.42$
6	`N-A`	22.29 ± 2.66	19.88 ± 2.36	$+2.40 \pm 0.40$
7	`N-∅`	12.29 ± 0.74	12.16 ± 0.73	$+0.128 \pm 0.012$
7	`NA-∅`	13.07 ± 0.71	12.58 ± 0.70	$+0.485 \pm 0.039$
7	`NA-∅` + γ-clamp	12.84 ± 0.49	12.67 ± 0.49	$+0.164 \pm 0.005$
7	`N-A`	12.54 ± 0.28	12.32 ± 0.29	$+0.222 \pm 0.016$
8	`N-∅`	7.073 ± 0.147	7.053 ± 0.147	$+0.0203 \pm 0.0009$
8	`NA-∅`	7.086 ± 0.137	7.018 ± 0.137	$+0.0687 \pm 0.0021$
8	`NA-∅` + γ-clamp	7.038 ± 0.084	7.015 ± 0.084	$+0.0235 \pm 0.0015$
8	`N-A`	7.076 ± 0.080	7.042 ± 0.081	$+0.0343 \pm 0.0017$

H.8 OUTPUT-NOISE DETAILS

All results above use zero read noise. We add $\texttt{out_noise} \in \{0, 0.02, 0.05, 0.1\}$ to N-∅, NA-∅, and N-A at out6. Each noise level uses five paired seeds. NA-∅ has the highest mean analog PPL at every noise level; Table 62 reports its paired gap from N-∅.

A checkpoint-controlled intervention separates scale from variant identity. On each of five frozen N-∅ checkpoints at $\texttt{out_noise} = 0.05$, scaling the readout input over eight fixed c values in $[0.5, 3]$ increases readout PPL cost monotonically (Table 63). Thus, δ tracks the scale-induced readout damage within a checkpoint under noise, rather than only separating training variants. In this sweep, both the NA-∅ readout cost and its analog-PPL gap against N-∅ decrease over the tested noise levels.

Table 62: Output-noise comparisons. Paired analog-PPL differences are means with 95% t_4 intervals over five seeds.

Output noise	NA-∅ − N-∅
0	$+13.37$ $[9.86, 16.89]$
0.02	$+11.36$ $[7.58, 15.13]$
0.05	$+8.80$ $[6.48, 11.12]$
0.1	$+4.77$ $[3.32, 6.22]$

Table 63: Scale intervention on five frozen N-∅ checkpoints at output noise 0.05. Values are mean ± SE; the correlation is computed within each checkpoint over eight scales.

Quantity	Mean ± SE
Readout PPL cost at $c = 0.5$	$+1.54 \pm 0.17$
Readout PPL cost at $c = 3$	$+115.49 \pm 21.84$
$r(\delta, \Delta\mathrm{PPL}_{\mathrm{ro}})$	0.8948 ± 0.0015

I FROZEN-HEAD PRECISION REPLAY AND POST-ADC REPAIRS

We test whether increasing final-head precision or fitting a digital post-ADC repair lowers loss while keeping the trained network fixed.

I.1 FROZEN-HEAD PRECISION REPLAY

We freeze five seed/chip checkpoints of the V32k/d128-L4 model trained for 15,000 steps with pulsed MpAdamW and an exposed out6 analog head. Within each seed, the out6, out7, out8, and digital heads replay the same captured h_P, programmed weights, and four fixed evaluation batches. The out7/out8 heads differ only in final-head output resolution; the digital head uses exact multiplication with the same weights. All trained parameters remain fixed.

Table 64: Frozen-head precision replay on V32k/d128-L4. CE and δ are five-seed means; paired CE changes from out6 have 95% t_4 intervals. The readout penalty is each row's CE minus the digital row.

Head	CE	δ	ΔCE vs out6 [95% CI]
out6	8.51482	2.24576	–
out7	8.01890	1.02704	-0.4959 $[-0.8089, -0.1830]$
out8	7.73303	0.54718	-0.7818 $[-1.0328, -0.5307]$
digital	7.60271	0	-0.9121 $[-1.2262, -0.5980]$

Increasing final-head output precision lowers CE and absolute logit error (Table 64). Out8 removes most of the immediate readout penalty, while exact multiplication removes it by definition. These comparisons measure the effect of final-head conversion at fixed trained parameters.

I.2 POST-ADC REPAIR LADDER

At fixed out6 precision, we fit digital output repairs to five step-5000 checkpoints from the primary exposed-readout d128-L4 runs ($V = 205$). Each fitted repair minimizes cross-entropy on 4,096 calibration tokens and their next-token targets, then is evaluated on 2,048 tokens from a disjoint held-out stream. All network parameters remain frozen. Trainable repairs use Adam for 400 steps at learning rate 0.03, sampling 256 calibration tokens with replacement per step. Gains start at one and biases at zero; scalar gains are optimized in log space.

For each token, centered vocabulary normalization is $\nu(\tilde{z}) := (\tilde{z} - \overline{\tilde{z}})/\sqrt{\mathrm{Var}(\tilde{z}) + 10^{-5}}$, where the mean and variance are over vocabulary coordinates. Table 65 defines the repairs and reports their held-out losses. Here $g > 0$ is a scalar gain and $a, b \in \mathbb{R}^V$ are per-class affine parameters. Every repair leaves analog precision unchanged.

Table 65: Post-ADC repairs on frozen d128-L4 checkpoints. CE is a five-seed mean; paired differences from normalized full affine have 95% t_4 intervals.

Repair	Function	Trainable parameters	Held-out CE
no operation	$\tilde{z}$	0	3.64267
temperature	$g\tilde{z}$	1	3.43036
normalization	$\nu(\tilde{z})$	0	3.92317
norm. + temperature	$g\nu(\tilde{z})$	1	3.45282
norm. + bias	$\nu(\tilde{z}) + b$	V	3.33662
norm. + temp. + bias	$g\nu(\tilde{z}) + b$	$V + 1$	3.30315
norm. + full affine	$a \odot \nu(\tilde{z}) + b$	$2V$	3.28315

Contrast	Paired ΔCE [95% CI]
norm.+temp. − norm.+full affine	$+0.1697$ $[0.0871, 0.2522]$
norm.+temp.+bias − norm.+full affine	$+0.0200$ $[0.0013, 0.0387]$

Relative to the unrepaired readout, normalization alone raises mean CE, while fitted repairs lower it (Table 65). Per-class bias narrows the gap between normalized temperature and full affine, but a difference remains detected. These calibration results do not establish training-time capacity requirements. Appendix G.2 separately compares post-ADC affine capacities during training.