Endogenous Signals Fail to Allocate: Per-Token Compute Value Is Real, Transient, and Not Learnable

Anonymous authors
Paper under double-blind review

Abstract

Token-level adaptive compute usually keys off endogenous signals (attention entropy, router scores, confidence) assumed to track per-token difficulty. On arithmetic with graded difficulty we show this assumption fails three ways: an entropy-driven gate collapses to a constant, a tag-initialized gate inverts, and an unsupervised learned head is vacuous. Difficulty-supervised targets under a budget constraint do produce ordered gates, but a pre-registered shuffled-targets control shows the ordering carries no accuracy benefit at 11M (at most 0.3 accuracy points), and the 1.4B replication differs from permuted targets by 0.0008 held-out loss. Measuring the field directly, the per-token gradient of loss with respect to each token's compute gate, explains why: the field has a life cycle (real structure mid-training, consumed by training, flat at convergence), and a causal test shows its structure is not trainable: chasing it online loses to random allocation of the same budget at every scale tested. The surviving recipe needs no allocator: token-level FFN dropout, where each token receives full FFN compute only inside randomly rotating windows. At 1.4B web-text fine-tuning it trails dense by 7.79% held-out loss (3 seeds) at 50% billed FFN training FLOPs, inside the pre-registered 2–10% band; a gather-based realization converts half the billed cost into a measured 1.35× training throughput. Failures on code and template-dominated math are reported.

Transformers spend most of their training compute on feed-forward blocks applied uniformly to every token at every step; does every token need the FFN at every step? The adaptive-compute literature answers with learned allocators (routers, entropy gates, difficulty heads) trained end-to-end to switch compute on where it "matters." We first show this program fails on its own terms in a controlled testbed with per-token ground truth: endogenous signals collapse, invert, or stay vacuous (§2.2), supervised targets produce correctly ordered gates whose ordering carries no accuracy benefit, and a pre-registered shuffled-targets control shows target content is irrelevant at 11M and 150M scale. Figure 1 previews the three findings.

The positive half of the paper is a measurement that explains these failures and a recipe it motivates. We measure the per-token *value of compute*, the gradient of the loss with respect to each token's FFN gate, and find it has a life cycle: real structure mid-training, consumed by training, flat at convergence (§2.4). An oracle bounds the converged field's headroom, and a causal test shows that *chasing* the field online is worse than random allocation of the same budget, at 11M and, replicated, in the starved-150M regime where the field's mid-life structure is demonstrably present (§4). What works is the degenerate allocator everyone skips: train with *token-level FFN dropout* (each token receives full FFN compute only inside randomly rotating windows) and deploy dense. At the 150M convergence pair it trails dense held-out loss by 5.3% at exactly half the billed FFN training FLOPs, and by 7.79% on 1.4B web-text fine-tuning (3 seeds); we scope these claims to compute-limited regimes per their pre-registered decision rules. It requires no labels or routers and composes with any attention architecture because it never touches attention.

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

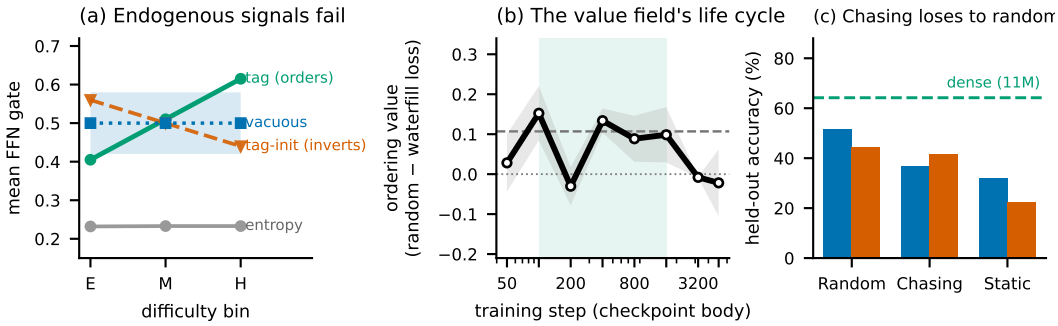

Figure 1: The paper in three panels. **(a)** Endogenous signals fail: entropy gates are flat (collapse), tag-init end-to-end gates decrease with difficulty (inversion; schematic of measured $r = -0.59$), unsupervised heads are flat and seed-unstable (vacuity, band); only tag supervision orders gates, with no accuracy benefit (§4.2). **(b)** The measured value of compute (Eq. (3)) has a life cycle: permutation-controlled ordering value on frozen bodies, real mid-training, flat at convergence. The gray envelope is the seed spread, the black line the seed mean; the green span marks the plastic window (mean ordering value > 0); the dashed line is the *entire* dense-vs-dropout gap the recipe pays (0.107), and the peak ordering value is $\sim 2\times$ it: consumed by training, never converted. **(c)** Causal test at two scales: chasing the measured field online loses to random allocation of the same budget; a constant gate collapses. Blue bars: 11M; vermillion: starved-150M; green dashed: the dense 11M reference (values in Table 1).

Contributions.

1. **A three-way failure diagnosis** of endogenous allocation signals (collapse, inversion, vacuity), each with a mechanistic account (§2.2); a standard Mixture-of-Depths router fails analogously.
2. **Compute-value supervision**: targets derived offline from the trained model itself (per-token loss, and $\partial\mathcal{L}/\partial g$ in one backward pass) match label-supervised allocation, with a shuffled-targets control (and a pre-registered decision rule) separating the specific measured targets from supervised variation alone (§2.4, §4.2); the value field's per-bin signature is measured at 42M and 150M, where the pre-registered migration prediction fails and the converged field is flat (§4.4).
3. **Iso-FLOPs methodology**: all method comparisons at matched *billed training FLOPs* (gated billing, forward + backward), with within-segment interpolation and a budget sweep; the "similar total budget" caveat of an earlier workshop version of this work is retired (§2.5).
4. **Scale replication, honestly scoped**: the failure taxonomy, the shuffled-control content-irrelevance result, and the causal field-chasing test all replicate at 150M (probe grid, starved regime); the dropout recipe's parity claim is scoped to compute-limited regimes by its pre-registered decision rule (-5.3% loss at 50% billed FFN FLOPs at 150M; -7.79% at 1.4B web-text fine-tuning; the template-dominated-data boundary is reported, not hidden) (§4.4, §4.5).
5. **Realization**: a tiered-FFN serving benchmark showing tokens/sec scaling with the mean gate, converting billed-FLOPs savings into wall-clock (§4.7).

1 Related Work

Adaptive computation and early exit. Adaptive Computation Time (Graves, 2016) and its transformer descendant (Dehghani et al., 2019) learn per-input halting and are known for halting scores collapsing to constants; our collapse diagnostic is the per-token version of the same failure, with the same underlying cause: weak, task-loss-only feedback (Bengio et al., 2013). Early-exit and cascades (Teerapittayanon et al., 2017; Xin et al., 2020; Elbayad et al., 2020; Schuster et al., 2022) and dynamic routing (Wang et al., 2018) key

compute depth to learned confidence signals; the confidence-versus-value gap we measure (easy tokens carry the largest loss gradients) is a training-time account of why confidence signals under-serve hard inputs (Guo et al., 2017). Mixture-of-Depths (Raposo et al., 2024) is the strongest recent allocator; our MoD baseline's router trains during a short fine-tune rather than a full pretraining run, bounding router-from-scratch at our budget (§4.3 reads it directly).

Conditional computation and mixture-of-experts routing. Sparse MoE (Jacobs et al., 1991; Shazeer et al., 2017; Lepikhin et al., 2021; Fedus et al., 2022) attaches learned routers to expert selection, and router training is notoriously brittle: load-balancing losses, router z-losses, and initialization are load-bearing (Zoph et al., 2022), and gradient estimation through discrete routing decisions is a recognized obstacle (Bengio et al., 2013). Our difficulty heads inherit this feedback poverty (the vacuity diagnostic). Slimmable networks (Yu & Huang, 2019) switch capacity at deployment; our budget-pinned gates are the per-token analogue.

Regularization as the null allocator. The surviving recipe is structured dropout, a lineage we position against explicitly rather than claim novelty over: elementwise dropout (Srivastava et al., 2014), stochastic depth (Huang et al., 2016), LayerDrop's per-layer structured dropout (Fan et al., 2020), and random token dropping during language-model training (Fan et al., 2021). The contribution is not the regularizer but the iso-FLOPs result that surrounds it: at matched billed compute, every signal-carrying allocator we test is matched by (or loses to) the random one, and the measured value field that would justify non-random allocation is not trainable online. Our variant also differs in granularity (per-token, per-layer FFN windows; attention untouched) and in deployment (dense inference, never gated).

Saliency and the value of compute. Eq. (3) is a per-token, per-layer saliency score for *compute* rather than weights, in the lineage of Optimal Brain Damage (LeCun et al., 1990) and Taylor-expansion pruning (Molchanov et al., 2019); to our knowledge its life cycle (real mid-training, consumed by training, flat at convergence) has not previously been measured for token-level compute allocation. The same first-order quantity underlies draft-model disagreement in speculative decoding (Leviathan et al., 2023; Chen et al., 2023): acceptance rates measure "hard for a small model"; our gate gradients measure "helpable by this layer," for layers instead of models.

Token importance at inference. Attention-mass and heavy-hitter token selection (Zhang et al., 2023) and token pruning/merging in vision (Rao et al., 2021; Bolya et al., 2023) exploit stable importance structure at inference; our converged-field result explains why such signals must be architectural (attention mass), not gradient-based.

Selection-based counterexamples. Training-time token selection does work at scale: Rho-1 trains only on tokens with excess loss against a frozen reference model, with strong math results at 1B/7B (Lin et al., 2024), and token dropping removes loss-identified unimportant tokens during BERT pretraining for $\sim$25% savings (Hou et al., 2022). Both are consistent with our claims, and the boundary matters: they use a *fixed external reference* to select *data tokens*, never feeding scores back as gate targets into the same body's training loop, so the chaser-destroys-signal pathology we measure (§4.3) does not apply; our negative result covers self-referential gradient signals for FFN-compute gating inside one model.

Controls and memorization. The shuffled-targets control follows the logic of random-label controls (Zhang et al., 2017): if permuted supervision matches real supervision, the specific content is not what carries the effect. Curriculum ordering (Bengio et al., 2009) is the supervised-allocation hypothesis our tag arm tests, at token rather than example granularity.

2 Method

2.1 Architecture and per-token gating

Each transformer layer (Vaswani et al., 2017) computes

$$x_2 = \mathrm{LN}\big(x_1 + \mathrm{attn}(x_1)\big), \qquad x_3 = \mathrm{LN}\big(x_2 + g_t \cdot \mathrm{FFN}(x_2)\big), \tag{1}$$

where $g_t \in [0, 1]$ is a per-token, per-layer gate produced by a small per-layer *DifficultyHead* (a two-layer MLP on the post-attention state x_2, sigmoid-squashed). Compute is billed proportionally: a position gated at g is billed g times the per-token FFN cost; attention is always paid in full. All variants share Eq. (1) and the accounting, so comparisons isolate the gating signal. We gate the FFN only, never the attention's keys or values: skipping a token's FFN is safe because downstream tokens still attend to it, whereas skipping K/V irreversibly deletes information that later positions may need.

2.2 Why endogenous signals fail: three diagnostics

End-to-end training offers the gate one channel of feedback: the task loss. *Collapse*: a signal with no difficulty information (attention entropy (Clark et al., 2019), used here as a diagnostic rather than a proposed allocator, $r \approx -0.06$ per token with difficulty) drives the gate to a constant. *Inversion*: a ground-truth initialization makes the gate vary, but easy problems are memorizable, so their cross-entropy gradient is largest and training allocates them *more* compute (per-problem gate–difficulty $r = -0.59$). *Vacuity*: a head trained only through the task loss receives no signal about what difficulty is; the gate stays flat and unstable across seeds. We report the two iteration-1 quantities as distinct: per-token entropy–difficulty correlation $r \approx -0.06$, and per-problem entropy-gate–difficulty $r = 0.23$ (the gate passes the tag embedding through a bottleneck; the two correlations measure different objects).

2.3 Explicit difficulty supervision with a budget constraint

Training is two-stage. **Stage 1** trains the body conventionally with the gate fixed at 1 and checkpoints it. **Stage 2** attaches the DifficultyHeads and minimizes

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}} + \lambda_{\mathrm{mse}} \tfrac{1}{N} \sum_t \big(\sigma(s_t) - \tau_t\big)^2 + \lambda_{\mathrm{budget}} \big(\bar{g} - 0.5\big)^2, \tag{2}$$

with $\lambda_{\mathrm{mse}} = 5.0$, $\lambda_{\mathrm{budget}} = 0.5$; the body trains at a near-frozen learning rate ($10^{-2}\times$ the heads'). Stage 1 is shared by every fork of the method (all Stage-2 arms load the same checkpoint per seed), so method comparisons are paired by body.

2.4 Measuring the value of compute

Where difficulty labels are absent (all natural-language settings), targets τ_t are *measured offline* from the Stage-1 checkpoint:

$$\mathrm{score}_t \;=\; -\sum_{\ell} \frac{\partial \mathcal{L}}{\partial g_{t,\ell}}\Big|_{g=1}, \tag{3}$$

the marginal loss reduction from one unit of FFN compute at token t, obtained for all tokens and layers in a single forward–backward pass with gates as differentiable leaves. Scores ≤ 0 are *irreducible* (compute does not help there) and receive the floor target 0.1; positive scores are quartile-binned to 0.2/0.4/0.6/0.8. This is the paper's key conceptual move: tags label *difficulty*; Eq. (3) measures the *value* of compute. The two are different objects, and the difference is testable: score–loss correlation is near zero in our runs (loss says "hard," the gate gradient says "helpable"), and a token's tag bin does not constrain its score (a hard bin mixes irreducible tokens with worth-treating ones). The inversion, end-to-end training overfeeding easy tokens, is the same phenomenon seen from the other side: training chases reducibility wherever it lives, but uncontrollably; the budget term is what makes chasing value safe. A loss-quantile variant (the proxy-difficulty ablation) bins tokens by per-token loss; a *shuffled* control permutes targets within each sequence, testing whether any varying supervision would do.

First-order oracle allocation. The same measurements give a reference point: sorting per-(token, layer) scores and filling until the budget is exhausted is the first-order-optimal allocation under a *frozen* body, evaluated zero-shot by overriding gates at inference; it bounds what allocation alone buys without retraining. Trained methods may exceed it (their bodies adapt to their gates), which we read as adaptation compounding allocation, not as a paradox.

2.5 Iso-FLOPs training protocol

Every run logs cumulative *billed training FLOPs*: the gated billing of §2.1 applied to forward, with backward billed at twice forward (total 3×). Methods are compared at matched achieved budgets by linear interpolation of the evaluation history *within* training segments only: two-stage runs have a kink at the stage boundary, and budgets common to all methods are capped at the shortest run's endpoint (no extrapolation). At matched billed FLOPs, lower-mean-gate methods receive more optimizer steps per billed dollar; that is the deployment claim under test, and steps-matched comparisons are reported alongside in Appendix A.

3 Experimental setup

Arithmetic. Two regimes with graded difficulty: *tagged* ([E]/[M]/[H] prefixes; easy 1–2-digit ±, medium 4–5-digit ±, hard 4×4, 5×4 and 6×5-digit multiplication at scale) and *mixed* (no tags; operands 1–7 digits; bin = larger operand's digit count). The hard regime is chosen per scale so the largest model sits inside a difficulty window: hard-bin accuracy neither saturated nor collapsed (maintained in $[40\%, 85\%]$ by the calibration probes). Frozen held-out sets, identical across methods and seeds; accuracy on answer tokens only. Models: an 11M testbed (the earlier workshop setting, exactly reproduced) and a 150M scaled model (RoPE (Su et al., 2024), flash attention (Dao et al., 2022), bf16, cosine schedule; identical across methods), probed healthy (10k steps) and starved (2.5k steps).

Natural language. Pretrained Pythia-1.4b (Biderman et al., 2023) continue-trained on web text (FineWeb-Edu (Penedo et al., 2024), 350M packed tokens, sequence length 2048, AdamW at 10^{-5} with cosine decay and 2% warmup, bf16 autocast over fp32 master weights): a dense arm and a token-level-FFN-dropout arm, same data and step budget, the recipe pair of §4.3 at language scale. The oracle machinery of §2.4 and the code/math domains (CodeSearchNet (Husain et al., 2019), MetaMathQA (Yu et al., 2024)) are implemented; the oracle was not run at NL scale. Metric: full-held-out loss; the rotation arm is reported at dense deployment (gates off), its pre-registered evaluation mode.

Methods compared. *Baseline* (dense; = Stage-1), ours (*gate-grad* targets; *loss-quantile* ablation), *shuffled* control, *Fixed-schedule* (static g=0.5), *MoD* at matched capacity, and the endogenous iterations (*entropy*, *random*, *tag-only*) at testbed scale.

4 Results

4.1 Endogenous signals fail; supervision fixes allocation

The three failure modes are already visible in the 11M testbed runs (ported from an earlier workshop version; Limitations (ii)): entropy 0.232/0.233/0.233 across E/M/H; tag-init inverted; unsupervised vacuous; supervised spread +0.21, ordering $r = 0.95$ in every seed.

4.2 Measured value matches labels at the testbed

Pre-registered decision rule. Written before any shuffled number existed: the measured-targets claim survives iff gate-grad beats the shuffled control by $\geq$1.0 accuracy point, paired by seed at matched billed training FLOPs, in all three seeds; no post-hoc reinterpretation either way.

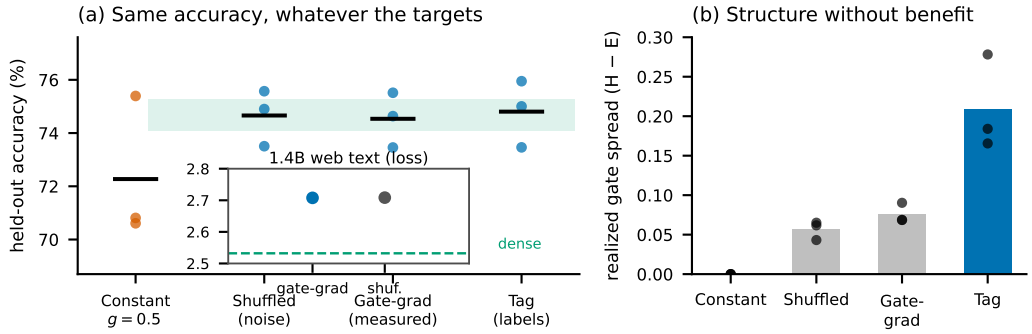

Figure 2: The content-irrelevance punchline, both scales. **(a)** 11M Stage-2 arms at an identical inference budget (∼19 MFLOPs/token; the oracle pass is billed to both measured arms); dots are individual seeds, all arms forking from the same Stage-1 body per seed. True-label, measured-value, and *noise* targets land in one green tie band (max paired difference 0.3 points); a constant gate in the same protocol sits 2.4 points below. *Inset*: the 1.4B replication, gate-grad vs. shuffled held loss ($|\Delta| = 0.0008$; the 1.0-point rule was pre-registered in accuracy at 11M and has no NL loss-domain twin; the two dots sit at nearly the same height). **(b)** Realized hard-minus-easy gate spread: only the tag arm builds difficulty-ordered gates ($r = 0.95$), structure without accuracy benefit. Numbers: Table 4 (Appendix C).

Table 1: The causal test, both scales: the gate is live from step 0 at budget 0.5, all arms at matched billed training FLOPs (one seed per arm; dense/dropout not run at starved-150M). The starved-150M regime is where the field is structured mid-life (waterfill beats random by +0.30/+0.38 across seeds). Dropout and random windows are indistinguishable; chasing the measured field online loses to random at both scales; a constant gate collapses.

	11M (3000 steps, 1 seed)		starved-150M (2500 steps, 1 seed)	
Arm	acc (%)	loss	acc (%)	loss
Dense (reference)	64.2	1.167	–	–
Token-level FFN dropout	51.3	1.275	–	–
Random windows	51.6	1.274	44.2	1.342
Online field-chasing	36.7	2.253	41.7	1.363
Static g=0.5	31.8	2.621	22.4	3.597

Outcome: the rule fired against the measured targets. Paired gate-grad − shuffled accuracy across seeds: −0.3, −0.1, and −0.0 points (Figure 2a). At testbed scale, *any* varying supervision under the budget constraint (even targets permuted into noise) recovers the same accuracy (the seed-paired lines of Figure 2a never separate) while all three dominate a constant gate by 2.4 points (Appendix E). Two honest readings follow. First, at 11M supervision supplies a dense-then-gate protocol whose accuracy is insensitive to target content; the tag arm's +0.21 ordering spread is real but not accuracy-load-bearing. Second, the value-of-compute concept is not refuted but falsifiable at scale: its per-bin signature must migrate with capability (§4.4), and its natural-language targets must beat shuffled ones precisely where compute has leverage (§4.5). Both tests were pre-registered; neither favored the measured targets. Gate–score agreement on held-out tokens is near zero in every seed (Pearson ≤ 0.04, Spearman ≤ 0.09): gate-grad varies within problems, so per-bin spread (Figure 2b) is not its allocation metric. One caveat bounds the conclusion: the pinned budget compresses realized spreads to 0.06–0.21, so *measured* content is tested at modest realized ordering; the load-bearing leg is the tag arm, which does realize difficulty-ordered gates and still ties noise. Whether measured-value ordering buys accuracy at larger realized spreads (weaker λ_{budget}) is open.

Table 2: 150M probe grid: 6 arms × 2 seeds at matched budget (mean±std). Dense wins the healthy recipe; every gated 2-stage arm ties regardless of target content; MoD ties dense with an *anti-ordered* router.

Arm	Acc	Balanced	MFLOPs/tok	$\bar{g}$	gate spread
Dense (reference)	68.0±7.0	77.1±7.5	467±0	1.00	0.00±0.00
Fixed g=0.5 (from scratch)	62.7±1.3	71.6±1.0	265±0	0.50	0.00±0.00
Fixed (2-stage)	62.9±8.9	73.0±9.1	273±0	0.52	-0.02±0.01
Ours (tag targets)	62.9±9.1	72.9±9.3	276±1	0.53	0.19±0.00
Shuffled (2-stage)	62.9±8.8	73.0±9.0	271±3	0.51	-0.02±0.01
MoD (single-stage)	67.5±6.1	77.2±6.1	269±0	0.25	-0.05±0.01

4.3 The causal test and the surviving recipe

Table 1: token-level FFN dropout and random 100-step windows are indistinguishable (1.275 vs 1.274 held loss): the mechanism is the noise-regularization of intermittent full compute, not any allocation signal. The two arms are temporally white (an i.i.d. Bernoulli(0.5) mask per step) and temporally correlated (a random mask held for 100 steps) noise respectively, so their tie rules out schedule structure; both are structured dropout (Srivastava et al., 2014; Huang et al., 2016; Fan et al., 2020) at FFN granularity. Online field-chasing against the measured value field is much worse than random (2.253 vs 1.274), and a constant gate collapses (2.621). Dense wins dense-inference at this horizon (1.167); the question is what half-FLOP training buys at convergence.

The 150M convergence pair. At 10k steps, the rotation-trained model holds held-out loss 1.086 vs dense 1.032 (+5.3%) at exactly 50% of the billed training FLOPs; the pre-registered rule (within ∼2%: parity advances; 2–10%: scoped) puts this in the scoped band.

Table 2: content-irrelevance replicates at 150M: tag, shuffled, and fixed 2-stage arms are accuracy-identical (62.9 each) and dense wins the healthy recipe (68.0). MoD is the strongest apparent counterexample to our framing and deserves a direct reading: at 0.58× billed compute it ties dense (67.5), beating every two-stage arm at matched budget. The bound: this is the 5000-step probe grid (our parity claim is the 10000-step convergence pair, where MoD was not run); n=2 with seed-paired diffs +8.9/−9.7 leaves it indistinguishable from dense; and its anti-ordered router (per-bin gates 0.25/0.24/0.21) means the efficiency is not difficulty-correct allocation. We read it as extending content-irrelevance to a production allocator; it deploys gated where we deploy dense, and its edge at the convergence horizon is the open question.

Table 1 is the decisive test. The starved-150M regime is where the value field is demonstrably *alive*: the discriminator shows real ordering structure (waterfill +0.30/+0.38 over random across two seeds). If that structure were exploitable, training online against the measured field should beat random allocation. It does not: online loses to random windows (41.7 vs 44.2), directionally matching the 11M causal result at a second scale (single seed per arm), and the static gate collapses outright (22.4, held loss diverging). Figure 1(b) shows the full trajectory: the ordering value rises, peaks, and decays to zero as the body converges (the step-5000 sign flips are seed-inconsistent and pre-registered as noise). The life cycle and the causal test together: the field's structure is real, measurable, and not a training signal.

4.4 Scale: the effect survives 11M → 150M

Pre-registered readout: value migrates with capability. An 11M model finds most hard-bin tokens irreducible; a 400M model would not. Prediction (written before the probes ran): the hard bin's mean gate-grad score rises and its fraction of non-positive scores falls toward the medium bin's, across 42M → 150M → 400M; a flat signature at $10 of probe cost would have falsified the mechanism cheaply.

The rule fired against the prediction. On healthy bodies (42M acc 0.496, 150M acc 0.563), the measured field is flat at convergence: frac(score≤0) ≈ 0.51 across E/M/H bins at both sizes, mean |score| at 10^{-6} (coin-flip; Table 7); no difficulty-bin structure, no migration.

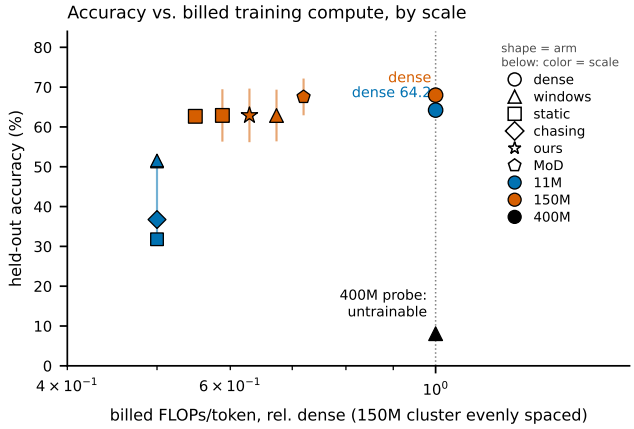

Figure 3: Accuracy vs. billed training compute (relative to the dense arm at the same scale, log axis; marker = arm, color = scale, values in Tables 1 and 2. The five gated 150M arms share a 0.57–0.59× budget, so their x positions are evenly spaced for readability rather than to scale). Gated 2-stage arms cluster at ∼0.58× dense compute and ∼5 points below it at 150M; MoD reaches dense compute savings with an inverted router. The 400M probe collapses (post-LN + Xavier init, untrainable at the preset per-size LRs tried), the honest edge of the scaling evidence; 1B was never run.

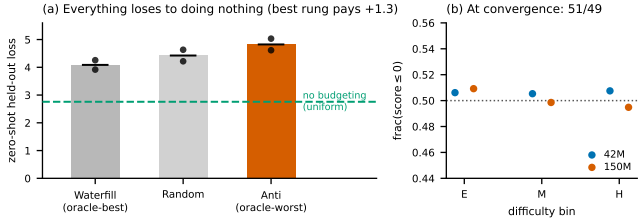

Figure 4: The allocation ladder, starved-150M (zero-shot held-out loss on frozen bodies; dots are seeds). **(a)** Every rung, including oracle-best waterfill, sits above the no-budgeting line: perfect allocation of a 0.5 budget pays +1.2–1.5 vs. full compute. **(b)** At convergence: a 51/49 coin flip (Table 7, App. C).

The compute-value concept is therefore parked as a *training-time* signal (Figure 4b): neither converged-body oracle allocation nor online chasing converts its mid-life structure into accuracy.

The same ordering holds at both trainable scales (Figure 3): dense first, gated 2-stage arms clustered, chasing and static behind; the architecture does not train at 400M, so no frontier claim is made beyond 150M (the 1.4B NL results are the scale evidence).

4.5 Natural language: the recipe transfers, with a boundary

The per-domain numbers (Table 6, Appendix C): Pythia-1.4b continue-trained on web text, Python code, and chain-of-thought math (350M packed tokens where available, 3000 steps, both arms share tokenizer, data, and the accounting of §2.5). Three outcomes, reported plainly. *Web text*: dense 2.532 vs. rotation-at-dense-deployment 2.729 (+7.79%; per-seed gaps 7.76/7.74/7.87%) at exactly half billed training FLOPs, inside the same pre-registered 2–10% band as the 150M testbed (+5.3%). At iso-billed budget (dense at 1500 steps) dense reaches 2.623 on its evaluation subsample, and 2.611 at iso-wall-clock (2222 steps at the measured 1.35×), both below rotation's 2.729 (the subsample reads ∼0.07 above the full set; the ordering holds under either basis): at this scale and horizon, training dense for fewer steps dominates; rotation should win only in steep loss-curve regimes, which this fine-tune is not. *Code*: +11.47%, just outside the band; marginal, one seed. *Math*: +95.34%, the recipe

fails. The mechanism is legible: MetaMathQA held-out loss is dominated by memorizable template structure, and intermittent full compute is precisely a regularizer against it: the recipe buys robustness by selling memorization. Evaluated *with the ablation live* the rotation models are far worse (web text 3.378): the trained body wants full compute at inference, and gets it. The scoping is stated rather than hidden: the half-cost recipe targets capability-limited regimes on distributed-structure data, not redundancy-dominated corpora. *The taxonomy row transfers.* Forking the web-text Stage-1 body into supervised stage-2 arms (§2.3): gate-grad 2.7077 vs. shuffled 2.7085, a paired $\Delta = 0.0008$ in loss (the 1.0-point rule was registered in accuracy at 11M; the delta is 0.4% of the dense-vs-rotation gap), with flat gate profiles in both arms. On language, measured value orders nothing and buys nothing over permuted targets: the content-irrelevance result is not a testbed artifact. Ablations (App. B): full-unfreeze stage-2 recovers 58% of the remaining gap (bodies adapt to their gates); the per-layer-target variant did not complete and is reported as not run.

4.6 Where does the compute go? Oracle gap and layer profile

Figure 4 prices allocation itself: ordering is real zero-shot, but not budgeting wins outright (uniform: 2.75–2.77; every allocation rung: 3.92–5.03), and chasing converts none of it (§4.3). Per-layer profiles are heterogeneous: one of twelve layers (0.60) concentrates the ordering.

4.7 Wall-clock realization

Billed FLOPs are an accounting device; the physical anchor is a measured realization (full details in Appendix F). Timing full Pythia-1.4b training steps on an A100, the masked path our driver uses realizes *nothing* (1.00×: zeroing saves nothing), while a gather-skip path that removes masked tokens' FFN rows reaches 1.35× dense at budget 0.5. The billed 2× arrives as a measured 1.35× because FFNs are only a fraction of step time; the gather is the whole result.

5 Discussion

Implications; which chasers does "not trainable" rule out? The recipe (train dense, measure, impose) needs no endogenous-signal search and plugs into serving stacks that already produce per-token difficulty estimates (Leviathan et al., 2023; Chen et al., 2023). Its target regime is training-cost-bound pipelines (fine-tuning, RLHF, continuous learning), where inference is already heavily optimized and the saving composes with those stacks because deployment is dense. The causal arms test the natural first-order chaser: measure $\partial\mathcal{L}/\partial g$, allocate hard against it, refresh every 100 steps. Stronger search over the same objective (bandits, RL, second-order corrections) faces the same structural problem: the scores are a property of the current body, and the ordering value decays as the body absorbs compute: the chaser destroys the signal it chases. The frozen-body oracle (Fig. 4) caps what *any* ordering could buy, and the cap is below not budgeting at all in every regime measured. We read "not trainable" as a property of gradient-defined value fields, not an impossibility proof for all conceivable allocators. The pathology is a first-order property of backpropagation consuming its own reducible loss; we predict it persists at pretraining scale.

When static schedules suffice. A constant g=0.5 from scratch collapses at every scale tested (11M: 31.8%; starved-150M: 22.4%), but at deployment the static choice is trivially optimal: the converged field is flat, so full compute everywhere is right. Static schedules suffice only when the loss does not benefit from intermittent compute (MetaMathQA is the counter-case).

Limitations. (i) The starved-150M causal replication is one seed per arm (Table 1; the 11M gap survives plausible noise, the starved gap does not; resolving runs queued in `DECISIONS.md`), and the 150M grid is n=2 (equivalence within $\sim\pm 9$ pts). (ii) The §4.1 taxonomy is ported from pre-repo artifacts, and the near-frozen protocol is the regime boundary (from-scratch co-adaptation untested). (iii) the 1.4B headline is three seeds; code and math single-seed.

AI use statement

An LLM assistant was used under the authors' direction for four purposes: finding prior art, running terminal commands and automation, copyediting, and code generation. All experimental design, decisions, kill-rules, and claims are the authors'.

Reproducibility statement

All code, manifests, and per-run JSONs are in the anonymized repository (`https://anonymous.4open.science/r/chemical-transformer-BB35`). Every table and figure generated in this repository regenerates from committed artifacts (`paper/generate_tables_iclr.py`, `scripts/*figure*`); pre-registered decision rules are committed with dates in `DECISIONS.md`; the workshop-era failure-taxonomy numbers of §4.1 are ported and labeled as such. Total rented-GPU spend is ∼$40, itemized in Appendix G (testbed runs on an Apple M3 at no cost; roughly half the ledger is re-runs after host failures and fixable bugs, which is what the manifests and kill-rules are for).

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *International Conference on Machine Learning*, 2009.

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

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, 2023.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your ViT but faster. In *International Conference on Learning Representations*, 2023.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling, 2023.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does bert look at? an analysis of bert's attention, 2019.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*, 2022.

Mostafa Dehghani, Jacob Devlin, Matt Merrill, Sam Cao, Dan Hendrycks, Colin Deutsch, Benjamin Vezer, Yasemin Soylu, Gintare Weller, Dominic Kumbong, et al. Universal transformers. In *International Conference on Learning Representations*, 2019.

Maha Elbayad, Jiatao Gu, Edouard Grave, and Michael Auli. Depth-adaptive transformer, 2020.

Angela Fan, Edouard Grave, and Armand Joulin. Reducing transformer depth on demand with structured dropout. In *International Conference on Learning Representations*, 2020.

Angela Fan, Benjamin Lavauzelle, Gabriel Synnaeve, Nicolas Usunier, and Sainbayar Li. NLP from scratch without large-scale pretraining. In *International Conference on Machine Learning*, 2021.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity, 2022.

Alex Graves. Adaptive computation time for recurrent neural networks, 2016.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks, 2017.

Linxi Hou, Zhejian Zhang, et al. Token dropping for efficient BERT pretraining. In *Association for Computational Linguistics*, 2022. arXiv:2203.13240.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q. Weinberger. Deep networks with stochastic depth. In *European Conference on Computer Vision*, 2016.

Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, Marc Brockschmidt, and Marc Gaunt. Codesearchnet challenge: Evaluating the state of semantic code search, 2019.

Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. In *Neural Computation*, volume 3, pp. 79–87, 1991.

Yann LeCun, John S. Denker, and Sara A. Solla. Optimal brain damage. In *Advances in Neural Information Processing Systems*, 1990.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*, 2021.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning*, 2023.

Zhenghao Lin, Zhibin Gou, Yutao Hu, et al. Rho-1: Not all tokens are what you need. In *Advances in Neural Information Processing Systems*, 2024. arXiv:2404.07965.

Pavlo Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Importance estimation for neural network pruning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Colin Raffel, Leandro Von Werra, and Thomas Wolf. The fineweb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems (Datasets and Benchmarks)*, 2024.

Yongming Rao, Jiwen Zhao, Zheng Zhu, Jie Lu, and Jie Zhou. DynamicViT: Efficient vision transformers with dynamic token sparsification. In *Advances in Neural Information Processing Systems*, 2021.

David Raposo, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam Santoro. Mixture-of-depths: Dynamically allocating compute in transformer-based language models, 2024.

Tal Schuster, Adam Fisch, Jaakkola Gupta, Mostafa Dehghani, Dara Bahri, Vinh Tran, Yi Tay, and Donald Metzler. Confident adaptive language modeling, 2022.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, 2017.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2024.

Surat Teerapittayanon, Bradley McDanel, and H. T. Kung. Branchynet: Fast inference via early exiting from deep neural networks, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E. Gonzalez. Skipnet: Learning dynamic routing in convolutional networks, 2018.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. Deebert: Dynamic early exiting for accelerating bert inference, 2020.

Jiahui Yu and Thomas S. Huang. Universally slimmable networks and improved training techniques, 2019.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. Metamath: Bootstrap your own mathematical questions for large language models. In *International Conference on Learning Representations*, 2024.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, 2023.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, William Fedus, Noam Shazeer, and Google Brain. St-moe: Designing stable and transferable sparse expert models, 2022.

A Steps-matched comparisons

Table 3: 150M arms at equal optimizer steps. The two-stage pipeline's own stage-1 boundary is 5000 steps, the same count as the single-stage arms, so the comparison isolates what stage 2 adds over the pipeline's own base at equal steps, and what it costs relative to dense.

150M arm	Acc	MFLOPs/token
Dense 5000 steps	68.0±7.0	467±0
Stage-1 body @5000 (pipeline's own base)	62.5±8.8	467±0
Ours (tag), full pipeline @7000	62.9±9.1	276±1
Shuffled, full pipeline @7000	62.9±8.8	271±3

Table 3: at 5000 steps the two-stage pipeline's base sits at 62.5 (467 MFLOPs/token, the same billed compute as dense); the full pipeline reaches 62.9 at 276 MFLOPs/token. Read together: stage 2 at 150M adds +0.4 points over its own body while cutting 41% of billed compute; the value of the pipeline at the healthy recipe is the compute reduction, not accuracy (dense wins both at equal steps).

B Ablations

Full-unfreeze Stage-2. The default protocol near-freezes the body ($10^{-2}\times$ the head learning rate). Unfreezing it fully on the web-text fork (seed 1, 2000 steps): held loss 2.6054 vs. the near-frozen 2.7077 and dense 2.5322: the body adapting to its gates recovers 58% of the remaining gap, consistent with the main-text reading that trained methods can exceed frozen-body allocation by adaptation, not by better ordering (gate profiles stay flat: $[0.33, 0.34, 0.35, 0.35]$).

Per-layer target freedom. Implemented end-to-end (per-layer oracle scores, per-layer quartile bins, per-layer gate targets); the single run hung on our 40GB budget and was abandoned under the results freeze; reported as not run, left to future work. The summed-score protocol of the main text is unaffected.

Shuffled control at all scales. 11M: Table 4; 150M: Table 2; 1.4B web text: §4.5: the paired gate-grad − shuffled difference is 0.0008 at language scale, replicating the testbed result.

C Per-seed numeric tables

Table 4: P0: paired Stage-2 arms from the same Stage-1 body per seed (3 seeds, mean±std). Billed training FLOPs are identical across the bottom two arms by construction (the oracle pass is billed to both); the tag arm has no oracle pass. The constant-gate arm of Figure 2a runs the same two-stage protocol with a fixed g=0.5 target: 72.3 ± 2.1 accuracy at the same 19.0 MFLOPs/token.

Stage-2 arm	Acc	Balanced	MFLOPs/tok	Gate spread
Tag targets (labels)	74.8±1.3	83.6±0.8	19.4±0.1	0.21±0.06
Gate-grad (measured value)	74.5±1.0	83.4±0.7	19.2±0.0	0.08±0.01
Shuffled control	74.7±1.1	83.5±0.7	19.3±0.0	0.06±0.01

Table 5: 150M probe grid per-seed accuracies (the main-text Table 2 reports mean±std over these; MFLOPs/token are seed-invariant). With n=2, the seed pairing, not the std, is the honest read: dense wins both seeds over every 2-stage arm; MoD splits; the 2-stage arms are within noise of each other in both seeds.

Arm	seed 0	seed 1	MFLOPs/tok
Dense	63.0	72.9	467
Fixed g=0.5 (scratch)	63.6	61.8	265
Fixed (2-stage)	56.6	69.2	273
Ours (tag targets)	56.4	69.4	276
Shuffled (2-stage)	56.6	69.1	269
MoD (single-stage)	71.9	63.2	269

Table 6: NL transfer: dense vs. token-level FFN dropout (Pythia-1.4b, 3000 steps, matched data; rotation evaluated at dense deployment). Loss and gap cells list per-seed values (web text: 3 seeds; code, math: 1). Web text sits inside the pre-registered band; code is marginal; on template-dominated math the recipe fails outright: its noise-regularization opposes the memorization that low loss there is made of.

Domain	n	dense loss	rot.@dense	gap (%)	FLOPs (%)
Web text (FineWeb-Edu)	3	2.532/2.532/2.532	2.729/2.728/2.732	7.76/7.74/7.87	50
Code (CodeSearchNet-py)	1	1.007	1.123	11.47	50
Math (MetaMathQA)	1	0.382	0.747	95.34	50

D Experimental details

Difficulty is a proxy throughout: operand-size-defined at testbed scale, reference-model-inherited for the NL bins.

Arithmetic. Vocabulary of 22 digit/operator tokens; frozen held-out sets of 256 sequences per split, identical across arms and seeds; accuracy on answer tokens only. 11M: d=384, 6 layers, 6 heads, d_{ff}=1536, sequence length 256, batch 32, LR 3×10^{-4} (the raw recipe trains stably at this scale). 150M: d=1024, 12 layers, RoPE, SDPA attention, sequence length

Table 7: Per-bin measured value signature by model size (converged bodies, validated recipe). frac(score≤0) ≈ 0.5 and mean scores at 10^{-6}: the converged field is coin-flip at every bin and both scales (pre-registered readout executed; prediction not confirmed); plotted as Figure 4(b).

Size	frac(score≤0) per bin (E/M/H)	mean score per bin
42M	0.51/0.51/0.51	+7.2e-07/+2.7e-06/-3.0e-06
150M	0.51/0.50/0.49	-2.5e-06/-5.8e-07/+1.6e-06

512, batch 32, validated recipe: LR 10^{-4}, cosine decay to 0.1×, 2% warmup, weight decay 0.1. The 11M causal test runs 3000 steps; the 150M grid 5000; the starved regime 2500; the 150M convergence pair 10000.

Natural language. Pythia-1.4b continue-trained on FineWeb-Edu (350M tokens packed to 2049-token rows, 167,398 train / 3,416 held-out sequences), AdamW 10^{-5}, cosine to 0.1× with 2% warmup, bf16 autocast over fp32 master weights, effective batch 8 sequences via 4-sequence micro-batches × gradient accumulation 2, 3000 steps, evaluations every 250 steps on a fixed 32-sequence subsample with a full held-out evaluation at the final step. Code domain: CodeSearchNet python (function docstring + body); math domain: MetaMathQA (question + response). The per-layer oracle stores per-(token, layer) scores in fp32 (negative partial losses) and quartile-bins each layer's positive scores.

Billing conventions, stated precisely. At 1.4B the counter bills FFN at 134.22 MFLOPs/token × 24 layers = 3221 (dense total 3246; $\bar{g}$=0.51 bills 1636, 0.50×) and adds the attention term once rather than per layer (25 MFLOPs/token, 0.8%), so the accounting is effectively FFN-proportional; with per-layer attention billing (≈19% of the FFN term at this config), $\bar{g}$=0.5 would correspond to ≈0.58× total. The 150M testbed counter bills attention per layer (≈14% non-FFN share; $\bar{g}$=0.51 → 0.58×). All headline claims are stated in *billed FFN* terms; the total-training-FLOPs saving at 1.4B is ≈40–45%, and the measured 1.35× wall-clock (§4.7) is the physical anchor.

Wall-clock. Measured on an A100: full forward+backward steps at batch 4×2048, 3 warmup + 10 timed iterations, synchronized; the gather-skip path index-selects active token rows before each FFN and index-copies results back.

E Budget redistribution

Table 8: Supervision targets vs. realized gates (budget redistribution). The floor + quartile targets have mean ≈ 0.31 and nominal spread 0.7; the pinned budget ($\bar{g} \to 0.5$) lifts and compresses them. The compression is the budget term redistributing, not the head failing to allocate: a pinned mean is what makes the inference budget a settable dial.

Arm	mean target	nominal spread	realized $\bar{g}$	realized spread
Tag targets	0.51 (fixed)	0.60	0.51	0.21±0.06
Gate-grad	∼0.31	0.70	0.51	0.08±0.01
Shuffled	0.30	0.70	0.51	0.06±0.01

F Wall-clock realization details

We time full training steps (forward + backward, 4×2048 tokens) of Pythia-1.4b under three execution strategies: dense; the *masked* path our training driver uses (compute every FFN row, multiply masked tokens by zero); and a *gather-skip* path that removes masked tokens' FFN rows (index-select on the flattened token axis). On an A100: dense runs at 2029 tok/s; masked-at-half runs at 1.00×, while gather-skip reaches 1.35× dense at budget 0.5, 1.13× at 0.75, and 1.67× at 0.25. The saving is real but sub-linear: attention, optimizer, and

memory movement do not shrink with the FFN budget, and the naive masked path that most token-dropping code ships realizes nothing.

G Compute ledger

Table 9: Rented-GPU ledger, Sep 12–23, 2026. Single numbers are exact where receipts exist; ranges are estimates from instance hours at listed rates.

Stage	Hardware	Spend ($)
150M probe grid, 16 runs (Sep 12–14)	2× RTX-class, ∼20 h	11–13
1.4B NL pair, web text (Sep 16)	RTX 6000 Ada, 8 h	4.40
Code + math pairs, 4 arms (Sep 17)	RTX 4090D, 5.3 h	4.20
Wall-clock bench + web-text seed-1 chain + wave-2 (Sep 17–18)	A100/6000 Ada, ∼13 h	12.2
Seed-2 attempts (host failures)	mixed	∼2.5
Seed-2 web-text pair, successful re-run (Sep 23)	RTX 6000 Ada, ∼4.9 h	∼3.4
Dud boots, queue stalls, orchestration failures	mixed	∼2.2
Total ∼$40 (was $36.45 before the seed-2 re-run)		

11M testbed runs (all failure-taxonomy, causal, and life-cycle results): Apple M3 via MPS, no cost. Two results were initially lost to the incidents above (one rotation arm to a training bug, later re-run; the seed-2 pair to host failures, re-run successfully on Sep 23 for the 3-seed headline) and are reported as such.