FROM FRONT-LOADING TO GRADUAL REFINEMENT: SIMPLEX GATING IN TRANSFORMER DECODERS

Anonymous authors
Paper under double-blind review

ABSTRACT

We observe pronounced *front-loading* of residual-norm growth in transformer decoders: the first block sharply amplifies the residual stream, while subsequent blocks produce much smaller relative increases. Our analysis identifies a structural asymmetry underlying this pattern: pre-normalization standardizes sublayer inputs but leaves residual-update magnitudes unconstrained, giving a write of a given size greater relative influence near the input, where the stream is smallest. Motivated by this asymmetry, we introduce *simplex gating* to regulate residual writes in both attention and MLP sublayers. An input-dependent scalar gate scales the attention output, while normalized group gates reweight MLP activations before the down-projection. The MLP gates are nonnegative and sum to one for each token, enabling adaptive allocation of a fixed total gate weight across activation groups. Under matched pre-training conditions, simplex gating substantially alleviates front-loading, reducing the first-block residual-norm growth ratio from 9.10× to 1.84×. It also improves pre-training convergence at two model scales, achieving consistently lower training loss. Ablations show that the simplex constraint mitigates compensatory activation growth that can weaken independent sigmoid gates. These findings demonstrate the practical benefits of controlling residual writes and suggest a path from front-loaded residual growth toward more gradual refinement across depth.

1 INTRODUCTION

Recurring patterns in transformer behavior can reveal opportunities for architectural improvement. For example, emergent activation outliers (Dettmers et al., 2022) were linked to attention heads attempting to produce negligible residual updates, motivating clipped softmax and gated attention (Bondarenko et al., 2023; Qiu et al., 2025). Similarly, observations of uneven layer effectiveness motivated changes to how normalization scales with depth (Sun et al., 2025), and the emergence of attention sinks motivated streaming-inference remedies (Xiao et al., 2024). These examples illustrate an empirical route to architectural design: identify a reproducible pattern, investigate its underlying mechanism, and evaluate an intervention through its effect on training.

In this work, we investigate a pronounced imbalance in the growth of the residual stream. Transformer decoders build representations by repeatedly adding attention and MLP outputs to a shared residual state (Vaswani et al., 2017; Elhage et al., 2021), a design inherited from deep residual networks (He et al., 2016). We refer to these increments as *writes*. Prior work has studied how predictions crystallize across depth by reading out intermediate states (nostalgebraist, 2020; Belrose et al., 2023), or by dissecting what individual layers compute (Geva et al., 2021; 2023). Here we instead track the size of each write: measuring the residual norm before and after each block reveals that relative growth is strongly concentrated at the beginning of the network: the first block sharply amplifies the stream, while subsequent blocks produce much smaller relative increases. We call this phenomenon *front-loading*. We observe it in both a LLaMA-style decoder trained from scratch and a pretrained Qwen decoder, motivating investigation beyond a single training configuration. The relevant imbalance concerns the size of each write relative to the stream it enters. This observation raises a question about architectural design: can moderating dominant early writes improve how representations develop across depth and, ultimately, improve training?

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.

Our analysis identifies a structural asymmetry that helps explain this pattern. In a pre-normalized decoder (Xiong et al., 2020), each sublayer reads a normalized copy of the residual stream and adds its output to the unnormalized stream. RMS normalization (Zhang & Sennrich, 2019) makes the sublayer's computation approximately insensitive to input scale, but leaves the magnitude of its write unconstrained. Consequently, a write of a given size has greater relative influence when the incoming stream is small. This asymmetry is particularly pronounced in the first block, which receives the token embedding before residual updates have accumulated. As the stream's norm grows, later writes require progressively larger magnitudes to maintain the same relative scale. These observations motivate controlling residual updates at their source.

We introduce *simplex gating*, a lightweight architectural modification that regulates writes in both attention and MLP sublayers. Input-dependent gating has a long history in deep networks and sequence models (Srivastava et al., 2015; Chung et al., 2014). An input-dependent scalar gate scales the attention output. In the MLP, we partition the intermediate activations into groups and reweight them before the down-projection. Input-dependent sigmoid scores are normalized to produce non-negative group weights that sum to one for each token, following sigmoid-based expert gating in mixture-of-experts models (Nguyen et al., 2024; DeepSeek-AI, 2024). The MLP gates therefore share a fixed total gate weight, while adapting its allocation across activation groups. All groups remain active, preserving dense computation. This design combines adaptive scaling of attention outputs with normalized allocation across MLP activation groups. It differs from prior interventions that damp the stream with input-independent scalings (Bachlechner et al., 2021; Touvron et al., 2021; Wang et al., 2024; Shleifer et al., 2021), or that redesign how the stream is aggregated (Xie et al., 2023; Zhu et al., 2025; Xie et al., 2025; Kimi Team, 2026): simplex gating instead meters the write itself.

We evaluate simplex gating through controlled pre-training experiments at two model scales. Under matched training conditions, the method substantially alleviates front-loading: in the smaller model, the first-block residual-norm growth ratio falls from $9.10\times$ to $1.84\times$. It also improves pre-training convergence, with the gated models achieving consistently lower training loss than their ungated counterparts at both scales. These training gains establish the practical value of the modification alongside its effect on residual-stream dynamics. Ablations examine the roles of normalization and gate placement. In particular, independent sigmoid gates can be weakened by compensatory growth in the activations they scale; normalization across MLP groups mitigates this behavior, supporting the simplex constraint.

To investigate how these gains relate to changes in the residual stream, we examine downstream dependence on the input embedding, the effective rank (Roy & Vetterli, 2007) of residual representations, and robustness to weight quantization (Nagel et al., 2021; Frantar et al., 2022; Lin et al., 2024). These analyses offer complementary perspectives on information retention, representational diversity, and sensitivity to perturbations. We use them to investigate how moderating early residual growth may preserve information for subsequent refinement, while grounding the architectural proposal in its measured pre-training gains.

2 PRELIMINARIES

Throughout this paper, we consider an autoregressive language model with pre-norm structure over a token sequence $w_1, \ldots, w_T$, with positions $t \in \{1, \ldots, T\}$ predicting the next token. Let y_t denote the target next token at position t. We write d for the hidden width and L for the number of decoder blocks. Each input token w_t is mapped to its embedding, and the targets are the one-step-shifted tokens:

$$x_0^{(t)} = W_E[w_t], \qquad y_t = w_{t+1},$$

where W_E denotes the embedding matrix. For each token t, $x_l^{(t)}$ denotes the residual state after l blocks, for $l = 0, \ldots, L$, with $x_0^{(t)}$ the token embedding.

Each decoder block follows the standard residual update

$$x_{l+1}^{(t)} = x_l^{(t)} + \mathrm{block}_l\big(x_l^{(t)}\big), \qquad l = 0, \ldots, L-1,$$

where block_l consists of a pre-normalized attention sublayer and a pre-normalized MLP sublayer, each of which writes an increment into the residual stream.

Pre-norm block. Following the LLaMA architecture (Touvron et al., 2023), each block applies an attention sublayer and an MLP sublayer in the pre-norm arrangement (Xiong et al., 2020): each sublayer reads a normalized copy of the residual stream and writes an increment back into it. Dropping the token index (t) and the layer index on sublayer parameters for brevity,

$$\tilde{x}_l = x_l + a_l, \qquad a_l = \mathrm{Attn}_l\big(\mathrm{RMSNorm}(x_l)\big), \tag{1}$$

$$x_{l+1} = \tilde{x}_l + m_l, \quad m_l = W_2\,\mathrm{SwiGLU}\big(\mathrm{RMSNorm}(\tilde{x}_l)\big), \tag{2}$$

where $\mathrm{RMSNorm}(x) = \gamma \odot x/\sqrt{\|x\|_2^2/d + \epsilon}$ with a learnable $\gamma \in \mathbb{R}^d$ (Zhang & Sennrich, 2019), $\mathrm{SwiGLU}(h) = \mathrm{SiLU}(W_1 h) \odot (W_3 h)$ with $W_1, W_3 \in \mathbb{R}^{D\times d}$ and $W_2 \in \mathbb{R}^{d\times D}$ (Shazeer, 2020), and Attn_l is causal multi-head self-attention followed by its output projection W_O; at position t it attends to the normalized states of positions $1, \ldots, t$. Hence $\mathrm{block}_l(x_l) = a_l + m_l$. Up to ϵ, $\mathrm{RMSNorm}(cx) = \mathrm{RMSNorm}(x)$ for any $c > 0$, so each sublayer's write depends only on the direction of its input; Section 3.2 builds on this property.

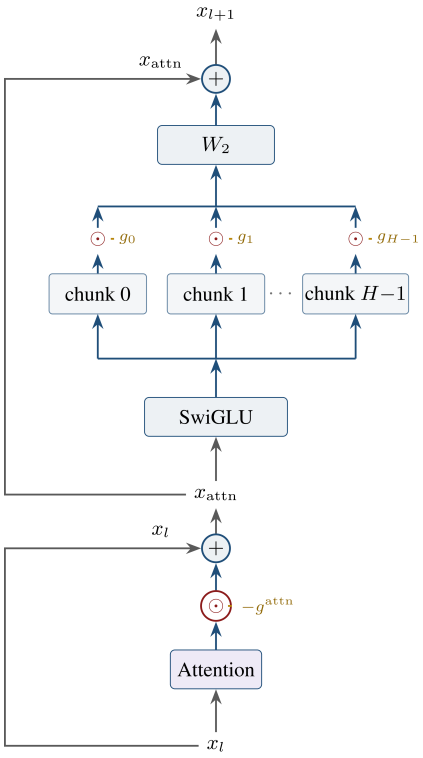


Figure 1: Gated writes in one decoder block: a scalar gate scales the attention output, and group gates scale MLP activations before W_2.

Gated writes. We introduce gating mechanisms in both the attention sublayer and the MLP sublayer, which makes our architecture slightly different from the classic LLaMA-style block above. Each gate is computed from the normalized input h that its sublayer reads, $h = \mathrm{RMSNorm}(x_l)$ for attention and $h = \mathrm{RMSNorm}(\tilde{x}_l)$ for the MLP, and scales the sublayer's write before it enters the residual stream; the gated writes below replace a_l and m_l in equations 1 and 2. For the attention sublayer, the gate is an input-dependent scalar,

$$\begin{aligned} a_l &= g^{\mathrm{attn}}\,\mathrm{Attn}_l(h), \\ g^{\mathrm{attn}} &= 2\sigma\big(W_g^{\mathrm{attn}} h\big), \end{aligned} \tag{3}$$

where $W_g^{\mathrm{attn}} \in \mathbb{R}^{1\times d}$. This follows the sigmoid output gating ofBondarenko et al. (2023) and Qiu et al. (2025). The factor 2 centers the gate at the identity ($2\sigma(0) = 1$), so the gate can amplify as well as attenuate the write. For the MLP sublayer, we partition the D hidden units of SwiGLU into H contiguous groups $\mathcal{G}_1, \ldots, \mathcal{G}_H$ of size D/H and weight each group by a normalized sigmoid score:

$$\begin{aligned} s_i &= \sigma\big((W_g h)_i\big), \quad g_i = \frac{s_i}{\sum_{j=1}^{H} s_j}, \\ m_l &= W_2\big(G^{\mathrm{MLP}} \odot \mathrm{SwiGLU}(h)\big) \\ &= \sum_{i=1}^{H} g_i\, W_{2,\mathcal{G}_i}\,\mathrm{SwiGLU}_{\mathcal{G}_i}(h), \end{aligned} \tag{4}$$

where $W_g \in \mathbb{R}^{H\times d}$, $G_k^{\mathrm{MLP}} = g_i$ for every $k \in \mathcal{G}_i$, and $W_{2,\mathcal{G}_i}$ and $\mathrm{SwiGLU}_{\mathcal{G}_i}(h)$ denote the columns of W_2 and the entries of $\mathrm{SwiGLU}(h)$ indexed by $\mathcal{G}_i$. The rightmost form shows that the gated MLP is a dense mixture of H sub-MLPs whose mixing weights lie on the simplex, $\sum_i g_i = 1$. This normalization is adapted from the expert gating of DeepSeek-V3 (DeepSeek-AI, 2024), which computes expert affinities with a sigmoid and normalizes them over the selected experts. Unlike DeepSeek-V3, we keep every group active (no top-k routing), so the gate reallocates the MLP's write across groups rather than selecting among them. Setting $g^{\mathrm{attn}} \equiv 1$ and $G^{\mathrm{MLP}} \equiv \mathbf{1}$ recovers the baseline block analyzed in Section 3.

Unrolling the recursion expresses the residual stream as the sum of the token embedding and the successive block writes:

$$x_l^{(t)} = x_0^{(t)} + \sum_{j=0}^{l-1} o_j^{(t)}, \qquad o_j^{(t)} = \mathrm{block}_j\big(x_j^{(t)}\big). \tag{5}$$

We call $o_j^{(t)}$ the *write* of layer j, and this paper studies how the relative sizes of these writes shape the shared residual stream. Given any residual state x, the model's own final decoding path computes

$$\mathrm{logits}(x) = \mathrm{lm_head}\big(\mathrm{ln_f}(x)\big), \qquad p(y \mid x) = \mathrm{softmax}\big(\mathrm{logits}(x)\big)[y], \tag{6}$$

where $\mathrm{ln_f}$ is the final normalization and $\mathrm{lm_head}$ is the output projection.

2.1 RELATIVE WORK

Prior work has responded to control the property of the residual stream along two lines. Input independent scale control damps the stream with factors: extra normalization inside the sublayers (Shleifer et al., 2021), residual-branch scaling (Wang et al., 2024), and depth-dependent LayerNorm scaling (Sun et al., 2025). Input-dependent stream redesign instead changes how the stream is aggregated: Attention Residuals replace the fixed unit-weight accumulation with softmax attention over preceding layer outputs (Kimi Team, 2026), and Hyper-Connections mix several parallel channels through learnable, input-dependent gates (Zhu et al., 2025). Notably, unconstrained learned mixing proved unstable at scale until the manifold-constrained variant projected the mixing matrices onto the doubly stochastic manifold (Xie et al., 2025)—hard constraints are what make residual control usable. Yet every remedy above governs the stream's scale or its mixing: none imposes a hard budget on the write itself, the increment a sublayer emits. This paper studies write-side control.

3 THE UN-METERED WRITE: RESIDUAL NORM GROWTH IN TRANSFORMER DECODERS

In this section we characterize the residual stream as a shared channel in which every block writes at a price that is independent of the size of the stream it enters. We show that this pricing follows directly from pre-normalization, that it concentrates the stream's growth in the first block, and that the first block's write is therefore not the largest write in the network but the most leveraged one. These structural facts motivate the write-side control studied in the rest of the paper.

We use a LLaMA-style decoder trained from scratch on OpenWebText for 40,000 iterations (152M parameters, 80B tokens), and a larger, instruction-tuned Qwen decoder as a generalization check. The LLaMA baseline has $L = 12$ blocks of width $d = 768$; Qwen2.5-3B-Instruct has $L = 36$ blocks of width $d = 2048$ (Table 3). For evaluation, LLaMA samples a token sequence from a fixed OpenWebText validation window, and Qwen samples from the `DKYoon/SlimPajama-6B` test split.

3.1 THE SCALE PROFILE OF THE RESIDUAL STREAM

A residual decoder implements $x_l^{(t)} = x_0^{(t)} + \sum_{j=0}^{l-1} o_j^{(t)}$ with $o_j^{(t)} = \mathrm{block}_j\big(x_j^{(t)}\big)$. No term in this sum is constrained: each block is free to write a vector of any magnitude and any direction, and nothing in the training objective records whether a write sharpens, duplicates, or overwrites what the stream already carries (Elhage et al., 2021). The composition of the stream is therefore an equilibrium reached by training dynamics. To make the scale of these writes visible, we measure the mean per-token ℓ_2 norm of each residual state,

$$\bar{x}_l = \frac{1}{T} \sum_{t=1}^{T} \big\| x_l^{(t)} \big\|_2, \qquad r_l = \frac{\bar{x}_{l+1}}{\bar{x}_l}, \tag{7}$$

and call r_l the *growth ratio* of block l.

Applying this readout to the LLaMA baseline, Figure 2 shows that the mean residual norm grows by $9.10\times$ at block 0, far more than at any later block, and reaches a final value of 218.4 from an

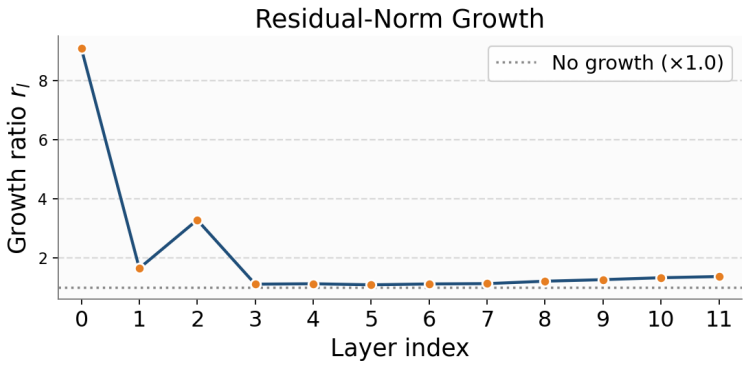


Figure 2: Per-layer residual-norm growth ratio $r_l = \bar{x}_{l+1}/\bar{x}_l$ for the LLaMA baseline.

Table 1: Residual-stream scale profile of two standard decoders. In both models the first block dominates the growth of the stream.

Model	emb $\|x\|$	L0 $\|x\|$	L0 growth	Final norm
LLaMA Baseline	0.94	8.59	9.10×	218.4
Qwen2.5-3B-Instruct	1.03	20.01	19.34×	474.0

embedding norm of 0.94 (Table 1). Qwen2.5-3B-Instruct exhibits the same profile, with a first-block growth ratio of $19.34\times$. We refer this very general phenomenon as **front-loading** in residual growth.

Two facts stand out. First, the first block delivers a large share of the network's total norm growth: in log space it contributes $\ln(8.59/0.94)/\ln(218.4/0.94) = 40.6\%$ for LLaMA and 48.4% for Qwen. Second, after this initial jump the growth ratios collapse: the remaining blocks grow the stream by an average factor of 1.34 (LLaMA) and 1.09 (Qwen), so the first block's relative increment exceeds the average later block's by $24\times$ and $190\times$, respectively. The single most consequential act of writing in a transformer decoder is also the earliest one.

3.2 An Un-metered Channel: Why the First Write Wins

We first complement the preceding ratio readout with two structural quantities computed from the same measurements: the *write cost* $\bar{o}_l$, the mean norm of the increment block l adds to the stream; the *leverage* $\rho_l = \bar{o}_l/\bar{x}_l$, the size of a write relative to the stream it enters.

$$\bar{o}_l = \frac{1}{T}\sum_{t=1}^{T}\left\|o_l^{(t)}\right\|_2, \qquad \rho_l = \frac{\bar{o}_l}{\bar{x}_l}. \tag{8}$$

Why is the largest relative write the first one? The answer is a property of pre-normalization, and it makes the concentration structural rather than a training accident. In a pre-norm block the write from each submodule, for example the attention, is a normalized stream $\mathrm{RMSNorm}(x_l^{(t)})$. Such true input is scale-invariant: for any scalar $c > 0$, $\mathrm{RMSNorm}(cx) = \mathrm{RMSNorm}(x)$. Hence the write depends only on the *direction* of the stream, never on its size: scaling the stream does not change a single sublayer output.

The write cost $\bar{o}_l$ is therefore decoupled from the stream it enters, and the leverage ρ_l of a write of fixed cost decays exactly as $1/\bar{x}_l$ as the stream accumulates. We call such a write *un-metered*: its cost does not scale with its impact, so the same physical write buys leverage that decays with depth. The first block writes when the stream is smallest, so—unless later blocks grow their write costs at least as fast as the stream accumulates—the largest relative growth must occur at the input. Sun et al. (2025) formalize the same geometry: under their assumptions the per-layer variance follows

$$\sigma^2_{x_{l+1}} \;=\; \sigma^2_{x_l}\,\Theta\Big(1+\frac{1}{\sigma_{x_l}}\Big), \tag{9}$$

whose relative increment $\Theta(1/\sigma_{x_l})$ is, up to the $\sqrt{d}$ translation between per-dimension scale and ℓ_2 norm, exactly the leverage decay of an un-metered write.

Growth concentrated at the initial layer is not be a training artifact to be tuned away; it is a property of the architecture, and any remedy must be architectural. If we view write as the unit of control, the natural question is whether the residual stream can be controlled at the source and what such control buys for training dynamics and the model's behavior. Section 4 introduces the proposed intervention, and Section 5 examines its effects on embedding dependence, residual effective rank, and quantization robustness.

4 Our Proposal: Simplex Gating

The most direct write-side intervention is input independent scalar, which can be fixed as in DeepNet (Wang et al., 2024) or learnable as in ReZero and LayerScale (Bachlechner et al., 2021; Touvron et al., 2021). However such rescaling has served mainly as a stability device (Zeng et al., 2022), not a capacity lever. Input-dependent gating instead conditions the write on the token, providing higher capacity and adjustment flexibility(Qiu et al., 2025).

Same as static scalar, the most naive gating procedure is putting single gating on both attention and MLP, which automatically adjusts the scale of output across tokens and neural depth. However, such pattern of write control has its inherent deficiency at MLP. Specifically, although shallow layers' gating collapse quickly, the activations in MLP, such as SwiGLU(h) would inversely inflate, impairing the controlling effect imposed by gating. Appendix A.1 discusses compensatory effect arising from gating.

In this section we propose *Simplex Gating*, which effectively controls the inflation issues on MLP. Our design is related to the routing of DeepSeek-V3, which uses a sigmoid gate whose selected affinity scores are normalized over the chosen experts (DeepSeek-AI, 2024), and to sigmoid gating in mixture-of-experts (Nguyen et al., 2024).

4.1 Architectural Specification

For an input residual x, let $h = \mathrm{RMSNorm}(x) \in \mathbb{R}^d$. Denote the number of gating groups (or chunks) by H, a hyperparameter of the gating structure. We apply simplex group gating to MLP, which is computed from the normalized residual h:

$$s_i = \sigma((W_g h)_i), \qquad W_g \in \mathbb{R}^{H\times d}, \tag{10}$$

$$g_i = \frac{s_i}{\sum_j s_j}, \qquad \sum_i g_i = 1. \tag{11}$$

We construct $G^{\mathrm{MLP}} \in \mathbb{R}^D$ as the element-wise gating tensor, where for components $k = \frac{D}{H}i, \ldots, \frac{D}{H}i + \frac{D}{H} - 1$, $G^{\mathrm{MLP}}_k = g_i$. The gate mass is fixed at $\sum_i g_i = 1$, which imposes a hard per-token budget on each sublayer's write. We choose to conduct the write control before the down-projection. The final MLP residual-update, illustrated by Figure 1, becomes:

$$x \leftarrow x + W_2\left(G^{\mathrm{MLP}} \odot \mathrm{SwiGLU}(h)\right). \tag{12}$$

Within the attention module, let $g^{\mathrm{attn}} = 2\sigma(W_g^{\mathrm{attn}} h)$, with $W_g^{\mathrm{attn}} \in \mathbb{R}^{1\times d}$. Denote the final output from attention as $o_{attn} \in \mathbb{R}^d$, the attention residual-update is:

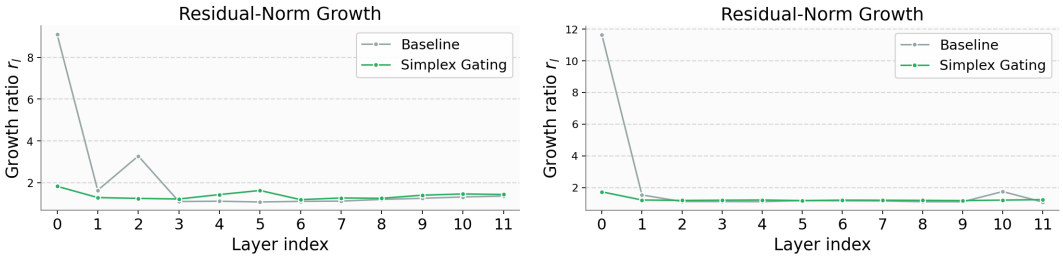
$$x \leftarrow x + g^{\mathrm{attn}}\, o_{attn}. \tag{13}$$

4.2 Experimental Results

We apply the gating approach mentioned above onto two LLaMA-style decoders (152M, 589M), using the data, and training protocol (80B tokens) of the LLaMA baseline described in Section 3. The gating mechanism is the only architectural change, as shown in equation 13 and equation 12.

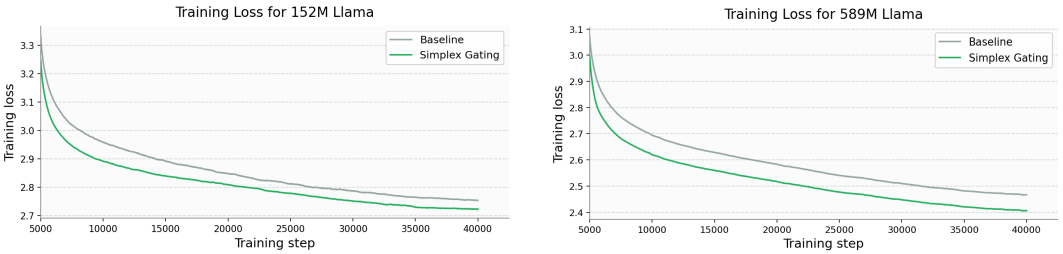
Via the introduced residual-norm growth rate r_l, we observe that in the gated model, the Layer-0 residual grows by $1.84\times$ instead of $9.10\times$ (Figure 3), and the final norm drops from 218.4 to 47.2.

Therefore the front-loading symptom in residual growth get greatly mitigated. The same intervention is visible in the training loss itself. Figure 4 shows the two scales side by side: the gated run stays below the ungated baseline throughout the whole training, and the two curves never cross. To investigate the changes accompanying these training gains, Section 5 examines downstream dependence on token embeddings, the effective rank of residual activations, and robustness to weight quantization.



(a) 152M LLaMA decoder (H=12). (b) 589M LLaMA decoder (H=20).

Figure 3: Per-layer residual-norm growth ratio for the ungated baseline and Simplex Gating at two decoder scales, on a fixed OpenWebText validation sample.



(a) 152M LLaMA decoder (H=12). (b) 589M LLaMA decoder (H=20).

Figure 4: Training loss over $40{,}000$ steps (80B tokens) for the ungated baseline and Simplex Gating, at two decoder scales. Both runs in each pair share the data, optimizer and learning-rate schedule; gating is the only architectural difference.

Why Normalized Gating in MLP: The output from MLP is the main source of overgrowth within layer-0. Without the normalization term $\sum_j s_j$, the raw value s_i would reduce sharply at the early training stage, causing explosive growth of activation SwiGLU(h). This phenomenon can be viewed as compensation effect for gating collapse and finally weaken gates' control on front-loading symptom. By introducing normalization term, the mean value of each gate is maintained around $\frac{1}{12}$, which completely removes compensation effect. The relevant details and experimental results are kept in Appendix A.1.

The Placement of Gating: Unlike the grouped gates before each linear down-projection in MLP, in attention modules we set single scalar gate at output side. Such setting is to differ our design from gated attention structure (Qiu et al. (2025)), showing that these two methods attain seperate gains. On the other hand, if we move the group gating in MLP at the output side, the training performance deteriorate slightly, though the effect is not pronounced.

5 Why Simplex Gating Works

In this section, we examine three properties that distinguish the gated model from the baseline. First, controlling front-loading preserves the information carried by the token embedding, and downstream layers come to rely on it; second, controlled writes keep the residual stream high-rank across depth;

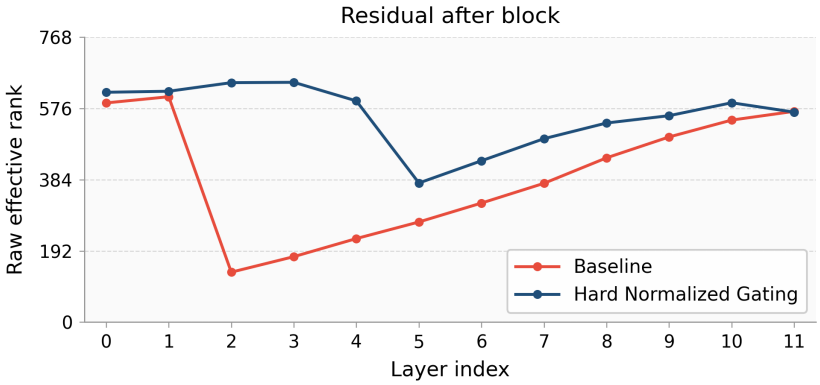


Figure 5: Effective rank of the residual stream after each block for the baseline and hard normalized gating. The gated model stays higher at every block except the last.

third, the gated model is more robust to weight-only quantization. Together, these properties show that the effect of write control extends beyond the residual-norm profile reported in Section 4.

Preserving token identity The front-loading within the baseline, particularly in layer 0, has a second consequence: its output floods the initial residual of transformer, which is the embedding of the input token. We would illustrate this point by a perturbation experiment. Through this perturbation metric, we could show that hard normalized gating is also beneficial for preserving token identity.

Let $x_0^{(t)}$ denote the initial residual at position t, which is token embedding. The perturbation protocol removes it entirely from the layer-1 residual:

$$x_1^{(t)} = x_0^{(t)} + \mathrm{block}_0(x_0^{(t)}), \qquad x_{\mathrm{pert}}^{(t)} = x_1^{(t)} - x_0^{(t)}. \tag{14}$$

Let $\mathrm{blocks}_{1:}(x_{\mathrm{pert}}^{(t)})$ denote the downstream representation of token t obtained by propagating its perturbed state through remaining layers. Then the mean perturbed loss is computed as:

$$\mathcal{L}_{\mathrm{pert}} = \frac{1}{T}\sum_{t=1}^{T} \ell_t\big(\mathrm{blocks}_{1:}(x_{\mathrm{pert}}^{(t)})\big). \tag{15}$$

and we measure the removal effect on prediction by the penalty:

$$\Delta_{\mathrm{rem}} = \mathcal{L}_{\mathrm{pert}} - \mathcal{L}_{\mathrm{standard}}, \tag{16}$$

whose positive values indicate worse downstream loss. Basically, Δ_{rem} measures whether downstream layers causally rely on token identity.

For the LLaMA baseline, removing the embedding changes the loss by only $\Delta_{\mathrm{rem}} = +0.0083$, so downstream layers barely use the original token identity. Qwen is even more extreme, which slightly improves the loss ($\Delta_{\mathrm{rem}} = -0.0009$) through removal. As a contrast, by controlling the write through gating, **Simplex Gating** variant comes to utilize more preserved token identity, reflected in a substantially larger penalty $+5.22$. The gate restores the token embedding, an atomic referent to which downstream predictions must attach, as a load-bearing part of the residual stream.

Controlled Writes Keep the Residual Stream High-Rank Controlling the write is also accompanied by an increase in the effective rank of the residual stream. We measure the effective rank of the residual activations as $\mathrm{erank}(X) = \exp(-\sum_i p_i \log p_i)$, with $p_i = \sigma_i / \sum_j \sigma_j$ (Roy & Vetterli, 2007), where σ_i are the singular values of the activation matrix collected over 32 sampled sequence from validation set (Figure 5). In the baseline, the large updates from the earlier blocks concentrate the residual spectrum in a few directions: the effective rank falls from 608 at Layer 1 to 135 at Layer 2 and recovers only gradually. With hard normalized gating, the spectrum remains far more balanced (646 at Layer 2; layer average 557 vs. 396), and the two models converge only at the last

Table 2: Weight-only INT4 quantization of the ungated baseline and Simplex Group Gating. Mean cross-entropy on fixed OWT samples; ΔCE is the mean loss increase over the unquantized model. Effective bits include scale storage ($4 + 16/g$ for group size g; 4.0 for per-channel).

	Effective bits	Baseline CE	Baseline ΔCE	Gated CE	Gated ΔCE
Unquantized (BF16)	32	2.7744	—	2.7364	—
INT4 per-channel	4.0	3.1048	$+0.3304$ (0.0038)	2.9867	$+0.2503$ (0.0028)
INT4 group-128	4.13	2.9563	$+0.1819$ (0.0022)	2.8892	$+0.1528$ (0.0016)
NF4 group-128	4.13	2.8773	$+0.1028$ (0.0013)	2.8228	$+0.0864$ (0.0011)

layer. This is consistent with the gated model preventing a few large early updates from dominating the residual stream, which leaves room for later layers to write in more diverse directions. We consider this phenomenon a direct measure of representational capacity, which may help explain the improved performance.

Robustness to Weight Quantization The gated model is also more robust to weight quantization than the ungated baseline. We quantize every linear projection of both models with weight-only INT4 (setup in Appendix B.5); the two networks receive nearly the same injected weight error. Under per-channel INT4 the gated model loses 0.2503 CE against 0.3304 for the baseline according to Table 2. The robustness advantage shrinks monotonically as the quantizer improves.

6 Discussion

This paper characterizes and controls the scale of residual writes in transformer decoders. In standard pre-norm decoders, the residual stream grows most sharply in the first block, and we show this concentration is structural: pre-normalization leaves writes un-metered, so the earliest write enjoys the largest leverage. Our remedy, simplex gating, couples a scalar sigmoid gate on the attention output with normalized gates on the MLP activations before the down-projection. Under matched training, first-block growth falls and the final residual norm drops sharply, and training performance improves at both 152M and 589M scales.

Our method differs from gated attention (Qiu et al., 2025) in three respects. In placement, gated attention applies data-dependent gates inside attention to induce nonlinearity and sparsity, whereas we meter the attention output with a single scalar and, more importantly, gate the MLP—the dominant source of first-layer overgrowth, where the simplex normalization imposes a hard per-token budget that cannot be refilled. Actually, the gains might be complementary rather than competing: attention-side nonlinearity and sparsity versus depth-balanced refinement, token-identity preservation, and quantization robustness.

Future work includes validating at multi-billion-parameter scale and longer training, unifying the simplex gate with mixture-of-experts routing, exploiting the quantization advantage through quantization-aware training, and extending evaluation from perplexity to downstream tasks.

AI USE STATEMENT

In this work, we used generative AI tools for the following tasks with required disclosure: helping to develop the conceptual framework of Section 3 and to refine the hypotheses stated there. We have not used generative AI tools for the remaining tasks with required disclosure (generating synthetic data, formulating or proving mathematical claims, designing experiments, cleaning or reformatting datasets). Additionally, we used generative AI tools for the following tasks with recommended disclosure: identifying relevant literature and verifying bibliographic entries, suggesting a structure for the motivation section, drafting and editing parts of the manuscript, and formatting references. We have reviewed all AI-assisted work: every suggested citation was checked against the original publication by the authors, and all AI-drafted text was verified against our training and evaluation logs. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies the scaling behaviour of residual streams in transformer decoders and introduces an architectural modification to them. All experiments use publicly available datasets (OpenWebText and the `DKYoon/SlimPajama-6B` test split) under their original licences; no human subjects, personal data, or sensitive attributes are involved, and no new dataset is released. We foresee no ethical issues specific to this work beyond the general risks inherited by language models trained on web text.

REPRODUCIBILITY STATEMENT

We specify every architectural detail needed to reproduce our results. Section 3 describes the training data, the optimization budget ($40{,}000$ iterations, 80B tokens), and the evaluation protocol; Section 4 gives the gating equations and the corresponding design choices; and Appendix A reports the ablations over gating variants and gate placement. All models are either trained from scratch under the protocol above or evaluated from public checkpoints.

REFERENCES

Thomas Bachlechner, Bodhisattwa Prasad Majumder, Huanru Henry Mao, Garrison W Cottrell, and Julian McAuley. Rezero is all you need: Fast convergence at large depth. In *Uncertainty in Artificial Intelligence (UAI)*, 2021.

Nora Belrose, Zach Furman, Logan Smith, Danny Halawi, Igor Ostrovsky, Lev McKinney, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023.

Yelysei Bondarenko, Markus Nagel, and Tijmen Blankevoort. Quantizable transformers: Removing outliers by helping attention heads do nothing. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS Workshop on Deep Learning*, 2014.

DeepSeek-AI. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm.int8(): 8-bit matrix multiplication for transformers at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. *Advances in Neural Information Processing Systems*, 2023.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021.

Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in auto-regressive language models. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. Openwebtext corpus. `https://skylion007.github.io/OpenWebTextCorpus`, 2019.

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

Kimi Team. Attention residuals, 2026.

Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In *Proceedings of Machine Learning and Systems*, 2024.

Markus Nagel, Marios Fournarakis, Rana Ali Amjad, Yelysei Bondarenko, Mart van Baalen, and Tijmen Blankevoort. A white paper on neural network quantization. *arXiv preprint arXiv:2106.08295*, 2021.

Huy Nguyen, Nhat Ho, and Alessandro Rinaldo. Sigmoid gating is more sample efficient than softmax gating in mixture of experts. In *Advances in Neural Information Processing Systems*, 2024.

nostalgebraist. Interpreting gpt: The logit lens. *LessWrong*, 2020. URL `https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens`.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Olivier Roy and Martin Vetterli. The effective rank: A measure of effective dimensionality. In *15th European Signal Processing Conference (EUSIPCO)*, pp. 606–610, 2007.

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

Sam Shleifer, Jason Weston, and Myle Ott. Normformer: Improved transformer pretraining with extra normalization. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, 2021.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

Wenfang Sun, Xinyuan Song, Pengxiang Li, Lu Yin, Yefeng Zheng, and Shiwei Liu. The curse of depth in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 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 *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. Deepnet: Scaling transformers to 1,000 layers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(10):6761–6774, 2024. doi: 10.1109/TPAMI.2024.3386927.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations (ICLR)*, 2024.

Shufang Xie, Huishuai Zhang, Junliang Guo, Xu Tan, Jiang Bian, Hany Hassan Awadalla, Arul Menezes, Tao Qin, and Rui Yan. Residual: Transformer with dual residual connections. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Zhenda Xie, Yixuan Wei, Huanqi Cao, Chenggang Zhao, Chengqi Deng, Jiashi Li, Damai Dai, Huazuo Gao, Jiang Chang, Kuai Yu, Liang Zhao, Shangyan Zhou, Zhean Xu, Zhengyan Zhang, Wangding Zeng, Shengding Hu, Yuqing Wang, Jingyang Yuan, Lean Wang, and Wenfeng Liang. mhc: Manifold-constrained hyper-connections. *arXiv preprint arXiv:2512.24880*, 2025.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *International Conference on Machine Learning (ICML)*, 2020.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, et al. GLM-130B: An open bilingual pre-trained model. *CoRR*, abs/2210.02414, 2022.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Defa Zhu et al. Hyper-connections. In *International Conference on Learning Representations (ICLR)*, 2025.

Table 3: Architectures of the LLaMA and Qwen decoders used in the experiments.

Property	LLaMA baseline	Qwen2.5-3B-Instruct
Layers	12	36
Hidden size	768	2048
Attention heads	12	16
Key/value heads	12	2 (GQA)
FFN hidden size	3072	11008
Vocabulary	50304	151936
Normalization	RMSNorm	RMSNorm
Activation	SwiGLU	SwiGLU
Tied embeddings	Yes	Yes
Training	From scratch	Instruction-tuned

A Gating Design Choices and Rationale

A.1 Soft gating is bypassed by activation growth

Simplex Gating differs from a soft-gated MLP in one critical respect: the proposed gate forces the per-group coefficients to sum to one, whereas independent soft gates are free to shrink. Note that when $H = 1$, soft gating is essentially naive scalar gating. This subsection tests whether that difference matters. We instantiate the proposed MLP with independent sigmoid gates, keep every other component identical, and examine the learned gate values and the raw activations they multiply. If the soft design already limits the Layer-0 write, the normalized budget would be unnecessary.

Specifically, Soft Group Gating computes

$$g_i = 2\sigma((W_g h)_i), \qquad W_g \in \mathbb{R}^{H\times d}. \tag{17}$$

Attention uses the single output gating as Hard Normalized Gating.

The learned Layer-0 MLP gates are strongly suppressed and nearly uniform, with an overall mean gate of approximately 0.082. Figure 6 shows that this suppression is learned early in training: between steps 1000 and 5000 the mean gate collapses and remains small.

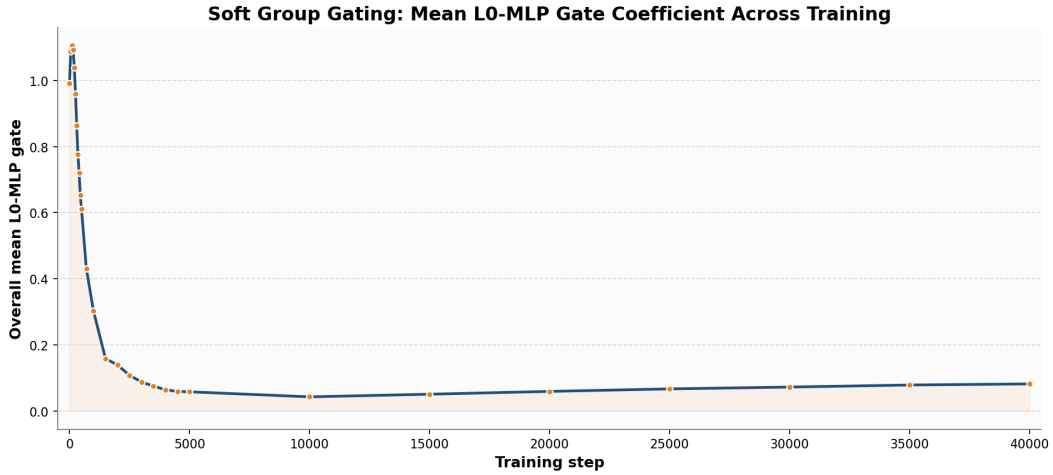


Figure 6: Soft Group Gating: mean L0-MLP gate coefficient across training.

Gate suppression, however, does not reduce the write as much as expected. The mean ℓ_2 norm of the raw SwiGLU activation reaches $\mathbf{14.84}$, about $\mathbf{4.07}\times$ the baseline value of $\mathbf{3.64}$. Figure 7 shows that the activation norm diverges from the baseline over the same training window in which the

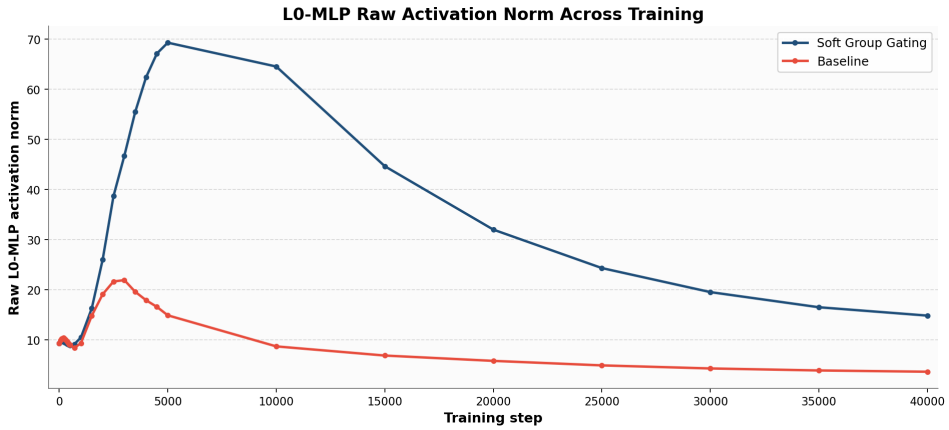


Figure 7: Raw L0-MLP activation norm across training for Soft Group Gating and the baseline.

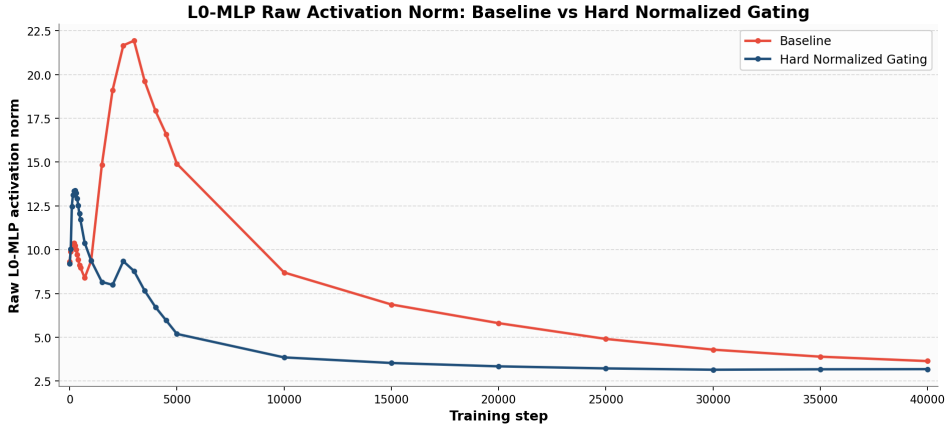


Figure 8: Baseline and Hard Normalized Gating raw L0-MLP activation norm across training.

gates collapse. The two trajectories are therefore coupled: the gate shrinks as intended, but the raw activation inflates and restores the write that the gate was supposed to suppress.

Hard Normalized Gating prevents this compensation. Because the normalized gates sum to one, the model cannot shrink all gate masses at once; the raw Layer-0 activation stays near the baseline at $\mathbf{3.15}$ rather than inflating to $\mathbf{14.84}$, and its training trajectory remains close to the baseline (Figure 8).

B Experimental Details

To ensure reproducibility, we provide the detailed configurations used for the pre-training experiments.

B.1 Model Architecture

Our baseline models follow the standard Llama-2 architecture (Touvron et al., 2023), featuring RMSNorm for pre-normalization, SwiGLU activation functions, and Rotary Positional Embeddings (RoPE). The specific architectural hyperparameters for the models used in our main comparisons are listed in Table 4.

Table 4: Architectures of the LLaMA Decoders .

Property	LLaMA 152M	LLaMA 589M
Layers	12	20
Hidden size	768	1280
Attention heads	12	20
Key/value heads	12	20
FFN hidden size	3072	5120
Vocabulary	50304	50304
Normalization	RMSNorm	RMSNorm
Activation	SwiGLU	SwiGLU
Tied embeddings	Yes	Yes
Training	From scratch	From scratch

B.2 Initialization and Optimization

Models are initialized using a normal distribution with mean 0. For the majority of layers, including `nn.Embedding` and standard linear projections, we utilize a fixed standard deviation of $\sigma = 0.02$. However, to control variance growth along the residual path, the initialization scale is adjusted specifically for the output mapping layers. Accordingly, the weights for the self-attention output projection (W_O) and the FFN downward projection (W_{down}) are initialized with $\sigma = 0.02/\sqrt{2L}$, where L is the number of layers.

We employ the AdamW optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.95$. The training utilizes a cosine learning rate schedule with a linear warmup phase. Detailed optimization hyperparameters are provided in Table 5.

B.3 Infrastructure

All experiments were conducted on a computing node equipped with $8\times$ NVIDIA H200 GPUs using a Distributed Data Parallel (DDP) strategy. The total training time for Llama 589M was approximately 1.2 days.

B.4 Dataset

The models are pre-trained on the OpenWebText dataset (Gokaslan et al., 2019), an open-source reproduction of the WebText dataset. The data is tokenized using the standard GPT-2 tokenizer. Due to the total training volume target, we apply repeated sampling on the dataset during the training process.

B.5 Quantization Stress-Test Setup

Every linear projection of the ungated baseline and Simplex Group Gating is quantized with weight-only INT4 in the simplest form (Nagel et al., 2021): symmetric per-output-channel round-to-nearest, $\hat{w} = \text{clamp}(\text{round}(w/s), -8, 7) \cdot s$ with $s = \max_j |w_j|/7$ along each output channel. Embeddings and normalization weights are kept in full precision, and the quantized weights are dequantized on the fly, so the forward pass runs in BF16 exactly as in the unquantized setting. Two more accurate 4-bit variants serve as robustness controls: group-wise scales shared by 128 input channels, and the non-uniform NF4 codebook of Dettmers et al. (2023). Both models are scored on the same samples drawn from the OpenWebText validation set, and every comparison is paired per block. The injected weight error is the relative Frobenius norm $\|\hat{W} - W\|_F / \|W\|_F$ over all quantized entries (0.1474 for the baseline, 0.1471 for the gated model).

Table 5: Pre-training hyperparameters.

Hyperparameter	Value
Peak Learning Rate	0.001
Min Learning Rate	0.0001
Warmup Iterations	2,000
Max Iterations	40,000
Batch Size	12
Block Size	4096
Grad Accumulation Iters	5
Number of GPUs	8
Weight Decay	0.1
Gradient Clipping	1.0
Precision	bfloat16
Optimizer	AdamW