SWIGNLU: NONLINEAR VALUE BOTTLENECKS FOR PARAMETER-MATCHED GATED FFNS

Anonymous authors
Paper under double-blind review

ABSTRACT

The feed-forward network holds most non-embedding parameters in a modern decoder, yet in SwiGLU the branch that carries its content is affine: a nonlinear gate multiplies a linear value projection. We ask whether the same feed-forward budget is better spent giving that value branch a learned nonlinear feature dictionary. SwiGNLU replaces $W_u x$ with $U_2\sigma(U_1 x)$, computing $W_\downarrow(\sigma(W_g x) \odot U_2\sigma(U_1 x))$; choosing $v_h \le dh/(d+h)$ makes the replacement no larger than SwiGLU and 0.03–1.0% cheaper in matmul FLOPs. Under one shared Muon/AdamW-on-qkv protocol, all twenty-eight completed paired comparisons favor SwiGNLU: eighteen at matched budgets, spanning 160M to 1.1B parameters on FineWeb and on independently trained Longmino models, with mean gains of 7.1–9.6 millibits per byte per cell; and ten that vary the token budget from $0.62\times$ to $9.9\times$ the Chinchilla-optimal ratio. The gain converts to 13–16% fewer tokens to reach the control's final quality, survives training to $9.9\times$ Chinchilla undiminished, transfers to held-out corpora with a larger relative margin in seven of eight cells, and moves a seven-task zero-shot battery by $+0.59$ points (16 of 17 pairs). Measured end to end on identical hardware, training step time matches within 1.5%, so equal quality arrives 12–15% sooner in wall-clock; the serial value path costs 2.7–5.1% per decoded token at batch 1 and shrinks with batch. Both families share the same best measured learning rate at three of four rungs, so SwiGNLU retrofits at the baseline's own tuned LR. A parameter-matched 2×2 shows a consistently positive interaction between the bottleneck and its interior nonlinearity, accounting for 34–37% of the combined improvement. Muon amplifies the gain without creating it: under pure AdamW at 305M, SwiGNLU wins all nine pairs at about half the margin. At 160M it also beats squared ReLU, GEGLU, and ReGLU at matched parameters. We release modules, recipe, paired run-level evidence, and complete curves.

1 INTRODUCTION

SwiGLU (Shazeer, 2020) computes $W_\downarrow(\sigma(W_g x) \odot W_u x)$. Its gate is nonlinear but its value branch is affine. We study the smallest direct replacement that gives that branch a learned nonlinear feature dictionary, in the key–value-memory view of the FFN (Geva et al., 2021), while keeping the FFN budget fixed: $W_u x$ becomes $U_2\sigma(U_1 x)$. The FFN holds most non-embedding parameters in a modern decoder, so a systematic improvement to its function class compounds across every block and every training token.

For output coordinate r,

$$y_r = \sum_{j=1}^{h} \sum_{k=1}^{v_h} W_{\downarrow,rj} U_{2,jk}\, \sigma(w_{g,j}^\top x)\sigma(u_{1,k}^\top x). \tag{1}$$

SwiGNLU combines h gate features with a shared dictionary of v_h nonlinear value features, and U_2 learns how each gate channel reads that dictionary. We do not claim its function class strictly contains SwiGLU at fixed budget; we test whether this reallocation is a better inductive bias, and what it costs to use.

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.

Table 1: Summary of measured effects, SwiGNLU against SwiGLU. Each entry is stated with its population and its method in the section named.

Category	Measure	Result	Sec.
QUALITY	Paired gain, per cell	7.1–9.6 mbpb	5
	Relative gain	0.92–1.24% of bits/byte	5
	Completed pairs	28 in 11 cells, all favoring SwiGNLU	5
COST	Tokens to equal quality	13–16% fewer	6
	Wall-clock to equal quality	12–15% less	6
	Training step time	within 1.5%	6
	Decode, batch 1	+2.7–5.1% per token	6
	Peak training memory	+0.4–6%	6
ROBUSTNESS	Pure AdamW, no Muon (305M)	+3.4 mbpb, 9 of 9 pairs	8
	Against squared ReLU (160M)	+2.6 mbpb, 3 of 3 pairs	8
	Token budget	undiminished to 9.9× Chinchilla	8
	Held-out corpora	larger relative margin, 7 of 8 cells	8
	Downstream macro	+0.59 points, 16 of 17 pairs	8
ADOPTION	Shared best measured LR	3 of 4 measured rungs	3
	Parameter and FLOP budget	0.03–1.0% under SwiGLU	2
MECHANISM	2×2 interaction	34–37% of the gain	7

Under one shared protocol, all twenty-eight completed paired comparisons favor SwiGNLU, with a flat 0.92–1.24% relative gain in bits per byte across a 73× span of training FLOPs; Table 1 collects the measured effects and names the section that establishes each. The supplementary material releases every run's records and curves, the serving measurements, the trainer, and one script that regenerates every table from those records.

The name SwiGNLU refers to the operator, not a recipe-independent claim: every comparison uses a Muon-class optimizer with AdamW on the unchanged qkv projection unless labeled otherwise, and we state that boundary wherever it binds.

2 THE DESIGN AND ITS BUDGET ARGUMENT

SwiGNLU replaces SwiGLU's value map $W_u \in \mathbb{R}^{h \times d}$ with $U_2\,\sigma(U_1 x)$, where $U_1 \in \mathbb{R}^{v_h \times d}$ and $U_2 \in \mathbb{R}^{h \times v_h}$, and changes nothing else:

$$\mathrm{SwiGLU}(x) \;=\; W_{\downarrow}\big(\sigma(W_g x) \odot \qquad W_u x\ \big), \tag{2}$$

$$\mathrm{SwiGNLU}(x) \;=\; W_{\downarrow}\big(\sigma(W_g x) \odot \boxed{U_2\,\sigma(U_1 x)}\big), \qquad \sigma = \mathrm{SiLU}. \tag{3}$$

Both activations are SiLU (Hendrycks & Gimpel, 2016; Elfwing et al., 2018), the same nonlinearity SwiGLU places at its gate, so the contrast holds the activation fixed and varies where in the block it acts: four projections and two SiLUs against three and one. With every projection bias-free, parameter cost per layer is $2dh + v_h(d + h)$ against $3dh$, so $v_h \le dh/(d + h)$ guarantees no increase, and at our chosen v_h the replacement runs 0.03–1.0% under SwiGLU in both parameters and forward matmul FLOPs (Table 2); at the standard $h = \frac{8}{3}d$ the cap is $v_h \approx 0.727d$. The default keeps h fixed and takes the largest legal v_h. Controlled factorial cells may instead adjust h for a closer integer match, within 0.1% of the baseline's parameters (Section 4). Activations sit outside the budget: backward keeps the interior SiLU's v_h-wide input and output, the source of the +0.4–6% peak training memory in Table 4 (whose 305M and 599M geometries also widen h).

The bottleneck rank is a genuine architecture-specific coordinate, so we report two comparisons throughout: a *common-policy* comparison that fixes optimizer routing and learning rate across arms (the causal contrast), and a *per-arm-optimum* comparison in which each family runs at its own measured best learning rate (the adopter's contrast).

Table 2: The reported ladder (d/L between 51 and 64 at every rung; $h \approx 8d/3$; batch 24/16/16/16/32 by rung). Δ columns are SwiGNLU relative to SwiGLU: at these geometries the treatment is strictly under budget in parameters and forward matmul FLOPs at every rung. The 305M and 599M rows show the best-legal geometry of Appendix A, in which SwiGNLU widens h to 3072 at 305M; the common-policy cells of Table 3 at those rungs use $(h, v_h) = (2816, 704)$ and $(4032, 512)$, at $+0.086\%$ and $+0.066\%$ parameters (Section 4). Token budgets are fixed; no run carries a wall-clock cap. The 1.1B budget is $0.95\times$ Chinchilla-optimal and spans 1.54 epochs of the FineWeb shard; both arms see identical repeated batches, so the contrast is unaffected, and the repetition is disclosed here; the single-epoch 160M Longmino ladder separates high token-per-parameter ratios from repetition by design (Section 8).

Rung	d	Layers	h (SwiGLU)	v_h	Δ params	Δ FLOPs	Tokens	Steps
160M	768	12	2048	512	-0.97%	-0.98%	2B	40,690
305M	1024	16	2752	512	-0.34%	-0.33%	4.4B	134,277
599M	1280	24	3392	896	-0.62%	-0.57%	10B	305,175
762M	1408	26	3776	1024	-0.03%	-0.03%	13.3B	405,883
1.1B	1600	31	4288	1152	-0.22%	-0.19%	21.3B	325,012

3 THE RETROFIT RECIPE

An engineer with a tuned SwiGLU stack needs three facts, all measured.

Sizing. Keep d and h; set $v_h = \lfloor dh/(d+h) \rfloor$, or the largest multiple of 64 at or below it, which gives the 160M, 599M, 762M and 1.1B cells of Table 2, 0.03–1.0% under budget. Quality is a broad plateau in v_h near the cap: at 762M a $2\times$ range moves wide validation by at most 1.7 mbpb, and no tested configuration in the released runs reverses the direction of the comparison.

Learning rate. Keep the baseline's tuned LR. At 305M, 762M, and 1.1B the two families' best measured learning rates under the shared protocol coincide (2.8×10^{-4}, 2.0×10^{-4}, 2.21×10^{-4}); at 1.1B that rate was located by a six-arm scout before the full runs, and both families sit interior to the bracket at the same point. At 762M the bracket below was probed (1.5×10^{-4}, worse for both). At 305M a revision fleet closed the bracket on both sides (Appendix G): one and two grid steps below 2.8×10^{-4}, SwiGLU loses 3.2 and 10.3 mbpb, and one step below, SwiGNLU loses 5.2 (four seeds per cell). 599M is the one rung where the two families' best measured rates differ; each at its own best measured rate still leaves a 5.2–7.5 mbpb gain. Under pure AdamW the two families again share their best measured rate (Section 8). The retrofit rule this implies is asymmetric and simple: moving to SwiGNLU never asks for a lower learning rate than the baseline's, and tolerates a slightly higher one.

Optimizer routing. Route the two new matrices (U_1, U_2) with the hidden matrices (Muon); AdamW (Loshchilov & Hutter, 2019) keeps embeddings, head, 1-D parameters, and the unchanged qkv projection ($\beta_2 = 0.9999$, weight decay 0.1 except on 1-D parameters, as used here; neither value is established as required). One warning from our own campaign: name-fragment optimizer routers fail silently when a fused implementation renames parameters. At 305M a dead carve (fragments that match no parameter name, so the matrices meant for AdamW silently stay on Muon) costs SwiGNLU 7.0 mbpb and SwiGLU 4.0, shrinking the gain by 38%. The check is one log line: the optimizer's hidden-matrix parameter count must equal non-embedding parameters minus the carved block and the 1-D parameters; the released trainer prints both.

4 EXPERIMENTAL SETUP AND MEASUREMENT DISCIPLINE

Backbone. Every controlled run uses the same decoder-only transformer (Vaswani et al., 2017): RMSNorm pre-norm, rotary embeddings, causal SDPA, untied head, sequence length 2048, GPT-2 BPE (50,257). The FFN is the only module that changes. Token budgets are 0.62–$0.95\times$ the compute-optimal ratio of Hoffmann et al. (2022) (Table 2); because a fixed-horizon win below that ratio could be an under-training artifact, Section 8 trains past it, to $9.9\times$. Training is bfloat16 autocast. Initialization is GPT-2 style, every matrix $\mathcal{N}(0, 0.02^2)$ with the attention-output and $W_\downarrow$

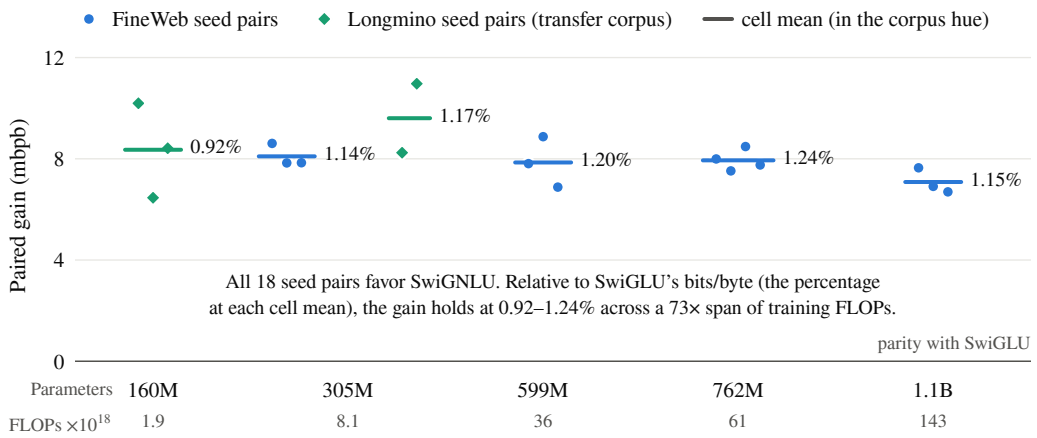

Figure 1: The eighteen matched-budget paired comparisons of SwiGNLU against SwiGLU; the ten budget-varying pairs are Figure 4. Each mark is one seed pair, with gain $= 1000(\mathrm{bpb}_{SwiGLU} - \mathrm{bpb}_{SwiGNLU})$, so above zero favors SwiGNLU: blue circles FineWeb, teal diamonds Longmino, both on 256-window validation. The rule through each cell is its mean and the percentage beside it is that mean as a share of the control's bits/byte. Rungs are five discrete configurations, spaced evenly; the second tick row gives each rung's training FLOPs ($6ND$).

matrices scaled by $1/\sqrt{2L}$; U_1 and U_2 get no variance correction, so SwiGNLU's value branch starts at 0.25–$0.39\times$ SwiGLU's RMS, and the shared LR of Section 3 holds regardless.

Corpora and pairing. FineWeb (Penedo et al., 2024) supplies the main ladder; Longmino is a 204-domain training mixture drawn from a public corpus with a different token/byte ratio; the artifact records the source and the content hash of every training bin, so both corpora can be rebuilt and verified. Within a corpus and scale the same seed initializes each architecture and generates an identical stream of training windows, so the primary observations are paired seed differences. FineWeb holds out its final 20M tokens; Longmino validation windows are stratified across the domain-ordered mixture.

Metrics. We report bits per UTF-8 byte via a corpus-specific measured token/byte factor. All primary comparisons are scored on 256 windows (524,288 tokens), stratified across the domain-ordered bin for Longmino; the 16-window metric recorded along every curve travels in the artifact as a sensitivity check, and the Longmino token ladder is scored on both bases (Appendix B).

Optimization and matching. The common policy is Muon (Jordan et al., 2024; Liu et al., 2025) on hidden matrices, AdamW elsewhere including qkv. Learning rate, schedule, warmup, batch, tokens, attention, and routing are identical within each comparison. Endpoint parameter differences are $+0.086\%$, $+0.066\%$, -0.028%, and -0.216% at 305M–1.1B; at the ladder's SwiGLU size scaling (82 mbpb per e-fold of parameters) the largest is worth 0.18 mbpb, against measured gaps of 7.1–8.1 mbpb. The post-search surfaces of Appendix A instead hold the treatment strictly under the baseline, and the effect reproduces there at every rung.

Population control. A run enters a cell only when every recorded identity field agrees and the run is valid, complete, and full-horizon. The released run table records the cells each run belongs to, and relaunches of one design and seed are averaged before any statistic.

5 Controlled paired comparisons: every pair favors SwiGNLU

The released corpus holds twenty-eight completed seed pairs in eleven cells, and we count them one way throughout: *eighteen matched-budget pairs*, the six scale–corpus cells of Table 3 and Figure 1, and *ten budget-varying pairs*, the 305M over-training cell and the four rungs of the Longmino token ladder (Section 8, Figure 4). All twenty-eight favor SwiGNLU, at every scale, on both corpora, on both metrics. We do not aggregate them into one nominal p-value: scales reuse seed labels and the

Table 3: Common-policy results, in mbpb. *Arm sizes* are SwiGNLUvSwiGLU; *pairs* counts shared seeds. All rows use 256-window (524,288-token) validation; Longmino windows are stratified across the domain-ordered bin. A gain is $1000(\mathrm{bpb}_{SwiGLU} - \mathrm{bpb}_{SwiGNLU})$. ***At best LR*** reruns both arms at their best measured rate (Section 5); †strictly legal geometry $h/v_h = 3072/512$ (-0.34% parameters), treatment from the revision fleet (Appendix G); ‡each family at its own best rate, spanning the legal and matched geometries.

Corpus	Scale	Arm sizes	Pairs	Mean gain ↑	Paired gain range	At best LR
FineWeb	305M	3v3	3	8.10	7.84–8.61	$7.06^{\dagger}$
FineWeb	599M	3v3	3	7.86	6.88–8.87	$5.2–7.5^{\ddagger}$
FineWeb	762M	4v6	4	7.94	7.52–8.48	6.91
FineWeb	1.1B	3v3	3	7.08	6.69–7.64	7.08
Longmino	160M	3v3	3	8.36	6.46–10.19	n/a
Longmino	305M	2v2	2	9.60	8.24–10.97	n/a

cells are not exchangeable replicates. The run-level paired deltas are the primary observations and are reported in full (Table 8).

Sensitivity to the shared learning rate. The shared LR is a choice, so we measured its consequences. At 305M and 762M the declared cells sit one grid step above the best measured setting under this protocol; rerunning both arms there (2.8×10^{-4} and 2.0×10^{-4}) moves both families down together and gives paired gaps of 6.97 (three pairs) and 6.91 mbpb (two pairs), a 13–14% haircut rather than a reordering. At 305M the revision fleet then closed both brackets and reran SwiGNLU at the strictly legal geometry ($3072/512$, -0.34% parameters), which gains 7.06 mbpb at the shared best rate (7.90, 7.28, 6.00; the control is the main campaign's, whose 160M runs the fleet reproduces within 0.4 mbpb). At 1.1B the shared LR is the best measured rate for both families by construction. At 599M the families' best measured rates differ (1.5 and 2.1×10^{-4}), and each at its own leaves 5.2–7.5 mbpb against 7.9 under the common policy; both estimands are reported. The last column of Table 3 carries these values, and Appendix A tabulates every bracket cell (Figure 5).

Scale. The five ladder rungs share aspect ratio (d/L 51–64), one protocol, and one implementation; the relative gain is 0.92–1.24% of absolute bits/byte at every rung, flat across a 73× span of training FLOPs. An earlier wide–shallow 1.1B configuration is excluded as a shape outlier (Appendix A).

6 Tokens, wall-clock, and the serving bill

Token efficiency. At each of 305M, 599M, and 762M we take the control population's mean final bits/byte as the target and scan each SwiGNLU curve for its first crossing (Figure 2). The control needs 1.164×, 1.150×, and 1.184× as many tokens respectively; equivalently, SwiGNLU reaches the target with 14.1%, 13.1%, and 15.5% fewer tokens, scanned on the 16-window metric, the one recorded along every curve (final-checkpoint wide gains are larger at 762M, so the multiple there is conservative). This is a within-cell measurement, not a ladder-wide compute law.

Wall-clock. We benchmarked full end-to-end models of both families co-scheduled on identical hardware (methodology in Appendix E). At each arm's own best configuration, training step time matches within 0.4–1.5% across all five rungs. Composed with the token multiples above and each rung's own step-time delta, SwiGNLU reaches the control's final quality in 12–15% less wall-clock on the same hardware (13.6%, 11.8%, and 15.2% at 305M, 599M, and 762M).

The serving bill. The added projection is serial, and decode is latency-bound, so it cannot be free (Table 4):

The decode premium is largest exactly where decode is latency-bound: at batch 1 the extra serial matmul costs 2.7–5.1% per token across the ladder, and it shrinks as batch grows and the step becomes bandwidth-bound, to −0.7–3.5% at batch 32 and within ±1% of parity at batch 32 with an 8K-token cache. Appendix E covers kernel fusion.

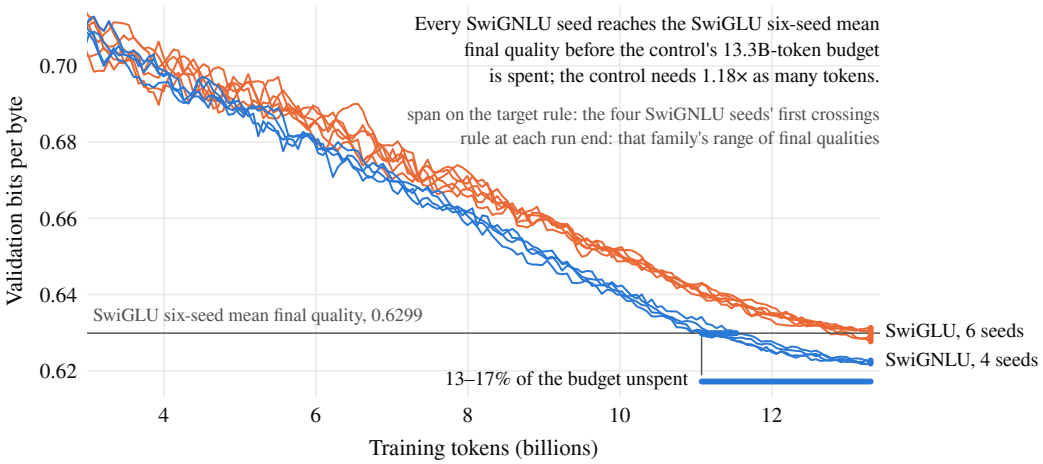

Figure 2: The margin as tokens at 762M: all ten validation curves (16-window metric, the one recorded across training) from 3B tokens onward, SwiGNLU in blue (four seeds) and SwiGLU in orange (six seeds). The horizontal rule is the SwiGLU six-seed mean final quality; the span on it marks the four SwiGNLU first crossings, the short rule at each family's run end spans its final qualities, and the bar below runs from the earliest crossing to the control's full budget. Every SwiGNLU seed crosses with 13–17% of the budget unspent.

Table 4: Measured SwiGNLU cost relative to SwiGLU, end-to-end single-GPU (H200), each arm at its production configuration (fused with `torch.compile`; decode replays a captured CUDA graph). Train is the per-step delta at batch 8; prefill spans batch 8 long-sequence (parity) to batch 1 at 512 or 2048 tokens (worst); decode is incremental single-token latency with a preallocated KV cache, each range spanning 512- to 8192-token contexts; memory is peak allocated during a training step at batch 1–8. Bootstrap 95% intervals on the decode ratios are within ± 0.25 points at batch 1 and ± 0.7 at batch 32.

Rung	Train step	Prefill (load)	Decode/token, batch 1	Decode/token, batch 32	Train peak memory
160M	+0.7%	+0.4–6.5%	+4.9–5.1%	+1.0–3.1%	+2.4–3.2%
305M	+0.6%	+0.5–6.7%	+4.2–4.9%	+0.8–3.5%	+2.2–4.2%
599M	+1.5%	+0.8–7.5%	+2.8–2.9%	−0.2–2.2%	+2.7–6.0%
762M	+0.4%	+0.2–6.2%	+3.4–3.6%	−0.7–2.6%	+2.1–5.1%
1.1B	+0.9%	−0.3–6.1%	+2.7–3.6%	+0.2–2.4%	+0.4–5.2%

7 Mechanism: Bottleneck and Nonlinearity Interact Super-Additively

SwiGNLU differs from SwiGLU by two entangled ingredients (Figure 3): an extra *nonlinearity* and a *bottleneck* that composes two linear maps. Each has a natural control: SwiGLU-dualact applies the activation to the full-width value path (nonlinearity without composition), and SwiGLU-lowrank keeps the bottleneck but deletes the interior activation (an exact rank-v_h factorization). With SwiGLU and SwiGNLU they close a 2×2 matched cell-by-cell in parameters, all four arms native FFNs trained from initialization.

The interaction is positive, that is super-additive, for every seed in all nine matched quadruples (Table 5). Under qkv it supplies 37% of the combined gain at 305M and 34% at 599M; nonlinearity alone supplies 26% and 14%, the linear bottleneck 36% and 52%. Neither "add an activation" nor "factorize the value map" explains the composed arm, which is the substantive answer to reading SwiGNLU as a LoRA-shaped rank trick (Hu et al., 2022; Lan et al., 2020): the purely linear factorization is in the table, and it is not the effect.

The 305M condition is the primary factorial: its four arms ran as one hardware cohort and its bottlenecked rows differ from the full-width rows by only 2.3% in h. The 599M replication carries two

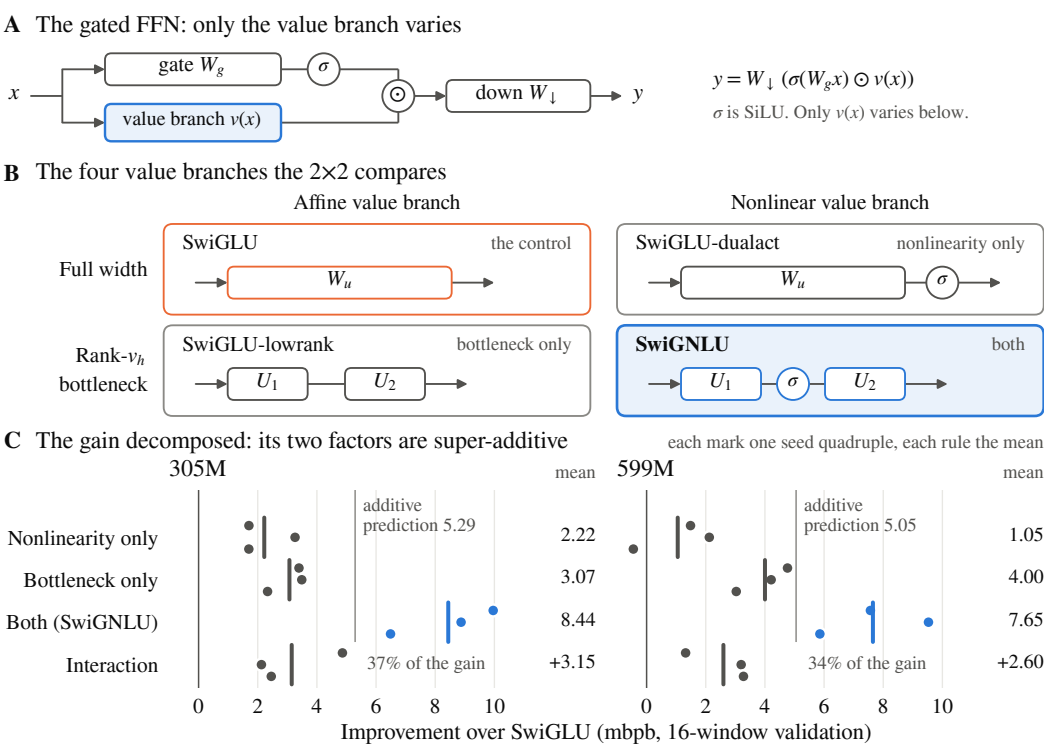

Figure 3: The declared value-path 2×2 at 305M and its 599M replication. Only the value branch varies: A is the gated FFN block, B the four branches the factorial compares, which differ solely in whether the branch passes through a rank-v_h bottleneck and whether it is nonlinear (the SwiGLU control framed in orange, SwiGNLU in blue). C plots each arm's improvement over SwiGLU on the 16-window metric: each mark is one seed quadruple, the rule through each row its mean, the hairline the additive prediction (the sum of the two single-factor means), and the last row the per-seed interaction, positive in every quadruple. Cell geometries are in Table 11.

Table 5: Factorial means as improvement over SwiGLU in mbpb on the 16-window metric, the one recorded for every arm of the factorial (↑, higher is better; the *both* column, SwiGNLU, is best in every row; the composed arms reproduce on the wide metric in Table 3); three matched seed quadruples per row. Interaction is $I_{\text{both}} - I_{\text{nonlin}} - I_{\text{bottleneck}}$ for improvements I; positive is super-additive. The sign convention matches Figure 3.

Scale and condition	Nonlinearity only	Bottleneck only	Both	Interaction
305M, no carve	0.45	0.55	5.59	4.59
305M, qkv	2.22	3.07	8.44	3.15
599M, qkv	1.05	4.00	7.65	2.60

caveats: its arms ran in two hardware groups, and its bottlenecked rows use h=4032 against 3392, so it identifies the interaction of the bottleneck *with its budget reallocation*. A same-h replication at 599M remains open. At 305M, by contrast, the bottleneck-only value map is genuinely lower-rank (704 against 1024) at an h only 2.3% wider. One candidate cause of its gain is the optimizer, since Muon orthogonalizes U_1 and U_2 separately rather than their product; the factorial has not been run under AdamW, where the composed gain survives at about half size (Section 8).

8 Robustness: Optimizer, Baselines, Budgets, Transfer, Tasks

Without Muon. We trained the 305M pair under pure AdamW ($\beta_2 = 0.95$, no Muon on any matrix), SwiGNLU at the strictly legal geometry (3072/512), three seeds at each of three learning

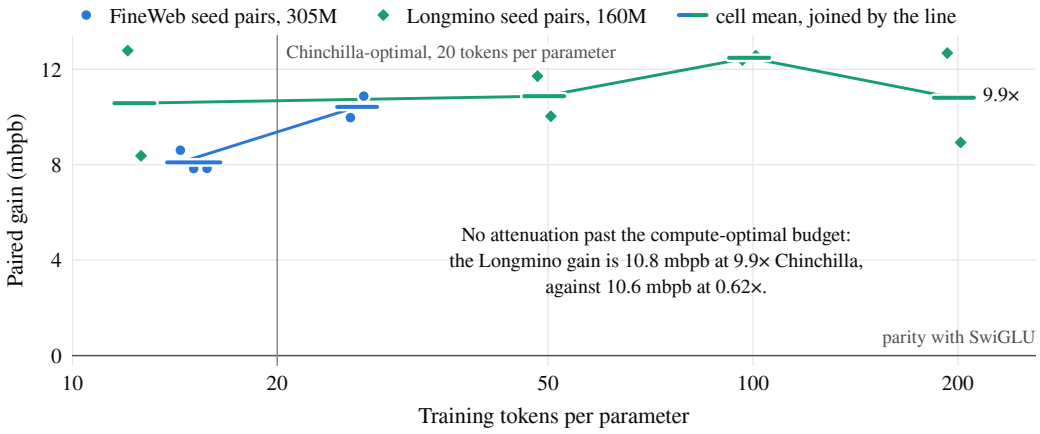

Figure 4: The gain against tokens per parameter (log axis), over every fixed-budget pair in the corpus: blue circles FineWeb at 305M, teal diamonds Longmino at 160M, the convention of Figure 1. Each mark is one seed pair, spread a few points in x so none hides another; the rule through each cell is its mean at the cell's budget, and the line joins the means. The vertical rule is the Chinchilla-optimal 20 tokens per parameter; ratios use total parameters. There is no attenuation through $9.9\times$ that budget. Longmino absolute bits/byte is about $1.5\times$ FineWeb's at these scales, so mbpb are not directly comparable across corpora: relative gains are 0.96–1.24% on the Longmino ladder and 1.14–1.50% on the FineWeb points.

rates. SwiGNLU wins all nine pairs, by 3.38, 3.91, and 5.45 mbpb at 2.8, 4.0, and 5.6×10^{-4} (Appendix G). Both families score best at 2.8×10^{-4}, the lowest rate probed but a flat optimum (the step to 4.0×10^{-4} costs 0.75 and 0.22 mbpb), so the shared-LR retrofit holds without Muon. The AdamW gain there is 0.46% of control bits/byte against 1.00% under Muon: Muon roughly doubles the margin without creating it.

Other FFN variants. At 160M on FineWeb, three seeds each at one learning rate, we compared SwiGNLU with GEGLU, ReGLU (Shazeer, 2020), and Primer's ungated squared ReLU (So et al., 2021) at $h = 3072$, all at SwiGLU's 162.1M parameters (SwiGNLU 0.97% under). Under the qkv routing, SwiGNLU and squared ReLU improve on SwiGLU by 8.3 and 5.7 mbpb while GEGLU and ReGLU trail it by 1.6 and 5.3; SwiGNLU beats squared ReLU in all three pairs, by 2.6 on average. SwiGLU prefers AdamW on $W_{\downarrow}$ to AdamW on qkv by 3.2 mbpb at this rung, and with each family at its better routing SwiGNLU still leads by 5.1 (Appendix G).

Past the compute-optimal budget. Fixed-horizon wins can be under-training artifacts, so we trained the 305M pair to $1.31\times$ the Chinchilla-optimal budget (8B tokens, 26.2 tokens/parameter, carve-matched): the gain is 10.4 mbpb, larger than at the declared horizon. At 160M on Longmino, where the corpus supports high ratios in a single epoch, we trained the pair at fixed 2B, 8B, 16B, and 32B-token budgets ($0.62\times$ to $9.9\times$ the Chinchilla-optimal ratio, 12 to 197 tokens per parameter), two seeds per budget (Figure 4). All eight pairs favor SwiGNLU, with per-budget means of 10.6, 10.9, 12.5, and 10.8 mbpb on the ladder's in-run validation (the held-out tail of the Longmino bin) and 7.2, 8.7, 10.9, and 9.2 rescored on the stratified windows of Table 3 (Appendix B); on either basis the gain shows no attenuation through $9.9\times$ Chinchilla.

Held-out corpora. Scoring the controlled FineWeb checkpoints on two held-out corpora (Dolmino-Math and FineMath4, at 305M, 599M, 762M, and 1.1B: eight cells) and comparing relative gains, so that the probe harness's different absolute level cancels, the SwiGNLU advantage is larger held-out than in-distribution in seven of eight cells, 1.3–2.3% of control bits/byte held-out against 1.1–1.2% in-distribution; in the eighth, 762M on FineMath4, the two are level at 1.21% over the three probed pairs (Table 9). Inside the probe harness alone the ordering holds at 305M, 599M and 1.1B, and at 762M the held-out gains bracket the in-distribution one (Appendix B). The sharpest control is Dolmino-Math (OLMo Team et al., 2024), whose absolute difficulty for these models is within 0.02 bits/byte of FineWeb's on the same harness at every rung: at 305M the advantage there is $1.76\times$ the

in-distribution one on that harness (17.8 against 10.1 mbpb of gain), so the transfer gain is not an artifact of harder text.

Downstream tasks. On a seven-task zero-shot battery (ARC-E/C (Clark et al., 2018), BoolQ (Clark et al., 2019), HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), SciQ (Welbl et al., 2017), WinoGrande (Sakaguchi et al., 2020); byte-length-normalized macro), the paired SwiGNLU advantage is $+0.59$ points (16 of 17 probed pairs, positive in all six cells; the 762M seed-2 control was never probed, and per-cell values are in Appendix B). At these scales individual task magnitudes sit inside seed noise (only HellaSwag is separately resolvable); the claim is the paired macro direction, not per-task movement. MMLU is omitted because 1.0–1.4B models trained on up to 3T tokens still score 24.9–26.6% five-shot, at its 25% chance floor (Zhang et al., 2024).

9 Limitations

(1) Apart from the labeled exceptions, every result uses the Muon/AdamW-on-qkv recipe. Without Muon the gain persists at about half its size, but that check covers one rung, its LR grid is open below the shared best point, and the 2×2 has not been run without Muon. (2) Every result comes from one decoder implementation and training stack. (3) The 599M factorial identifies the bottleneck with its budget reallocation, and 599M is the one rung where the common-policy and per-arm estimands differ (7.9 against 5.2–7.5 mbpb). (4) The 1.1B rung's token budget spans 1.54 epochs of its corpus; the repetition is disclosed rather than controlled, and its three pairs share the one scouted learning rate. (5) The decode premium is measured on one GPU (H200) with our kernels; other stacks will shift its constants but not its serial-projection cause. (6) Every run uses one tokenizer (GPT-2 BPE) and a 2048-token context, so margins under larger vocabularies or longer contexts are unmeasured. (7) Backbone shape is fixed per rung; whether retuning the baseline's shape buys the same margin at fixed parameters is untested. (8) The comparison with squared ReLU, GEGLU, and ReGLU covers one rung, one learning rate, and two routings; per-variant LR tuning is untested.

10 Related work

Gated FFNs and GLU variants (Shazeer, 2020; Dauphin et al., 2017) established the gate as the site of FFN nonlinearity; Narang et al. (2021) replicated the family's advantage independently. The GLU-activation family (GEGLU, ReGLU) swaps the activation, an axis our factorial holds fixed at SiLU; Section 8 compares both, and Primer's squared ReLU (So et al., 2021), directly at 160M. Work that places nonlinearity elsewhere augments a unit rather than replacing it: GAU's gated attention unit (Hua et al., 2022) and multiplicative-interaction networks (Jayakumar et al., 2020) motivate composition but do not test a parameter-capped nonlinear value bottleneck. Smith (2026) attaches nonlinear low-rank branches in parallel to existing projections at a 4–24% parameter premium; the branch's activation is required for it to exist at all, so its linear ablation is empty by construction, where ours is the substantive null. Wang et al. (2026) keeps SwiGLU's gate and mixes elementwise activations token-adaptively at near-parameter-neutrality ($O(d)$ gating parameters), enriching the nonlinearity while leaving each channel in its original frame. Low-rank factorizations and adapters (Lan et al., 2020; Houlsby et al., 2019; Hu et al., 2022) share the down-then-up shape while remaining linear, frozen-attached, or both. SwiGNLU differs in being the feed-forward unit itself, with channels remixed through the interior activation and the whole construction capped at SwiGLU's budget; the hourglass-FFN line (Liao et al., 2026) makes the topology-versus-width distinction independently relevant. Our baseline is SwiGLU, the unit both MoA and NOBLE also report against; we make no relative claim about either (Appendix D).

11 Conclusion

A gated FFN need not spend its entire value budget on one affine projection. Reallocating it to a nonlinear value bottleneck improves quality in every completed seed pair; the gain survives long budgets and pure AdamW, exceeds squared ReLU, GEGLU, and ReGLU, transfers off-distribution, and reaches the control's quality in 12–15% less wall-clock for a 2.7–5.1% batch-1 decode premium. The retrofit is three lines: cap v_h, keep the tuned LR, route the new matrices with the hidden ones.

AI USE STATEMENT

Large language models were used substantially in this project: as coding assistants for the measurement infrastructure, for statistical verification (with all reported quantities re-derived by released code), and in drafting and revising this text under author direction. The authors verified the code, evidence, citations, and final claims and retain responsibility for them.

ETHICS STATEMENT

This work reports controlled language-model comparisons at up to 1.1B parameters on public web corpora; it introduces no new data collection, human subjects, or deployment. The declared artifact accounts for 10,304 GPU-hours across its 204 runs, produced within a larger exploratory program, and the revision fleet adds 512 GPU-hours across 64 runs; Appendix F states the accounting boundary.

REPRODUCIBILITY STATEMENT

The supplementary material (anonymized for review) contains the SwiGNLU module and its Muon/AdamW recipe (`swignlu.py`); the trainer that produced every run, with every compared feed-forward block (`train.py`); the records of all 268 runs with their complete training curves, and the raw serving measurements (`results/`); and one script, `tables.py`, that regenerates every table in the paper from those records. The tokenized corpora are not redistributed; the supplementary README names their sources and the SHA-256 of each tokenized file.

REFERENCES

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of NAACL-HLT*, 2019.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *International Conference on Machine Learning*, 2017.

Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018. doi: 10.1016/j.neunet.2017.12.012. arXiv:1702.03118.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021.

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

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

Neil Houlsby, Andrei Giurgiu, Stanisław Jastrzębski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning*, 2019.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Weizhe Hua, Zihang Dai, Hanxiao Liu, and Quoc V. Le. Transformer quality in linear time. In *International Conference on Machine Learning*, 2022. arXiv:2202.10447.

Siddhant M. Jayakumar, Wojciech Marian Czarnecki, Jacob Menick, Jonathan Schwarz, Jack W. Rae, Simon Osindero, Yee Whye Teh, Tim Harley, and Razvan Pascanu. Multiplicative interactions and where to find them. In *International Conference on Learning Representations*, 2020.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024. https://kellerjordan.github.io/posts/muon/.

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 *International Conference on Learning Representations*, 2020.

Feng-Ting Liao, Meng-Hsi Chen, Guan-Ting Yi, and Da-shan Shiu. Revisiting the shape convention of transformer language models. *arXiv preprint arXiv:2602.06471*, 2026.

Jingyuan Liu et al. Muon is scalable for LLM training. *arXiv preprint arXiv:2502.16982*, 2025.

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

Sharan Narang, Hyung Won Chung, Yi Tay, William Fedus, Thibault Fevry, Michael Matena, Karishma Malkan, Noah Fiedel, Noam Shazeer, Zhenzhong Lan, Yanqi Zhou, Wei Li, Nan Ding, Jake Marcus, Adam Roberts, and Colin Raffel. Do transformer modifications transfer across implementations and applications? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021. arXiv:2102.11972.

OLMo Team, Pete Walsh, Luca Soldaini, Dirk Groeneveld, et al. 2 OLMo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro von Werra, and Thomas Wolf. The FineWeb datasets: Decanting the web for the finest text data at scale. *arXiv preprint arXiv:2406.17557*, 2024.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

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

Ethan Smith. NOBLE: Accelerating transformers with nonlinear low-rank branches. *arXiv preprint arXiv:2603.06492*, 2026.

David R. So, Wojciech Mańke, Hanxiao Liu, Zihang Dai, Noam Shazeer, and Quoc V. Le. Primer: Searching for efficient transformers for language modeling. In *Advances in Neural Information Processing Systems*, 2021. arXiv:2109.08668.

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*, 2017. arXiv:1706.03762.

Mingze Wang, Jinbo Wang, Yikuan Xia, Kai Shen, and Shu Zhong. More expressive feedforward layers: Part I. token-adaptive mixing of activations. *arXiv preprint arXiv:2605.26647*, 2026.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, 2017.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. TinyLlama: An open-source small language model. *arXiv preprint arXiv:2401.02385*, 2024.

A POST-SEARCH ROBUSTNESS

Table 6 compares each family's lowest-mean replicated explored cell, requiring SwiGNLU to remain at or below the SwiGLU parameter count. These rows answer the tuned-cell question but are post-search and carry no hypothesis-test interpretation.

Table 6: On the 16-window metric recorded for every run, since the 160M rung has no 256-window rescore; the 1.1B row is therefore 6.15 here against 7.08 on the wide metric in Table 3. Positive favors SwiGNLU. Each row is each family's lowest-mean replicated cell under the legality rule; the 305M and 762M controls are the strengthened cells the final-campaign LR study found (their own best measured rates), and the 160M control routes $W_{\downarrow}$ rather than qkv to AdamW, which is why these margins are smaller than the common-policy gains of Table 3: the control here is allowed its best configuration, the treatment must stay strictly under the parameter budget. The 305M treatment is the revision runs' $3072/512$ cell at 2.8×10^{-4} (Appendix G); the main campaign's best cell gives 3v3, 3.52, 4.65 there.

Scale	Arm sizes	Worst-seed separation	Mean separation (mbpb) ↑
160M	5v3	1.97	4.04
305M	4v3	4.31	6.00
599M	3v2	6.70	7.91
762M	2v2	5.43	6.96
1.1B	3v3	2.57	6.15

An earlier wide–shallow 1.1B configuration ($d/L = 439$) is excluded from the ladder as a shape outlier: its runs are not among the released runs, and its treatment arm ran with the dead optimizer carve that Section 3 prices at ∼38% of the effect, so it is diagnostic rather than a ladder test.

Learning-rate brackets. At 599M the brackets separate, the one rung where they do (wide bits/byte, cell means): SwiGLU improves from 0.6569 at the recipe's 3.0×10^{-4} to 0.6543 at 1.5×10^{-4} and degrades again at 1.05×10^{-4} (0.6608–0.6611, two seeds), while SwiGNLU at the parameter-matched geometry improves to 0.6468 at 2.1×10^{-4} (two seeds) and degrades at 1.5×10^{-4}. Each family at its own best measured rate leaves 5.2–7.5 mbpb, from the legal cell to the matched geometry. At 305M the revision fleet closes both brackets: SwiGLU is best at 2.8×10^{-4} against both neighbors, and SwiGNLU at the legal geometry is best at the same rate against both of its neighbors (Table 7).

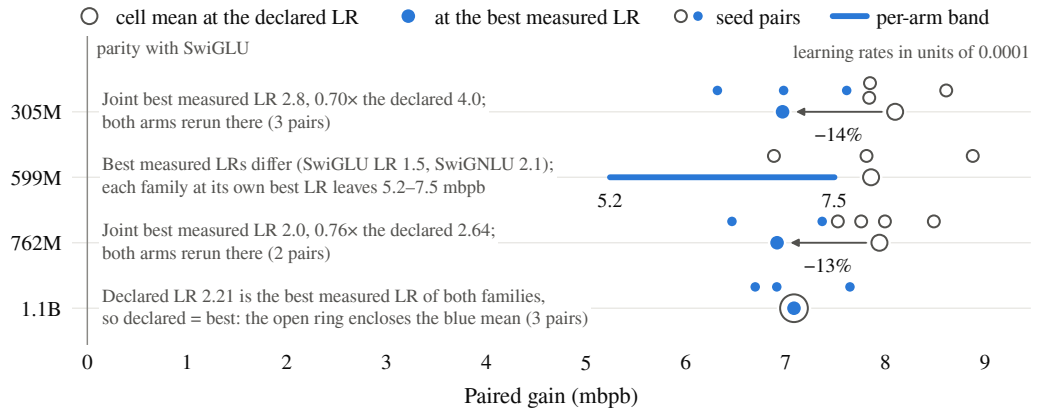

Figure 5: Sensitivity of the paired gain to the shared learning rate. Per rung, the open mark is the cell mean at the declared LR (the common-policy cell of Figure 1) and the filled mark the mean with both arms rerun at the best measured LR; the arrow carries the change, and the small marks are the seed pairs. At 1.1B the two rates coincide, so the marks nest. 599M is the one rung whose families' best measured rates differ, so it carries the measured 5.2–7.5 mbpb per-arm band instead of a joint mark (Section 5; Table 7).

Table 7: Learning-rate brackets under the qkv carve, cell means of 256-window validation. Rows marked † come from the revision fleet (Appendix G); every other row is a main-campaign cell. SwiGLU rows give h alone, since SwiGLU has no v_h. At 599M each family's own best measured rate (SwiGLU at 1.5×10^{-4}, SwiGNLU at 2.1×10^{-4}) leaves 7.5 mbpb; the legal-geometry cell at its best measured LR leaves 5.2 mbpb. The 1.1B scout that located 2.21×10^{-4} ran at 10% of the horizon and is not among the released runs.

Rung	Arm (geometry h or h/v_h)	LR	Seeds	Wide bits/byte
305M	SwiGLU 2752	4.0×10^{-4}	3	0.7119
305M	SwiGLU 2752	2.8×10^{-4}	3	0.7091
305M	SwiGLU 2752†	2.0×10^{-4}	4	0.7123
305M	SwiGLU 2752†	1.4×10^{-4}	4	0.7193
305M	SwiGNLU 2816/704	4.0×10^{-4}	3	0.7038
305M	SwiGNLU 2816/704	2.8×10^{-4}	3	0.7021
305M	SwiGNLU 3072/512 (legal)	4.0×10^{-4}	3	0.7038
305M	SwiGNLU 3072/512 (legal)†	2.8×10^{-4}	4	0.7019
305M	SwiGNLU 3072/512 (legal)†	2.0×10^{-4}	4	0.7071
599M	SwiGLU 3392	3.0×10^{-4}	3	0.6569
599M	SwiGLU 3392	1.5×10^{-4}	1	0.6543
599M	SwiGLU 3392	1.05×10^{-4}	2	0.6610
599M	SwiGNLU 4032/512 (matched)	3.0×10^{-4}	3	0.6490
599M	SwiGNLU 4032/512 (matched)	2.1×10^{-4}	2	0.6468
599M	SwiGNLU 4032/512 (matched)	1.5×10^{-4}	1	0.6499
599M	SwiGNLU 3392/896 (legal)	3.0×10^{-4}	3	0.6490
599M	SwiGNLU 3392/896 (legal)	1.5×10^{-4}	1	0.6513
762M	SwiGLU 3776	2.64×10^{-4}	6	0.6407
762M	SwiGNLU 3776/1024	2.64×10^{-4}	4	0.6327
762M	SwiGLU 3776	2.0×10^{-4}	2	0.6384
762M	SwiGNLU 3776/1024	2.0×10^{-4}	2	0.6315
1.1B	SwiGLU 4288	2.21×10^{-4}	3	0.6160
1.1B	SwiGNLU 4288/1152	2.21×10^{-4}	3	0.6089

B PER-SEED EFFECTS, HELD-OUT PROBES, AND THE DOWNSTREAM BATTERY

Every paired effect. Table 8 lists each seed pair of Table 3 and Section 8. Gains are $1000(\mathrm{bpb}_{SwiGLU} - \mathrm{bpb}_{SwiGNLU})$ on the wide metric where it exists.

Held-out probes. Table 9 scores the controlled FineWeb checkpoints on a separate probe harness (fixed windows per corpus; its absolute level differs from the training-time validation metric, e.g. 0.94 against 0.71 bits/byte for the same 305M model on FineWeb), which is why Section 8 compares relative gains: each paired gain divided by the control's mean bits/byte on that text. Dolmino-Math sits within 0.02 bits/byte of FineWeb on that harness for the FineWeb-trained controls (0.950/0.856/0.824/0.794 against 0.938/0.869/0.845/0.808 at 305M/599M/762M/1.1B), so it is a difficulty-matched held-out corpus; FineMath4 is 0.3–0.4 bits/byte harder. Within the probe harness alone, the relative gain on FineWeb against Dolmino-Math and FineMath4 is 1.08% against 1.87% and 2.20% at 305M, 1.33% against 2.02% and 2.11% at 599M, 1.33% against 1.35% and 1.21% at 762M, and 1.35% against 2.31% and 1.97% at 1.1B. For the Longmino-trained cells Dolmino-Math is not held-out (Longmino contains mathematical text); those cells give in-distribution/FineWeb/Dolmino-Math/FineMath4 gains of 7.0/9.4/6.4/8.4 mbpb at 160M (3 pairs) and 8.8/11.4/8.0/10.4 at 305M (2 pairs) and are outside the count.

Downstream battery. Table 10 gives every cell: byte-length-normalized zero-shot accuracy, seven-task macro, paired within seed, deltas as SwiGNLU minus SwiGLU in percentage points. Pooled over the 17 probed pairs (the 18 of Table 3 less 762M seed 2, whose control checkpoint was not probed) the delta is +0.59 (16 positive), +0.60 with the cell as the unit; the single negative pair is 305M seed 0 (−0.35). Per task over the 12 pairs probed before the final campaign (305M, 599M, one 762M pair, and the two Longmino cells): HellaSwag +1.07 (12 of 12), ARC-E +1.02 (7),

Table 8: All 28 seed pairs of the main campaign; every one favors SwiGNLU. Above the rule are the six matched-budget cells (18 pairs), then the over-training cell and the four token-ladder cells (10 pairs), with the ladder on its in-run held-out-tail windows. Below the rule, the same four ladder cells are rescored on the stratified whole-bin windows of Table 3 (Section 4).

Corpus	Cell	Seeds	Per-seed gain (mbpb)	Mean
FineWeb	305M	0,1,2	8.61, 7.84, 7.84	8.10
FineWeb	599M	0,1,2	7.81, 8.87, 6.88	7.86
FineWeb	762M	0,1,2,3	7.99, 7.52, 8.48, 7.76	7.94
FineWeb	1.1B	1,2,3	7.64, 6.91, 6.69	7.08
Longmino	160M	0,1,2	10.19, 6.46, 8.42	8.36
Longmino	305M	0,1	8.24, 10.97	9.60
FineWeb	305M, 1.31× Chinchilla	0,1	9.97, 10.87	10.42
Longmino	160M, 0.62×	0,1	12.78, 8.37	10.58
Longmino	160M, 2.5×	0,1	11.71, 10.03	10.87
Longmino	160M, 4.9×	0,1	12.39, 12.57	12.48
Longmino	160M, 9.9×	0,1	12.68, 8.93	10.81
Longmino	160M, 0.62×, stratified windows	0,1	8.43, 6.06	7.24
Longmino	160M, 2.5×, stratified windows	0,1	10.11, 7.38	8.74
Longmino	160M, 4.9×, stratified windows	0,1	11.15, 10.56	10.85
Longmino	160M, 9.9×, stratified windows	0,1	10.40, 7.99	9.20

Table 9: Paired SwiGNLU gain in mbpb, and in parentheses as a percentage of the control's bits/byte on that text. The in-distribution column is the wide-validation metric of Table 3; the two held-out columns are the probe harness, whose absolute level differs; the text gives the within-harness comparison at every rung. The relative held-out gain exceeds the relative in-distribution gain in seven of eight cells. At 762M the probes cover control seeds 0, 1, and 3 but not seed 2, so that cell holds three of the four pairs of Table 3; over those three the in-distribution gain is 7.8 (1.21%), level with FineMath4.

Cell	Pairs	In-distribution	Dolmino-Math	FineMath4
305M	3	8.1 (1.14%)	17.8 (1.87%)	28.8 (2.20%)
599M	3	7.9 (1.20%)	17.3 (2.02%)	25.2 (2.11%)
762M	3	7.9 (1.24%)	11.1 (1.35%)	14.1 (1.21%)
1.1B	3	7.1 (1.15%)	18.4 (2.31%)	22.4 (1.97%)

PIQA +0.50 (9), ARC-C +0.47 (7), SciQ +0.29 (8), BoolQ +0.26 (9), WinoGrande +0.02 (7). LAMBADA, probed as raw accuracy outside the macro, moves +1.45 (12 of 12). Only HellaSwag is resolvable on its own; the other tasks' seed-to-seed spread (0.5–1.8 points) exceeds their effects, which is why the paper claims the paired macro direction and not per-task magnitudes.

Table 10: Downstream battery per cell: byte-length-normalized zero-shot accuracy, seven-task macro, in percentage points. *Control* is the SwiGLU mean; deltas are SwiGNLU minus SwiGLU within seed. Sixteen of seventeen probed pairs are positive; the pooled mean is +0.59. The 762M row holds seeds 0, 1, and 3, because the seed-2 control was not probed.

Corpus	Cell	Pairs	Control macro	Per-seed Δ	Mean Δ
FineWeb	305M	3	48.1	−0.35, +0.06, +0.84	+0.18
FineWeb	599M	3	51.4	+0.95, +1.07, +0.31	+0.77
FineWeb	762M	3	52.9	+0.62, +0.84, +1.13	+0.86
FineWeb	1.1B	3	55.4	+0.69, +1.05, +0.63	+0.79
Longmino	160M	3	42.4	+0.40, +0.05, +0.41	+0.29
Longmino	305M	2	44.5	+0.92, +0.45	+0.69

C TRAINING RECIPE AND FACTORIAL DETAILS

Shared recipe. Every run in the corpus uses linear warmup over 2,000 steps to its peak learning rate, then cosine decay to $0.1\times$ the peak; AdamW with $\beta = (0.9,\ 0.9999)$ and $\epsilon = 10^{-8}$ on embeddings, head, one-dimensional parameters, and the qkv projection, with decoupled weight decay 0.1 on all but the one-dimensional parameters; Muon on every other matrix with Nesterov momentum 0.95, five Newton–Schulz iterations in bfloat16, the RMS-matching update scale $0.2\sqrt{\max(m, n)}$ of Liu et al. (2025) so one learning rate serves both branches, and the same decoupled weight decay; and global gradient-norm clipping at 1.0. Global batch is 24/16/16/16/32 sequences of 2,048 tokens by rung (Table 2).

Data. Both corpora are tokenized once with GPT-2 BPE into a flat `uint16` bin that every run reads in the same seeded order. FineWeb (Penedo et al., 2024) measures 4.16 bytes/token on its validation windows; Longmino is the Allen Institute's public 204-domain `dolma3_longmino_mix` release, 86.4B tokens at 2.66 bytes/token, which is the corpus-specific factor behind every Longmino bits/byte figure. Neither bin is redistributed. The artifact names each source and records the SHA-256 of each tokenized bin (FineWeb `fbd9999e73ce85da...`, Longmino `b1bd3524d892e5eb...`), so a replication rebuilds the bins and verifies them before training.

Table 11 states what the four arms of Figure 3 hold fixed and what they are allowed to move. Within each condition the bottlenecked arms carry the FFN hidden width that holds total parameters closest to the full-width arms, so the two rows of the 2×2 differ in topology rather than in budget.

Table 11: The declared value-path 2×2 cells, from `runs.csv`. The bottleneck rows spend the parameters the factorization frees on FFN width, which is why the 599M replication identifies the bottleneck together with its budget reallocation (Section 7). The 599M arms ran in two hardware groups, which is what the wall-clock spread records.

Held fixed within a condition	305M	599M
Learning rate	4.0×10^{-4}	3.0×10^{-4}
Bottleneck rank v_h	704	512
Optimizer	Muon, $\beta_2 = 0.9999$, AdamW carve on qkv	
Seeds per cell	3, in all four cells	
Varies by row		
FFN hidden width h, full width	2752	3392
FFN hidden width h, bottleneck	2816	4032
Total parameters, full width	305.34M	598.61M
Total parameters, bottleneck	305.60M ($+0.086\%$)	599.01M ($+0.066\%$)
Wall-clock per run	12.3–12.5 h	17.5–25.6 h

Per-seed interactions, on the improvement convention of Table 5, are in Table 12. The positive sign in all nine matched quadruples is the replication fact; the released run table holds each arm's raw values. An affine-deviation probe is consistent with the mechanism: under the qkv carve the trained SwiGNLU value path is 2.3–2.5$\times$ further from its best affine approximation than the full-width SwiGLU-dualact arm ($2.1\times$ without the carve), while SwiGLU and SwiGLU-lowrank are exactly affine. It is a usage diagnostic, not causal evidence.

Table 12: Per-seed interaction in mbpb (↑, positive is super-additive), one matched quadruple per column.

Condition	Seed 0	Seed 1	Seed 2
305M, no carve	2.65	3.52	7.60
305M, qkv	4.86	2.12	2.46
599M, qkv	1.32	3.20	3.28

D CLOSEST ALTERNATIVES

MoA mixes elementwise activations token-adaptively; NOBLE adds nonlinear low-rank branches beside existing projections. Each is reported by its authors against SwiGLU under its own protocol,

as SwiGNLU is here, and the three change different parts of the block: the activation mixture, an added branch, and the value path. We compare against the ubiquitous baseline and make no relative empirical claim about MoA or NOBLE; a comparison among the three would need shared code, optimizer, LR-selection budget, tokens, validation, and verified parameter and FLOP accounting.

E SERVING BENCHMARK METHODOLOGY

Each cell benchmarks the full end-to-end model, single GPU, exclusive: CUDA events with one synchronize per repetition, explicit warmup excluded from statistics (compilation confined to warmup), peak-memory counters reset per configuration, decode as genuinely incremental single-token steps against a preallocated KV cache checked against a full forward pass (largest logit difference 1.8×10^{-7}), fixed decode context so samples form a distribution, per-record raw samples, reversed-order replicates against drift, and bootstrap confidence intervals on every reported ratio. Both arms run fused and unfused, compiled and eager; every record carries hardware, versions, dimensions, and status, and failed cells are visible rather than absent. Kernel fusion of the stacked projections pays only under `torch.compile` (about -3% step time for both families); fused without compilation is 1–4% slower for both, so the released module is unfused, as in training.

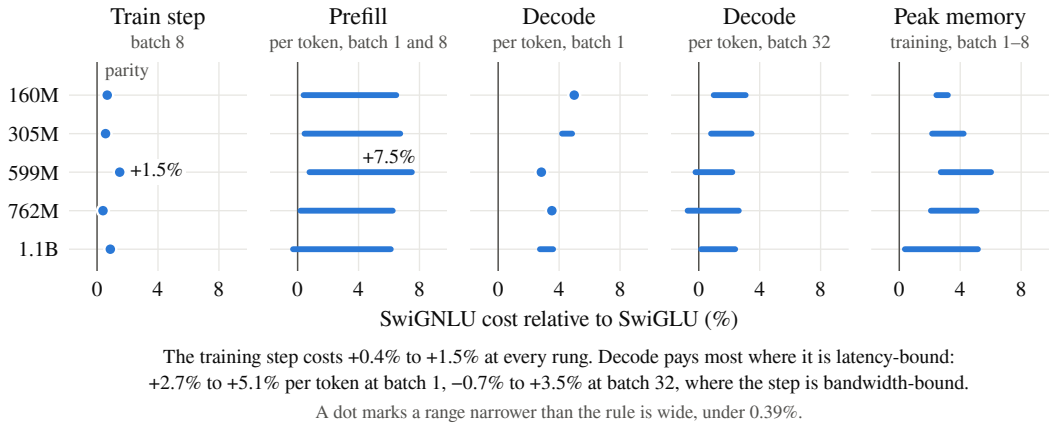

Figure 6: The serving bill of Table 4 as ranges: SwiGNLU cost relative to SwiGLU in percent, per rung, each arm at its production configuration. A rule spans the measured range across a measure's configurations (prefill over sequence lengths 512–4096 at batch 1 and 8, decode over 512- to 8192-token contexts, memory over training batch 1–8); a dot marks a range narrower than the rule is wide. Zero is parity and all five panels share one scale; the two labelled values are the largest measured costs in training and in serving.

F COMPUTE AND WALL-CLOCK ACCOUNTING

The declared corpus contains 204 unique runs totaling 10,304 GPU-hours (elapsed time times GPU count); GPU count is read from checkpoint headers for 203 runs and from the launch log for one. The 39 runs of the final confirmation campaign (the 1.1B pairs, the token-ratio ladder, the over-training controls, and the learning-rate verification and bracket runs) each occupied one eight-GPU node and contribute 3,653 GPU-hours. That campaign ran inside a 4,608 GPU-hour allocation (48 H200s for 96 hours) that also funded the learning-rate scout, the serving matrix, and final scoring. Historical `wall_h` is not a cross-run throughput metric (world size varies and the field resets on resume); every wall-clock statement in this paper comes instead from the co-scheduled serving benchmark and same-allocation cohorts recorded with end-to-end elapsed time. The revision fleet of Appendix G adds 64 single-GPU runs totaling 512 GPU-hours (412 at 305M, 100 at 160M), outside the declared corpus.

G REVISION FLEET: OPTIMIZER, LEARNING-RATE BRACKET, AND FFN VARIANTS

For this revision we trained 64 further runs on one eight-GPU H200 node, each on a single GPU with the same trainer, the batched Newton–Schulz Muon of the main campaign, 256-window validation, and the full token budget of its rung (4.4B tokens at 305M, 2.0B at 160M); they total 512 GPU-hours. Every comparison below is paired within the fleet except the 305M best-LR gain of Section 5, whose control cell comes from the main campaign. The fleet reproduces the main campaign's 160M SwiGNLU runs to within 0.18 and 0.39 mbpb (seeds 1 and 2, 256-window), which bounds the offset between campaigns. All 19 within-fleet pairs of SwiGNLU against SwiGLU favor SwiGNLU. Twelve runs were relaunched once after a startup failure. Their records and curves are in the released run table, and `tables.py` prints every table in this appendix. The fleet's 305M learning-rate cells are in Table 7; the other two experiments follow.

Table 13: Pure AdamW at 305M: AdamW on every parameter with $\beta = (0.9,\ 0.95)$, $\epsilon = 10^{-8}$, and weight decay 0.1, with warmup, schedule, batch, and tokens as in Appendix C. SwiGLU at h=2752 (305.3M) against SwiGNLU at $3072/512$ (304.3M), three seeds per cell; cell means of 256-window bits/byte, gains in mbpb. SwiGNLU wins all nine pairs, and its seeds agree more tightly, spanning 0.8, 0.1, and 0.7 mbpb against SwiGLU's 3.5, 4.0, and 1.8.

LR	SwiGLU	SwiGNLU	Per-seed gain (seeds 0, 1, 2)	Mean gain ↑
2.8×10^{-4}	0.7420	0.7386	3.79, 1.65, 4.71	3.38
4.0×10^{-4}	0.7427	0.7388	5.36, 1.28, 5.10	3.91
5.6×10^{-4}	0.7464	0.7409	5.23, 5.00, 6.11	5.45

Table 14: FFN variants at 160M on FineWeb, Muon at 6×10^{-4}, three seeds each: improvement over SwiGLU in mbpb on 256-window bits/byte (positive favors the arm), per seed in parentheses. The *qkv* routing puts the qkv projection on AdamW, the paper's protocol; *down* puts $W_{\downarrow}$ on AdamW instead. SwiGLU scores 0.7766 under qkv and 0.7734 under down. Squared ReLU is ungated at h=3072. *Under the down routing the fragment matched no parameter of the ungated MLP, and the optimizer log reports 84.9M Muon parameters, the uncarved count, so that arm trained with every hidden matrix on Muon: the silent routing failure of Section 3, which makes its value no routing-matched comparison. SwiGNLU minus squared ReLU under qkv is +3.02, +1.69, +3.22.

Arm	Params	qkv routing	down routing
SwiGNLU	160.6M	+8.29 (9.10, 7.76, 8.02)	+3.11 (1.31, 5.64, 2.37)
Squared ReLU	162.1M	+5.65 (6.08, 6.06, 4.81)	−0.95*
GEGLU	162.1M	−1.56 (−2.08, −1.31, −1.29)	−1.09 (−4.28, −0.17, +1.18)
ReGLU	162.1M	−5.29 (−5.06, −5.34, −5.48)	−0.08 (−3.55, +3.42, −0.10)