How to Loop MoE: Flatten the Experts, Untie the Attention

Anonymous authors
Paper under double-blind review

Abstract

Looped Transformers reuse one block of layers several times: by spending extra computation they push a model of fixed size further, and so use its parameters more fully; while sparse mixture-of-experts (MoE) models activate only a few of many experts for each token. Looped MoE bridges these two design philosophies and gives MoE models new potential for better expert usage, but it raises a question: how to loop a MoE? We answer it with Foil. With the expert parameters and the expert compute per token held fixed, Foil (1) *flattens* the experts, halving the expert layers, doubling the experts per layer and doubling the passes, so that every routing decision chooses from a larger pool, and (2) *unties* the attention, giving each pass its own attention parameters while the experts and routers stay shared. Experiments show that Foil clearly outperforms the unflattened looped baseline: at 20B tokens every Foil model has lower pretraining loss than the baseline; at 100B tokens the loss improves monotonically with the degree of flattening, the most flattened Foil ending 0.012 nat below the baseline at equal parameters and compute, with downstream accuracy on par or better; untying the attention also yields more balanced and more confident routing at equal shape. Our ablations analyse why Foil works and turn the findings into design guidance for looped MoE: the returns of looping and of widening the expert layers amplify each other, routing confidence tracks healthy expert use better than load balance, and a sparse looped MoE should therefore use more experts per layer and more passes. Code and configurations are available at https://anonymous.4open.science/r/how-to-loop-moe-iclr27-submission-014B.

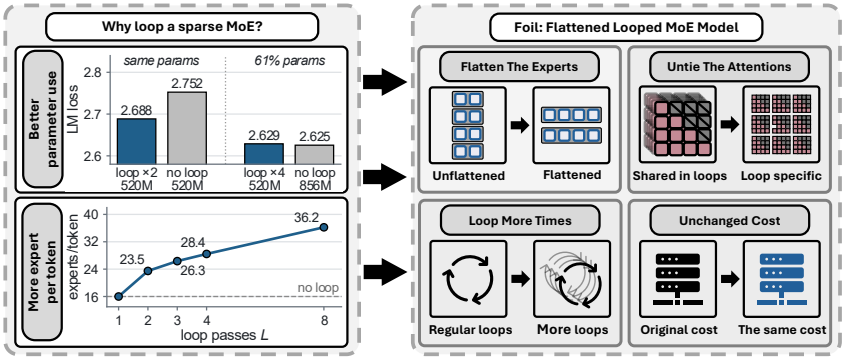

Figure 1: Motivation of designing Foil. *Left: why loop a sparse MoE*. (1) At equal parameters, looping lowers the loss, and matches non-looped models with much less parameters; (2) Tokens reaches more distinct experts the more passes it makes. *Right: Foil's design*. (1) Flattens the experts, (2) Unties the attention. (3)Loops more times, (4)Keep parameters and compute unchanged.

1 Introduction

Looped models apply one block of Transformer layers repeatedly to an evolving hidden state, so that the depth of computation is set by the number of passes rather than by the number of stored

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.

layers (Dehghani et al., 2019). A model of fixed size can thus be made stronger by computing more, which uses its parameters more fully: both theory and experiments show that looped Transformers suit computations that need many iterative steps (Giannou et al., 2023; Saunshi et al., 2025), and looping has recently been scaled to large pretraining runs and to spending more computation at inference time (Geiping et al., 2025; Zhu et al., 2025). As hardware compute grows far faster than memory capacity and bandwidth (Gholami et al., 2024), trading computation for stored parameters is increasingly attractive, and how best to loop a model has become an active question (Prairie et al., 2026; Huang et al., 2026).

Sparse mixture-of-experts (MoE) models take a different route to efficiency: each layer holds many experts, but every token is routed to only a few of them, so that the parameter count far exceeds the computation per token; current large models hold hundreds of experts per layer and route each token to only a few (DeepSeek-AI, 2024; Kimi Team, 2025). The price is that a token sees only k experts in each layer, and the more experts there are, the less often each of them is used. Whether the experts are actually used, and whether they specialise, has been a central question since the first sparse MoE models (Fedus et al., 2022; Zoph et al., 2022), and a balanced load alone does not answer it (Li et al., 2026).

These two design philosophies are, however, naturally compatible. A routing decision exposes a token to only a few of the available experts; looping changes this: every pass gives the token a new routing decision in the same layer, so it can reach different experts, and different combinations of them, without storing any additional expert parameters. Our measurements make this potential concrete in three observations. *(1) Looping brings better performance or fewer parameters.* At equal parameters, looping twice lowers the loss by 0.064 nat, and a looped model with far fewer parameters nearly matches a non-looped model with twice the layers by spending more computation per token (Figure 1, top). *(2) Looping lets a sparse MoE use more of its experts.* The number of distinct experts a token reaches grows with the passes, from 16 without looping to 36 with eight passes (Figure 1, bottom). *(3) Looping unlocks equivalent-parameter properties for MoE.* Because experts are called repeatedly, the equivalent number of experts and the number of possible routing combinations grow multiplicatively as the looped block is flattened, while the real experts and the compute stay fixed.

Earlier work has combined looping with experts (Csordás et al., 2024; Chen et al., 2026b; Jaggi, 2026; Li et al., 2025), but none has studied how the experts should be distributed over the looped block—how many experts a layer holds, how many layers the block has and how many times it is looped—when the expert parameters and the compute are fixed. Nor has it been asked which parameters a pass should own: with attention shared across passes, flattening discards the attention parameters of the layers it removes, whereas giving each pass its own attention keeps the parameter budget and lets successive passes process the shared experts' inputs differently, at no extra compute. This raises our question: **how to loop a MoE when its parameters and compute are fixed?** We answer it with Foil. Foil (1) *flattens* the experts, scaling down the number of layers in the looped block while scaling up the experts per layer and the number of passes in proportion, so that every routing decision chooses from a larger pool, and (2) *unties* the attention, giving each pass its own attention parameters while the experts and routers stay shared across passes. Our contributions are:

- We propose **Foil**, which flattens the experts and unties the attention of a looped MoE while holding the parameters and the compute per token fixed (Section 2).
- We validate Foil in 20B-token pretraining and 100B-token continued training: it clearly lowers the pretraining loss of the unflattened baseline, matches or exceeds it downstream, and untying the attention yields healthier routing than tying it at the same shape (Section 3).
- Through systematic ablations we characterise several phenomena of looped MoE and distil a design suggestion: a sparse looped MoE should use appropriately more experts per layer and more passes (Section 4).

2 Methodology

2.1 Notation and accounting

We first fix the notation for the looped block and count what it stores and computes; Figure 2 shows the vanilla looped MoE and Foil side by side, and Section 2.2 defines Foil.

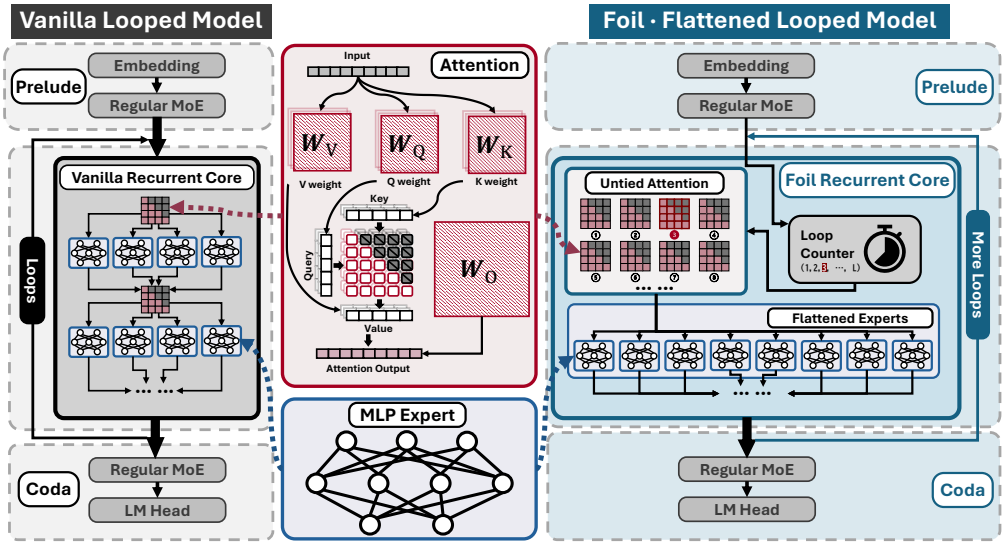

Figure 2: Vanilla looped MoE versus *Foil*. Both models share the same skeleton: a prelude (embedding and one regular MoE layer), a recurrent core applied for several passes, and a coda (one regular MoE layer and the LM head). The vanilla core ties both attention and experts across passes. Foil flattens the core—it keeps the total number of experts fixed while using fewer layers, more experts per layer and more passes—and unties the attention: the experts are shared across passes, but every pass has its own attention set, selected by the pass index. Both models call the same number of experts per token. When the core has several layers, every layer carries one attention set per pass; the ellipsis marks the remaining sets. Red: attention; blue: experts.

Resource and routing accounting. Consider a block with D separate banks of E experts, each containing p_{exp} parameters, traversed L times. Assume unrestricted top-k routing with $1 \le k \le E$, exactly k distinct experts executed per visit, and no dropped assignments. Then $E_{\text{real}} = E \times D$, $D_{\text{eff}} = D \times L$, $E_{\text{eq}} = E \times D \times L$, $E_{\text{comp}} = k \times D \times L$ and $P_{\text{experts}} = E \times D \times p_{\text{exp}}$ (router scoring, below 1% of the expert compute for the most flattened shape, is not counted; Appendix A).

Appendix A gives a rigorous formulation of the looped block, of these identities and of the bounds on expert coverage.

2.2 Foil: flatten the experts, untie the attention

We build on the looped skeleton of Huginn (Geiping et al., 2025): token embedding, a *prelude* of one ordinary MoE layer, a looped block of D layers applied L times, a *coda* of one ordinary MoE layer, and the output layer. The prelude and coda have 8 experts each, run once and are never flattened, so all configurations differ only in the looped block, whose shape we write as (E, D, L) (Figure 2).

Flattening enlarges the routing pool at fixed expert budget and compute. One flattening step maps (E, D, L) to $(2E, D/2, 2L)$: half the layers, twice the experts per layer, twice the passes. It keeps the expert parameters ($E_{\text{real}} = E \times D$), the expert calls per token ($E_{\text{comp}} = k \times D \times L$) and the effective depth ($D_{\text{eff}} = D \times L$) fixed, and enlarges the pool E of every routing decision and the equivalent expert count $E_{\text{eq}} = E \times D \times L$, the number of experts in the non-looped model obtained by unrolling the passes. Three steps from $(8, 8, 2)$ give $(16, 4, 4)$, $(32, 2, 8)$ and $(64, 1, 16)$, raising E_{eq} from 128 to 1024.

Untying the attention restores discarded parameters at no extra compute. A conventional looped model shares the whole block across passes, attention included, so flattening also removes attention sets: the looped block of $(8, 8, 2)$ holds eight, that of $(64, 1, 16)$ a single one reused on every pass, while the experts are untouched. We therefore share the experts and routers across

passes but give every pass its own attention, $DL = 16$ sets along the whole sequence, with no pass embedding (Figure 2, right). Untying adds no computation; it only restores the attention parameters that sharing discards. We call the flattened models with shared attention *proto-Foil* and those with untied attention *Foil*.

We propose three Foils: Foil-1 $(64, 1, 16)$, Foil-2 $(32, 2, 8)$ and Foil-3 $(16, 4, 4)$. The baseline, Base $(8, 8, 2)$, is the looped model with untied attention and differs from the Foils only in shape; the controls, Base-tied and proto-Foil-3, -2, -1, share the attention. Base and the Foils have 553.7M parameters each, the shared-attention models 490.8–520.2M. The further models of the ablations (Section 4) are named by their shape. All models route each token to the $k = 2$ most probable experts of a layer under a linear router with softmax and renormalised weights, with SwiGLU experts (all settings and parameter counts in Table 3).

2.3 Three metrics describe how the experts are used

We measure expert use on the recurrent core over 255,500 probe tokens; $S_{t,d}^{(\ell)}$ are the k experts token t selects at layer d and pass ℓ, and $p_{t,(1)} \geq \cdots \geq p_{t,(E)}$ its sorted router probabilities.

Distinct experts reached per token (U_t).

$$U_t = \sum_{d=1}^{D} \Big| \bigcup_{\ell=1}^{L} S_{t,d}^{(\ell)} \Big| \;\leq\; D \min(E, kL)$$

counts the distinct experts token t reaches over its passes; its mean, as a fraction of the bound, shows how many experts looping lets a token use.

Load balance (B_2).

$$B_2 = \frac{1}{E \sum_{i=1}^{E} q_i^2} = \frac{N_2}{E},$$

where q_i is the share of routing requests expert i of a layer receives, passes pooled, and N_2 the effective number of experts (Wu et al., 2026, Eq. S10). This is Jain's fairness index (Jain et al., 1984): it lies in $[k/E, 1]$, equals 1 for even load and shows whether the load concentrates on a few experts; model values pool N_2 over layers.

Routing confidence: the median margin ratio (MMR).

$$\mathrm{MMR} = \frac{p_{t,(1)}}{p_{t,(E/2)}} = \exp\big(z_{t,(1)} - z_{t,(E/2)}\big)$$

compares the router's first choice with the median-ranked expert of the whole pool (z: router logits), a margin taken against the median expert rather than within or at the edge of the selected set. It equals 1 for an indifferent router at any E, is averaged geometrically over decisions, and targets balanced load with indifferent routing (Li et al., 2026).

The metrics are read together, relative to themselves, or at equal shape. Router scores are not rescaled, and both B_2 and MMR depend on E; only same-direction changes of both are read as healthier routing. A traffic-matched masking test checks whether rarely used experts are dispensable (Appendix I); related metrics are in Appendix B.

3 Experiments

Every model is trained on the same data in the same order, from the same initialisation seed and under the same learning-rate schedule, for 20B tokens (training details in Appendix C); this is 35–40 tokens per parameter for the eight compared models, enough by the Chinchilla ratio (Hoffmann et al., 2022), and we scale the main models up to 100B tokens. The router applies a softmax at temperature 1 over the experts of a layer and selects the top two, without noise or bias terms.

Downstream, the main text reports three zero-shot tasks, one representative for each of word prediction (LAMBADA, standard split), sentence continuation (HellaSwag) and coreference resolution

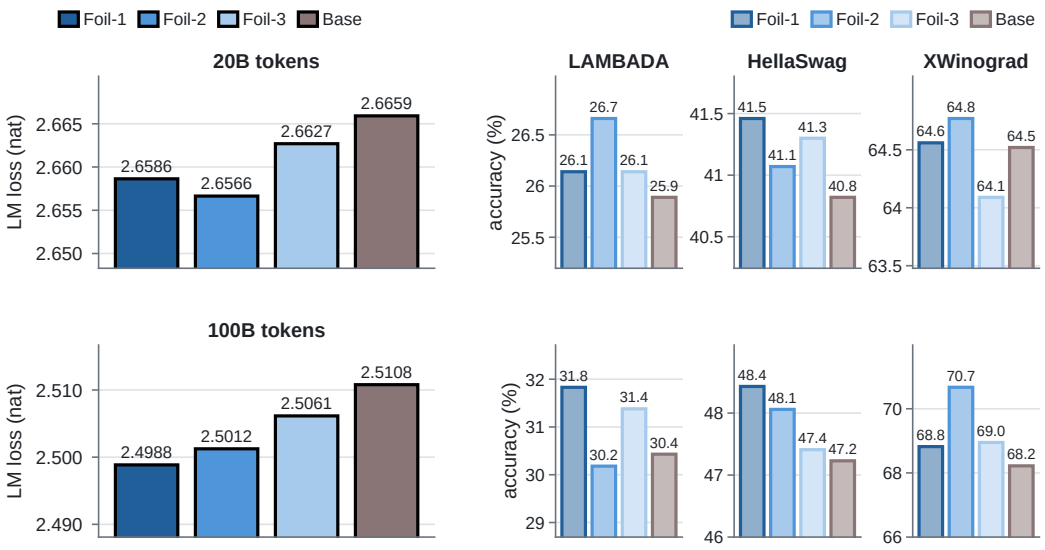

Figure 3: Flattening with untied attention: loss and downstream accuracy of the Foils and Base. Top row: 20B tokens; bottom row: after continued training to 100B tokens. In each row, solid bars on the left show the final loss (lower is better) and tinted bars on the right the zero-shot accuracy on the three representative tasks (higher is better), each panel with its own vertical axis; bars from left to right: Foil-1, Foil-2, Foil-3, Base. At 20B **all three Foils are below Base**; at 100B **the loss falls monotonically with flattening** and Foil-1 ends 0.012 **nat below Base**. Downstream accuracy is on par or slightly better; further tasks are in Appendix E.

(XWinograd, English). We report (1) language-modelling loss at the end of training, compared between models as paired differences with standard errors, (2) downstream task results, and (3) routing metrics (B_2 and MMR). Full results are provided in Appendix D, including a seed-change experiment that supports the robustness of the results.

*The runs of this study consumed approximately **12.6k NVIDIA H200 GPU-hours** (about 18.2k including earlier control and failed runs); each run used at most one node with eight H200 GPUs.*

3.1 Flattening with more passes raises the ceiling of model ability

Holding the real expert count $E_{\text{real}} = 64$ and the expert calls per token $E_{\text{comp}} = 32$ fixed, we flatten Base step by step, halving the layers of the recurrent core, doubling the experts per layer and doubling the passes, which gives Foil-3, Foil-2 and Foil-1; every pass keeps its own attention.

Flattening improves the model at both training lengths (Figure 3):

- **At 20B tokens**, all three Foils reach a lower loss than Base. The loss falls through the second flattening step and then levels off: Foil-1 ends 0.007 nat below Base, slightly above Foil-2. On the three representative tasks, the Foils are at or above Base, within their standard errors.
- **At 100B tokens**, every flattening step lowers the loss and Foil-1 is the strongest, 0.012 nat below Base. On the three representative tasks, all three Foils score slightly above Base, by one to two standard errors.

The 100B results confirm the 20B findings and enlarge them: the gain of the flattest shape grows with training, while Foil-2's stays put. Further downstream results are in Table 5. The remaining experiments use 20B tokens.

3.2 Untied attention unlocks the flattened model's potential

At each of the four shapes we compare a pair of models that differ only in whether the attention is tied across passes: Base-tied and proto-Foil-3/2/1 share one attention set over all passes, whereas

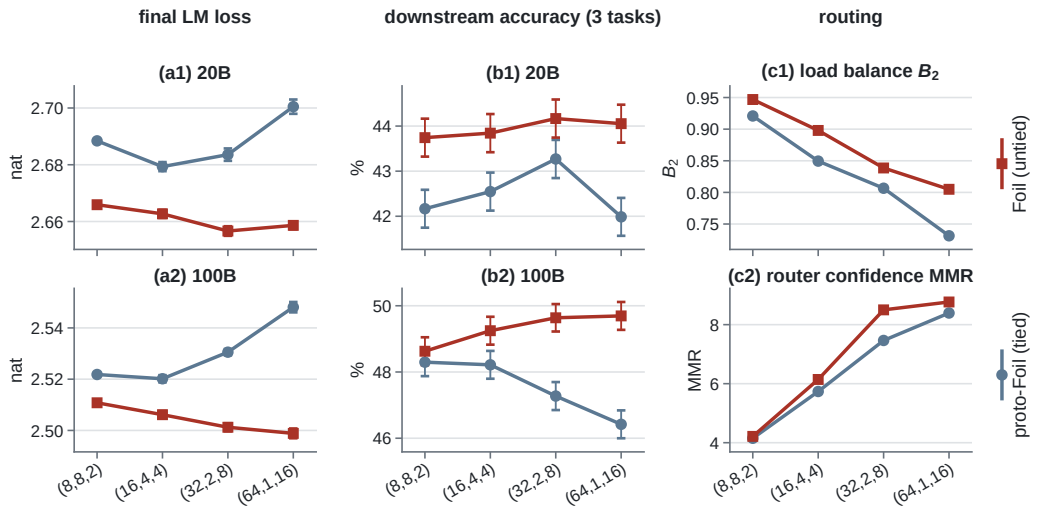

Figure 4: Tied versus untied attention at equal shape: loss (a), downstream accuracy (b) and routing (c). Horizontal axis: flattening steps from $(8, 8, 2)$ to $(64, 1, 16)$; each step to the right doubles E and L and halves D. Red squares: untied attention (Base and Foil-3, -2, -1); grey-blue circles: tied attention (Base-tied and proto-Foil-3, -2, -1). (a1, a2) Final loss at 20B and 100B tokens; error bars are twice the standard error of the paired difference from the $(8, 8, 2)$ model of the same line. (b1, b2) Mean zero-shot accuracy over the three representative tasks, ± 1 standard error. (c1, c2) Load balance B_2 and routing confidence MMR of the recurrent core at 20B tokens, from a single-seed probe without error bars; since both depend on E, only the two points at the same shape are compared. With tied attention the loss rises again after the first flattening step, while with untied attention it levels off at 20B and keeps falling at 100B; the gap is largest for the most flattened pair (0.049 nat at 100B). At every shape the untied model is both more balanced and more confident.

Base and Foil-3/2/1 keep one per pass (Table 3). The tied models follow the classic looped design and serve as controls, but flattening discards their attention parameters (520.2M to 490.8M), whereas Base and the Foils all have 553.7M at equal compute.

Untying lowers the loss at every shape, more so when flatter. At 20B tokens, the untied model is better in all four pairs, most of all when fully flattened: Foil-1 is 0.042 nat below proto-Foil-1 (Figure 4, a1–a2). After continued training to 100B tokens, the gap grows steadily with flattening, to 0.049 nat for the Foil-1 pair.

Downstream, the untied model scores higher in every pair. On the mean of the three representative tasks, the untied model is ahead in all four pairs at both token budgets; at 20B the difference exceeds two standard errors in every pair except the Foil-2 pair, and at 100B it widens with flattening and exceeds two standard errors in the two most flattened pairs, reaching 3.3 points for Foil-1 over proto-Foil-1 (Figure 4, b1–b2). Further downstream results are in Table 5.

At equal shape, untied attention routes more evenly and more confidently. Reading the two routing metrics together and only at equal shape (Section 2.3), the untied model has both a higher load balance B_2 and a higher routing confidence MMR than its tied counterpart in all four pairs, with the smallest gap at the Base pair (Figure 4, c1–c2). The two metrics move in the same direction, which we read as healthier routing.

4 Ablation Studies

We run the ablations at 20B tokens, where Section 3 showed that the trends agree with those at 100B tokens and that changing the initialisation seed alone barely moves the loss. Unless stated

Table 1: Final loss (nat) of the tied-attention models at 20B tokens and its decrease from the model with half the passes or half the experts per layer (standard errors 0.0006–0.0012); –: not applicable.

D	E	L	E_{real}	k/E	Final loss (nat)	Δ loss vs. model with half the passes	Δ loss vs. model with half the experts per layer
4	8	2	32	1/4	2.7874	–	–
		4	32	1/4	2.7196	0.068	–
		8	32	1/4	2.6875	0.032	–
	16	2	64	1/8	2.7490	–	0.038
		4	64	1/8	2.6793	0.070	0.040
		8	64	1/8	2.6384	0.041	0.049
8	4	2	32	1/2	2.7277	–	–
		4	32	1/2	2.6747	0.053	–
	8	2	64	1/4	2.6884	–	0.039
		4	64	1/4	2.6287	0.060	0.046
		8	64	1/4	2.5961	0.033	–
	16	2	128	1/8	2.6414	–	0.047

otherwise, the models in this section tie the attention across loop passes, so that adding passes adds no parameters (untying it keeps the gains of flattening, Section 3.2), and are named by their shape (E, D, L). A *gain* is a decrease of the final loss (nat), paired as in Section 3.

4.1 Wider, sparser layers slow the diminishing returns of looping

We add passes, from two to four and from four to eight, to four layer configurations (E, D) and measure the gain of each step at fixed (E, D): $(8, 4)$ and $(4, 8)$ with $E_{\text{real}} = 32$, and $(16, 4)$ and $(8, 8)$ with $E_{\text{real}} = 64$.

Wider, sparser layers gain more from additional passes. From four to eight passes (Table 1, passes column), $(16, 4)$ gains 0.041 nat, against at most 0.033 for $(8, 4)$, which has half its real experts, and $(8, 8)$, which has twice its layers. From two to four passes (Table 1), $(16, 4)$ gains more than $(8, 8)$ and is on par with $(8, 4)$. At $E_{\text{real}} = 32$, the sparse $(8, 4)$ gains clearly more from two to four passes than the half-active $(4, 8)$, although $(4, 8)$ spends twice the expert compute per pass. Since each comparison changes more than one quantity, we conclude only that wider, sparser layers make the returns of looping decline more slowly, without attributing this to a single cause.

4.2 More passes enlarge the gain from widening

Here *widening* doubles the experts per layer E at fixed D and L, so the expert calls per token E_{comp} stay the same while the real experts E_{real} double.

Widening and looping amplify each other. Widening $(8, 4, L)$ to $(16, 4, L)$ gains more the more passes the model makes, from 0.038 nat at $L = 2$ to 0.049 at $L = 8$, and widening $(4, 8, L)$ to $(8, 8, L)$ likewise gains more at $L = 4$ than at $L = 2$: the more passes, the more widening pays. For each 2×2 block we take the gain of doing both minus the gains of widening alone and of looping alone; of the three such interactions, two are clearly positive and one is on par with zero, and none is negative (Table 1, experts column, as its increase with L).

Widening shows diminishing returns in the number of experts added. Widening once more, from $(8, 8, 2)$ to $(16, 8, 2)$, gains only 1.2 times as much as widening from $(4, 8, 2)$ to $(8, 8, 2)$, although it adds eight experts per layer instead of four (Table 1, experts column); at two passes the added width is used less, in line with the finding above that widening pays more with more passes. Since our design cannot widen a layer at fixed E_{real}, part of this decline may come from the change in E_{real}.

Takeaway 1

Widening the expert layers and looping more are complementary: each enlarges the other's gain. Under a fixed expert-parameter and per-token compute budget, this is exactly what flattening does: wider, sparser layers looped more often.

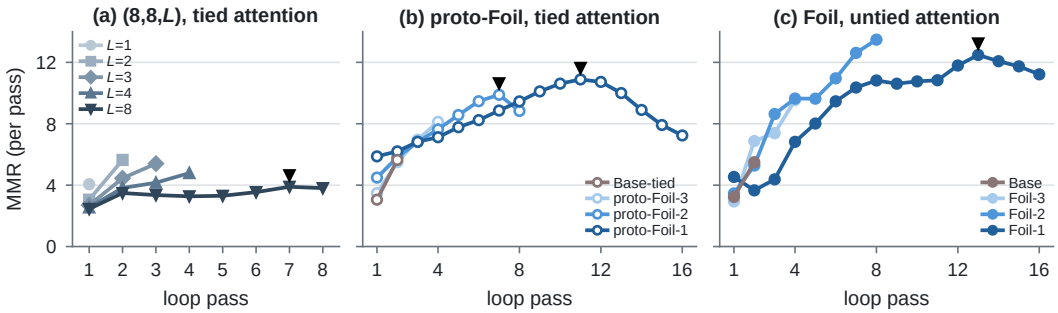

Figure 5: Routing confidence (MMR) per loop pass at 20B tokens (geometric mean over the core layers). (a) At fixed $(E, D) = (8, 8)$, more passes lower the confidence of every pass, and beyond $L = 2$ the model-level MMR falls from 4.15 to 3.36; $(8, 8, 8)$ peaks at pass 7. (b, c) Along the flattening sequence, confidence generally rises over the passes; proto-Foil-2, proto-Foil-1 and Foil-1 reach a peak (black triangles) and then fall, whereas Foil-2, the Foil with the lowest loss at 20B, has no peak. After a peak, further flattening or more passes very likely gain least (Section 4.4).

4.3 Load balance alone does not indicate healthy specialisation

Load balance is the classic measure of expert utilisation, but it has been questioned: a balanced load can hide a router that has no preference among the experts, which motivated measuring routing confidence as well (Li et al., 2026).

With untied attention, flattening routes more confidently, less evenly, and reaches a lower loss. Along the flattening sequence with untied attention, B_2 falls steadily from the unflattened model (Base) to the most flattened one (Foil-1), while MMR rises (Figure 4, c1–c2). Following the readings of Section 2.3, we read these only as trends of each metric, not as absolute comparisons across different E. Yet the most flattened model reaches a lower loss than the unflattened one (Figure 3): the lower balance does not come with a worse model.

The least-used experts are not the least useful. Masking the experts that carry the least 10% of the routing traffic and comparing with traffic-matched random groups (Appendix I), the least-used group raises the loss more than the random mean in seven of the eight models compared in Section 3 (Base-tied, proto-Foil-3/2/1, Base, Foil-3/2/1), and in none does it fall below the range of the random groups; the exception, proto-Foil-2, is on par with the random median (Table 13 and Figure 6). For sparse MoE, load balance alone is therefore not a suitable indicator of healthy expert specialisation.

4.4 Routing confidence and its per-pass peak index the looping gain

In Section 3.2, the MMR difference between Foil and proto-Foil is largest for the Foil-2 pair (14%; Figure 4, c2), and Foil-2 has the lowest 20B loss of the Foil family including Base (Figure 3). This led us to examine MMR pass by pass.

Routing confidence and the gain per pass decline together. In the $(8, 8, L)$ series, where every model has eight experts per layer and absolute values are therefore comparable, the model-level MMR falls from 4.15 to 3.36 and the average gain per pass over the non-looped $(8, 8, 1)$ from 0.064 to 0.022 nat as L goes from 2 to 8; the two decline together at every step, and $(8, 8, 8)$ is lowest in both (Table 9). We state only that the two move in the same direction, not that one causes the other.

After a peak, further flattening or more passes very likely bring the smallest or no gain. In many models MMR rises with the pass index up to some pass and falls afterwards (Figure 5): $(8, 8, 8)$, proto-Foil-2, proto-Foil-1 and Foil-1 peak, whereas the Base and Base-tied pair, the Foil-3 and proto-Foil-3 pair, Foil-2 and the $(8, 8, L)$ models with $L \leq 4$ are most confident on their last pass. After its peak, flattening proto-Foil-2 further to proto-Foil-1 raises the loss by 0.017 nat (Figure 4, a1); Foil-1, which peaks, is 0.0020 nat above Foil-2 at 20B, about three standard errors

(Figure 3); and $(8, 8, 8)$ has the smallest average gain per pass of its series. Conversely, Foil-2, which has no peak, is the Foil with the lowest loss. These are few cases, so we state the relation as very likely rather than as a rule.

Takeaway 2

Within a range, routing confidence (MMR) is a diagnostic of router differentiation beyond load balance; its per-pass peak signals that the gains from looping are close to exhausted, a practical guide for designing looped models.

5 Related Work

Looped Transformers. Looped models apply the same layers repeatedly to an evolving hidden state, trading computation for depth and reuse for parameters, from the Universal Transformer to latent reasoning and scaling laws for recurrent depth (Dehghani et al., 2019; Giannou et al., 2023; Saunshi et al., 2025; Geiping et al., 2025; Zhu et al., 2025; Bae et al., 2025; McLeish et al., 2026; Schwethelm et al., 2026; Prairie et al., 2026). Their looped blocks are dense; ours is a sparse MoE, whose experts and attention must be arranged over layers and passes.

Sparse mixture-of-experts. Sparse MoE models activate only a few experts per token and thus grow their parameters far beyond their computation, and they now underlie most large language models (Fedus et al., 2022; Lepikhin et al., 2021; Zoph et al., 2022; Jiang et al., 2024; DeepSeek-AI, 2024; Qwen Team, 2025; Muennighoff et al., 2025; Kimi Team, 2025; 2026; GLM-5 Team, 2026; MiniMax, 2026); the granularity of the experts is a further design dimension, splitting experts into smaller ones and activating more of them (Dai et al., 2024; He, 2024; Ludziejewski et al., 2024). Earlier work combines looping with experts by turning the layers of a looped Transformer into shared mixtures of experts or by looping a whole MoE block (Csordás et al., 2024; Chen et al., 2026b); MoEUT ablates how many distinct consecutive layers form its repeated group and uses two for its smaller models. Other work ties experts across neighbouring layers (Jaggi, 2026; Tan et al., 2025; Chen et al., 2026c): MoRE lets adjacent layers share one larger expert pool, each layer keeping its own router (Qiu et al., 2026), and Megrez2 is similar (Li et al., 2025). In dense models, One Wide FFN shares one widened feed-forward layer across the encoder layers while keeping per-layer attention (Pires et al., 2023), a dense counterpart of flattening with untied attention. Each proposes one form of sharing, but none compares, at fixed expert parameters and compute, the layout (E, D, L) of the looped block and whether its attention is shared across passes; Jaggi (2026) and Megrez2 share experts with per-layer attention, as we do, but start from compressing deep models rather than varying the layout of a looped block.

Expert utilisation. Load balance is usually maintained by auxiliary losses, capacity limits or bias-based balancing without an auxiliary loss (Fedus et al., 2022; Zoph et al., 2022; Wang et al., 2024); beyond imbalance, experts can degrade through representation collapse or a router without preference (Chi et al., 2022; Li et al., 2026), and recent routing and balancing methods build on the same score-distribution and effective-count views (Shahout et al., 2025; Nguyen et al., 2026; Wu et al., 2026). We keep a standard balance metric, add MMR, which measures confidence against the median of all experts, and a traffic-matched masking test, and interpret them only through the three readings of Section 2.3.

6 Conclusion

How should a MoE be looped? We answer with Foil, which flattens the experts into fewer, wider layers looped more often and unties the attention across passes. At equal parameters and compute, Foil outperforms the unflattened looped baseline, its loss improves monotonically with flattening at 100B tokens, and at every shape it beats the attention-sharing proto-Foil in loss, downstream accuracy and, at 20B, routing balance and confidence. Our ablations show that widening and looping amplify each other, that load balance alone does not indicate healthy specialisation while MMR moves with the looping gain, and that its per-pass peak marks where flattening gains least. Limitations and future work are discussed in Appendix J.

AI USE STATEMENT

In this work, we used generative AI tools (large language models) for the following tasks: implementing methods (training, evaluation and analysis code), assisting with translation of text drafted by the authors in another language, providing feedback on the experimental design, and interpreting results by tabulating and cross-checking numerical outputs. We have not used generative AI tools to clean or reformat the training or evaluation datasets. Additionally, we used generative AI tools for literature search, for drafting and editing prose, and for producing figures and tables from our own results. We have reviewed all AI-assisted work: all code was tested and verified by the authors, every number in the paper was checked against the experiment logs, and all claims and conclusions were formulated and confirmed by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Appendix C gives the training data, tokenizer, optimiser, learning-rate schedule and auxiliary-loss coefficients, and Table 3 lists every model configuration with its parameter counts. Appendix A states the formal setup and derives the identities and bounds of Section 2.1. Appendices E and I describe the downstream-evaluation and expert-masking protocols, and Appendix B defines the routing metrics. Code, configurations and evaluation scripts will be released in the anonymous repository linked in the abstract. Limitations are discussed in Appendix J.

REFERENCES

Sangmin Bae, Adam Fisch, Hrayr Harutyunyan, Ziwei Ji, Seungyeon Kim, and Tal Schuster. Relaxed recursive Transformers: Effective parameter sharing with layer-wise LoRA. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/54d6a55225cebbdc16fbb0e45c5bdf2b-Abstract-Conference.html.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Agustín Piqueres Lajarín, Hynek Kydlíček, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Ben Burtenshaw, Clémentine Fourrier, Haojun Zhao, Hugo Larcher, Mathieu Morlon, Cyril Zakka, Colin Raffel, Leandro Von Werra, and Thomas Wolf. SmolLM2: When smol goes big — data-centric training of a fully open small language model. In *Conference on Language Modeling*, 2025. URL https://colm.cc/virtual/2025/poster/816.

Lizhang Chen, Jonathan Li, Qi Wang, Runlong Liao, Shuozhe Li, Chen Liang, Ni Lao, and Qiang Liu. ϕ-balancing for mixture-of-experts training. arXiv preprint arXiv:2605.15403, 2026a. URL https://arxiv.org/abs/2605.15403.

Wenkai Chen, Tianshu Li, Wenyong Huang, Yichun Yin, Lifeng Shang, and Chengwei Qin. LoopMoE: Unifying iterative computation with mixture-of-experts for language modeling. arXiv preprint arXiv:2606.04438, 2026b. URL https://arxiv.org/abs/2606.04438.

Yilong Chen, Naibin Gu, Junyuan Shang, Zhenyu Zhang, Yuchen Feng, Jiawei Sheng, Tingwen Liu, Shuohuan Wang, Yu Sun, Hua Wu, and Haifeng Wang. Mixture of universal experts: Scaling virtual width via depth-width transformation. arXiv preprint arXiv:2603.04971, 2026c. URL https://arxiv.org/abs/2603.04971.

Zewen Chi, Li Dong, Shaohan Huang, Damai Dai, Shuming Ma, Barun Patra, Saksham Singhal, Payal Bajaj, Xia Song, Xian-Ling Mao, Heyan Huang, and Furu Wei. On the representation collapse of sparse mixture of experts. In *Advances in Neural Information Processing Systems*, volume 35, pp. 34600–34613. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/df4f371f1f89ec8ba5014b3310578048-Abstract-Conference.html.

Róbert Csordás, Kazuki Irie, Jürgen Schmidhuber, Christopher Potts, and Christopher D. Manning. MoEUT: Mixture-of-experts Universal Transformers. In *Advances in Neural Information Processing Systems*, volume 37, pp. 28589–28614. Curran Associates, Inc., 2024. URL

https://proceedings.neurips.cc/paper_files/paper/2024/hash/321387ba926b8e58d3591c0aeb52ffc2-Abstract-Conference.html.

Damai Dai, Chengqi Deng, Chenggang Zhao, Runxin Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y. Wu, Zhenda Xie, Y. K. Li, Panpan Huang, Fuli Luo, Chong Ruan, Zhifang Sui, and Wenfeng Liang. DeepSeekMoE: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1280–1297. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.70. URL https://aclanthology.org/2024.acl-long.70/.

DeepSeek-AI. DeepSeek-V3 technical report. arXiv preprint arXiv:2412.19437, 2024. URL https://arxiv.org/abs/2412.19437.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal Transformers. In *International Conference on Learning Representations*, 2019. URL https://iclr.cc/virtual/2019/poster/1068.

William Fedus, Barret Zoph, and Noam Shazeer. Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL https://jmlr.org/papers/v23/21-0998.html.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up test-time compute with latent reasoning: A recurrent depth approach. In *Advances in Neural Information Processing Systems*, volume 38, pp. 41340–41391. Curran Associates, Inc., 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/3b01972cf31e6fa0fe29e4b8b5c2a0a1-Abstract-Conference.html.

Amir Gholami, Zhewei Yao, Sehoon Kim, Coleman Hooper, Michael W. Mahoney, and Kurt Keutzer. AI and memory wall. *IEEE Micro*, 44(3):33–39, 2024. doi: 10.1109/MM.2024.3373763. URL https://arxiv.org/abs/2403.14123.

Angeliki Giannou, Shashank Rajput, Jy-Yong Sohn, Kangwook Lee, Jason D. Lee, and Dimitris Papailiopoulos. Looped Transformers as programmable computers. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 11398–11442. PMLR, 2023. URL https://proceedings.mlr.press/v202/giannou23a.html.

GLM-5 Team. GLM-5: from vibe coding to agentic engineering. arXiv preprint arXiv:2602.15763, 2026. URL https://arxiv.org/abs/2602.15763.

Xu Owen He. Mixture of a million experts. arXiv preprint arXiv:2407.04153, 2024. URL https://arxiv.org/abs/2407.04153.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Thomas Hennigan, Eric Noland, Katherine Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karén Simonyan, Erich Elsen, Oriol Vinyals, Jack Rae, and Laurent Sifre. An empirical analysis of compute-optimal large language model training. In *Advances in Neural Information Processing Systems*, volume 35, pp. 30016–30030, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/c1e2faff6f588870935f114ebe04a3e5-Abstract-Conference.html.

Shengding Hu, Yuge Tu, Xu Han, Ganqu Cui, Chaoqun He, Weilin Zhao, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Xinrong Zhang, Zhen Leng Thai, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, Chao Jia, Guoyang Zeng, Dahai Li, Zhiyuan Liu, and Maosong Sun. MiniCPM: Unveiling the potential of small language models with scalable training strategies. In *Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=3X2L2TFr0f.

Benhao Huang, Chufan Shi, Junlin Chen, Shicheng Wen, Zhengzhong Liu, Eric Xing, and Xuezhe Ma. Towards looped models done right—part I: Topology, input injection, recurrent-state design. Institute of Foundation Models blog, 2026. URL https://ifm-research.notion.site/Towards-Looped-Models-Done-Right-3ade511912ec8128987dfeb7a5580043. Blog post (Institute of Foundation Models), not peer-reviewed; accessed 2026-09-25.

Martin Jaggi. Tying the loop – tied expert layers in mixture-of-experts language models. arXiv preprint arXiv:2606.16825, 2026. URL https://arxiv.org/abs/2606.16825.

Rajendra K. Jain, Dah-Ming W. Chiu, and William R. Hawe. A quantitative measure of fairness and discrimination for resource allocation in shared computer system. Technical Report DEC-TR-301, Digital Equipment Corporation, September 1984. URL https://arxiv.org/abs/cs/9809099.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts. arXiv preprint arXiv:2401.04088, 2024. URL https://arxiv.org/abs/2401.04088.

Kimi Team. Kimi K2: Open agentic intelligence. arXiv preprint arXiv:2507.20534, 2025. URL https://arxiv.org/abs/2507.20534.

Kimi Team. Kimi K3: Open frontier intelligence. arXiv preprint arXiv:2607.24653, 2026. URL https://arxiv.org/abs/2607.24653.

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. URL https://iclr.cc/virtual/2021/poster/3196.

Boxun Li, Yadong Li, Zhiyuan Li, Congyi Liu, Weilin Liu, Guowei Niu, Zheyue Tan, Haiyang Xu, Zhuyu Yao, Tao Yuan, Dong Zhou, Yueqing Zhuang, Bo Zhao, Guohao Dai, and Yu Wang. Megrez2 technical report. arXiv preprint arXiv:2507.17728, 2025. URL https://arxiv.org/abs/2507.17728.

Linwei Li, Hongye Jin, Binxuan Huang, Xiaotian Han, and Xin Liu. The death of z-loss in modern LLMs: a story of expert collapse and specialization. Blog post, 2026. URL https://alltoall.notion.site/z-loss-in-llms-a-story-of-expert-collapse-and-specialization. Not peer-reviewed; accessed 2026-09-25.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://iclr.cc/virtual/2019/poster/935.

Jan Ludziejewski, Jakub Krajewski, Kamil Adamczewski, Maciej Pióro, Michał Krutul, Szymon Antoniak, Kamil Ciebiera, Krystian Król, Tomasz Odrzygóźdź, Piotr Sankowski, Marek Cygan, and Sebastian Jaszczur. Scaling laws for fine-grained mixture of experts. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 33270–33288. PMLR, 2024. URL https://proceedings.mlr.press/v235/ludziejewski24a.html.

Sean McLeish, Ang Li, John Kirchenbauer, Dayal Singh Kalra, Brian Bartoldson, Bhavya Kailkhura, Avi Schwarzschild, Jonas Geiping, Tom Goldstein, and Micah Goldblum. Teaching pretrained language models to think deeper with retrofitted recurrence. In *Conference on Language Modeling*, 2026. URL https://colm.cc/virtual/2026/poster/2181.

MiniMax. The MiniMax-M2 series: Mini activations unleashing max real-world intelligence. arXiv preprint arXiv:2605.26494, 2026. URL https://arxiv.org/abs/2605.26494.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, Yuling Gu, Shane Arora, Akshita Bhagia, Dustin Schwenk, David Wadden, Alexander Wettig, Binyuan Hui, Tim Dettmers, Douwe Kiela, Ali Farhadi, Noah A. Smith, Pang Wei Koh, Amanpreet Singh, and Hannaneh Hajishirzi. OLMoE: Open mixture-of-experts language models. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/9b224ace8963c9385ad5e2b5c9039b97-Abstract-Conference.html.

Nam V. Nguyen, Thong T. Doan, Luong Tran, Van Nguyen, and Quang Pham. LibMoE: A library for comprehensive research on mixture of experts in large language models. *Transactions on Machine Learning Research*, 2026. URL https://openreview.net/forum?id=PB2ju8tq0n.

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. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, volume 37, pp. 30811–30849, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/370df50ccfdf8bde18f8f9c2d9151bda-Abstract-Datasets_and_Benchmarks_Track.html.

Telmo Pires, António Vilarinho Lopes, Yannick Assogba, and Hendra Setiawan. One wide feedforward is all you need. In *Proceedings of the Eighth Conference on Machine Translation*, pp. 1031–1044, 2023. doi: 10.18653/v1/2023.wmt-1.98. URL https://aclanthology.org/2023.wmt-1.98/.

Hayden Prairie, Zachary Novack, Taylor Berg-Kirkpatrick, and Daniel Y. Fu. Parcae: Scaling laws for stable looped language models. arXiv preprint arXiv:2604.12946, 2026. URL https://arxiv.org/abs/2604.12946.

Eric S. Qiu, Utku Umur Acikalin, Justin Lovelace, Christian Belardi, Arjun B. Mulchandani, Carla P. Gomes, and Kilian Q. Weinberger. MoRE: Mixture of reused experts. In *Conference on Language Modeling*, 2026. URL https://arxiv.org/abs/2609.18176.

Qwen Team. Qwen3 technical report. arXiv preprint arXiv:2505.09388, 2025. URL https://arxiv.org/abs/2505.09388.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. In *Advances in Neural Information Processing Systems*, volume 34, pp. 8583–8595. Curran Associates, Inc., 2021. URL https://papers.nips.cc/paper_files/paper/2021/hash/48237d9f2dea8c74c2a72126cf63d933-Abstract.html.

Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. Reasoning with latent thoughts: On the power of looped Transformers. In *International Conference on Learning Representations*, 2025. URL https://iclr.cc/virtual/2025/poster/28971.

Kristian Schwethelm, Daniel Rückert, and Georgios Kaissis. How much is one recurrence worth? iso-depth scaling laws for looped language models. arXiv preprint arXiv:2604.21106, 2026. URL https://arxiv.org/abs/2604.21106.

Rana Shahout, Colin Cai, Yilun Du, Minlan Yu, and Michael Mitzenmacher. From score distributions to balance: Plug-and-play mixture-of-experts routing. arXiv preprint arXiv:2510.03293, 2025. URL https://arxiv.org/abs/2510.03293.

Zheyue Tan, Zhiyuan Li, Tao Yuan, Dong Zhou, Weilin Liu, Yueqing Zhuang, Yadong Li, Guowei Niu, Cheng Qin, Zhuyu Yao, Congyi Liu, Haiyang Xu, Boxun Li, Guohao Dai, Bo Zhao, and Yu Wang. ReXMoE: Reusing experts with minimal overhead in mixture-of-experts. arXiv preprint arXiv:2510.17483, 2025. URL https://arxiv.org/abs/2510.17483.

Kushal Thaman. One must imagine experts happy: Rebalancing neural routers via constrained optimization. In *ICLR 2025 Workshop on Sparsity in LLMs (SLLM): Deep Dive into Mixture of Experts, Quantization, Hardware, and Inference*, 2025. URL https://openreview.net/forum?id=gsAkhArtfT.

Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts. arXiv preprint arXiv:2408.15664, 2024. URL https://arxiv.org/abs/2408.15664.

Ziang Wu, Peng Jin, Qishen Yin, Munan Ning, Hao Li, Peizhen Zhang, and Li Yuan. Relax within, balance across: Geometry-guided load balancing for vision-language mixture-of-experts. arXiv preprint arXiv:2608.00574, 2026. URL https://arxiv.org/abs/2608.00574.

Rui-Jie Zhu, Zixuan Wang, Kai Hua, Tianyu Zhang, Ziniu Li, Haoran Que, Boyi Wei, Zixin Wen, Fan Yin, He Xing, Lu Li, Jiajun Shi, Kaijing Ma, Shanda Li, Taylor Kergan, Andrew Smith, Xingwei Qu, Mude Hui, Bohong Wu, Qiyang Min, Hongzhi Huang, Xun Zhou, Wei Ye, Jiaheng Liu, Jian Yang, Yunfeng Shi, Chenghua Lin, Enduo Zhao, Tianle Cai, Ge Zhang, Wenhao Huang, Yoshua Bengio, and Jason Eshraghian. Scaling latent reasoning via looped language models. arXiv preprint arXiv:2510.25741, 2025. URL https://arxiv.org/abs/2510.25741.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. ST-MoE: Designing stable and transferable sparse expert models. arXiv preprint arXiv:2202.08906, 2022. URL https://arxiv.org/abs/2202.08906.

A Formal setup and resource accounting

This section states the looped block of Section 2.1 formally and derives the identities and bounds used there.

Problem formulation. We seek to improve sparse mixture-of-experts language models by reorganising how their parameters are stored and reused. As in Section 2.1, E is the number of experts in each physical layer of the looped block, D the number of layers in the block, L the number of passes through it, and k the number of experts selected per token at each layer visit. Given a token sequence $x_{1:T}$, the prediction objective is the next-token negative log-likelihood

$$\mathcal{L}_{\mathrm{LM}} = -\mathbb{E}_{x_{1:T}}\Big[\frac{1}{T}\sum_{t=1}^{T}\log p_\theta(x_t \mid x_{<t})\Big]. \tag{1}$$

The goal is to lower this loss while controlling the stored expert parameters and the selected-expert computation per token; the quantities below are those that Foil (Section 2.2) holds fixed or enlarges.

The looped block and attention sharing. Let $H_d^{(\ell)}$ be the hidden states of the sequence after physical layer d on pass ℓ, with $H_0^{(1)}$ the output of the prelude. Write $\mathcal{A}_d^{(\ell)}$ for the attention sublayer of layer d on pass ℓ and $\mathcal{M}_d$ for its MoE sublayer, each including normalisation and the residual connection. The looped block computes

$$H_d^{(\ell)} = \mathcal{M}_d\big(\mathcal{A}_d^{(\ell)}(H_{d-1}^{(\ell)})\big), \qquad H_0^{(\ell)} = H_D^{(\ell-1)} \quad (\ell > 1), \tag{2}$$

and passes $H_D^{(L)}$ to the coda; attention, routing and expert outputs are recomputed on every visit. The experts and router of $\mathcal{M}_d$ are shared across passes, but their inputs change with ℓ, so sharing does not force the same expert selections on different passes. Tied attention imposes $\mathcal{A}_d^{(\ell)} = \mathcal{A}_d$ for all ℓ; Foil gives every pass its own attention map. At fixed (E, D, L), every tied model is recovered from an untied one by setting its per-pass attention weights equal, so the tied function class is contained in the untied one; this guarantees neither strict inclusion nor better optimisation. Different flattened shapes impose different sharing constraints, so the containment does not extend across shapes.

Routing. Let $u_{t,d}^{(\ell)}$ be the normalised input of token t to the MoE sublayer of layer d on pass ℓ, $f_{d,e}$ the e-th expert of that layer, and g_d and $w_{d,e}$ its router scores and expert combination weights. The selected experts and their combined output are

$$S_{t,d}^{(\ell)} = \mathrm{TopK}\big(g_d(u_{t,d}^{(\ell)}), k\big), \tag{3}$$

$$y_{t,d}^{(\ell)} = \sum_{e \in S_{t,d}^{(\ell)}} w_{d,e}(u_{t,d}^{(\ell)})\, f_{d,e}(u_{t,d}^{(\ell)}), \tag{4}$$

and $\mathcal{M}_d$ adds $y_{t,d}^{(\ell)}$ to the residual stream.

Resource and routing identities. Let each expert contain p_{exp} parameters, and assume unrestricted top-k routing with $1 \le k \le E$, exactly k distinct experts executed per visit, and no dropped assignments. Counting each physical expert once for storage, each layer visit once for effective depth, and each selected expert once per call gives the identities of Section 2.1,

$$\begin{aligned} E_{\text{real}} &= E \times D, & P_{\text{experts}} &= E \times D \times p_{\text{exp}}, \\ D_{\text{eff}} &= D \times L, & E_{\text{comp}} &= k \times D \times L, \\ E_{\text{eq}} &= E \times D \times L, & |\mathcal{R}_{\text{formal}}| &= \binom{E}{k}^{D \times L}, \end{aligned}$$

where $\mathcal{R}_{\text{formal}}$ is the set of formal routes of one token, the ordered sequences $(S_{t,d}^{(\ell)})_{d \le D,\, \ell \le L}$ of unordered top-k selections; its size follows from the $\binom{E}{k}$ choices at each of the $D \times L$ visits. A trained model need not realise every route, and different routes can implement the same function; likewise, E_{eq} counts parameter-tied expert slots in the unrolled computation, not independently learned experts. Holding $E \times D$, $D \times L$ and k fixed while increasing E enlarges $|\mathcal{R}_{\text{formal}}|$ without increasing the number of expert calls, but this alone establishes neither greater functional capacity nor lower loss.

Bounds on expert coverage. The number of distinct experts token t reaches over its passes satisfies

$$k \times D \;\le\; U_t = \sum_{d=1}^{D} \Big| \bigcup_{\ell=1}^{L} S_{t,d}^{(\ell)} \Big| \;\le\; D \min(E, kL). \tag{5}$$

In each layer the union contains the k distinct experts of any one visit, and it contains at most the E experts of the layer and at most the kL selections made over the L visits; summing over the D layers gives both bounds.

Three comparison regimes. The identities separate three ways of comparing looped MoE models.

- *Additional passes at fixed parameters.* Holding E and D fixed while increasing L preserves the stored parameters when all block parameters are shared and no pass-specific parameters are introduced. Effective depth and expert calls increase in proportion to L. This comparison measures the benefit of additional computation through parameter reuse.
- *Parameter–computation trade-offs.* Holding E and $D \times L$ fixed while reducing D and increasing L preserves the number of layer evaluations and expert calls but reduces the number of stored experts. With unchanged sublayer dimensions and sequence length, the leading forward arithmetic is matched.
- *Flattening at fixed expert parameters.* For an integer $a \ge 1$ dividing D, the map $(E, D, L) \mapsto (aE, D/a, aL)$ preserves $E \times D$, $D \times L$ and $k \times D \times L$ while multiplying E_{eq} by a. The per-layer cap on distinct experts rises from $\min(E, kL)$ to $a \min(E, kL)$, whereas the whole-block cap stays $D \min(E, kL)$. Flattening therefore enlarges the pool available at each routing decision without raising the maximum number of distinct experts a token can reach across the block. Each flattening step of Section 2.2 uses $a = 2$.

Total parameters of Foil. The total parameter count of Foil is

$$P_{\text{total}} = P_{\text{outside}} + E \times D \times p_{\text{exp}} + D \times L \times P_{\text{attn}} + D \times P_{\text{router}}(E), \tag{6}$$

where P_{outside} counts the parameters outside the looped block and the remaining terms count its experts, attention and routers (our RMSNorm layers have no gain and hence no parameters). With attention shared across passes (proto-Foil), the attention term is $D \times P_{\text{attn}}$ instead. For a bias-free linear router of input width d_{model}, $P_{\text{router}}(E) = d_{\text{model}} \times E$, so the router total is also preserved by flattening. Since $D \times L$ is fixed along the flattening sequence, Foil keeps the attention parameters, and hence the total, unchanged, whereas with shared attention they decrease with D. Likewise, fixed $D \times L$ and $k \times D \times L$ preserve the leading attention and selected-expert arithmetic, but not router scoring: a linear router produces $E \times D \times L$ expert scores per token, at a cost of $O(E \times D \times L \times d_{\text{model}})$, which grows along the sequence. E_{comp} therefore measures expert computation, not total FLOPs including routing and dispatch.

B EXPERT-COLLAPSE DIAGNOSTICS: DEFINITIONS AND LIMITS

All quantities are empirical summaries of the same $N = 255{,}500$ probe tokens at the final checkpoint, restricted to the recurrent core. Index tokens by t, physical layers of the core by d and passes by ℓ, as in Section 2. For each decision, the finite router logits $z^{(\ell)}_{t,d,i}$ define $p^{(\ell)}_{t,d,i} = \exp(z^{(\ell)}_{t,d,i}) / \sum_j \exp(z^{(\ell)}_{t,d,j})$. The recorded top-$k$ set $S^{(\ell)}_{t,d}$ contains exactly $k = 2$ distinct experts, with ties resolved by the model's routing rule. Load uses these selections, not probability mass or the renormalised mixture weights.

Class 0: coverage of physical experts. The distinct-expert count is

$$U_t = \sum_{d=1}^{D} \left| \bigcup_{\ell=1}^{L} S^{(\ell)}_{t,d} \right|, \qquad kD \le U_t \le D \min(E, kL). \tag{7}$$

For the bounds see Equation (5). We report $\bar{U} = N^{-1} \sum_t U_t$ and $\bar{U}/[D \min(E, kL)]$. At $L = 1$ the ratio is identically one and is omitted from Table 10. Along the flattening sequence the number of distinct experts a token reaches falls (Base 24.1, Foil-3 17.8, Foil-2 12.5, Foil-1 9.7 at 20B; Table 10): flattening enlarges the pool of each routing decision, not the number of experts a token uses. The increase with additional passes reported in the introduction (Figure 1) is at fixed shape.

Class 1: load concentration. The count share and effective expert count of layer d are

$$q_{di} = \frac{1}{NLk} \sum_{t=1}^{N} \sum_{\ell=1}^{L} \mathbf{1}\{i \in S^{(\ell)}_{t,d}\}, \qquad N_{2,d} = \frac{1}{\sum_{i=1}^{E} q_{di}^2}, \qquad B_{2,d} = \frac{N_{2,d}}{E}. \tag{8}$$

$N_{2,d}$ is the inverse Simpson effective count (Wu et al., 2026, Eq. S10); it equals s for traffic uniformly spread over s experts. Its normalisation $B_{2,d}$ is Jain's fairness index (Jain et al., 1984). Since $\sum_i q_{di} = 1$ and $0 \le q_{di} \le 1/k$, Cauchy–Schwarz and $q_{di}^2 \le q_{di}/k$ give

$$\frac{1}{E} \le \sum_i q_{di}^2 \le \frac{1}{k}, \qquad \frac{k}{E} \le B_{2,d} \le 1. \tag{9}$$

$B_{2,d} = 1$ if and only if the load is uniform. The lower endpoint $B_{2,d} = k/E$ holds if and only if the same k experts are selected at every decision in that layer, the maximally concentrated load permitted by top-k dispatch. Intermediate values quantify concentration without specifying a universal failure threshold. Unlike MaxVio, which measures the largest relative overload (Wang et al., 2024, Eq. 4), B_2 depends on all expert shares.

Counts are pooled over passes before taking the reciprocal. The model score is

$$B_{\mathrm{model},2} = \frac{\sum_{d=1}^{D} N_{2,d}}{DE} = \frac{1}{D} \sum_{d=1}^{D} B_{2,d}. \tag{10}$$

Pooling can conceal concentration within a pass: when $E = kL$, partition the experts into L disjoint groups of k and assign all tokens to group ℓ on pass ℓ. This gives pooled $B_2 = 1$, although each pass has $B_2 = k/E$. A per-pass score instead uses $q_{d,\ell,i} = (Nk)^{-1} \sum_t \mathbf{1}\{i \in S^{(\ell)}_{t,d}\}$; averaging these scores generally differs from pooling counts.

Class 2: median margin ratio. Suppress the decision indices and order logits as $z_{(1)} \ge \cdots \ge z_{(E)}$. With $m = \lceil E/2 \rceil$, define

$$R = \frac{p_{(1)}}{p_{(m)}} = \exp\big(z_{(1)} - z_{(m)}\big), \qquad \mathrm{MMR}(\mathcal{I}) = \exp\left(\frac{1}{|\mathcal{I}|} \sum_{a \in \mathcal{I}} \big[z_{a,(1)} - z_{a,(m)} \big] \right). \tag{11}$$

All experimental pools have even E: the denominator is the upper of the two central probabilities, at descending rank $E/2$, rather than their arithmetic mean. The nonempty index set $\mathcal{I}$ contains the token–layer–pass decisions being summarised; the full-core score weights all NDL decisions

equally. This is the geometric mean of decision-level ratios, not a ratio of averaged probabilities. The logit form cancels the softmax normaliser and avoids division by probabilities that may underflow.

MMR is at least one and equals one exactly when the top m logits coincide at every included decision. Complete router indifference, $p_i = 1/E$ for every expert at every decision, is sufficient but not necessary: $z = (0, 0, 0, 0, -c, -c, -c, -c)$ with $c > 0$ also gives $R = 1$. Replacing every logit vector by $\alpha z + \beta \mathbf{1}$, with $\alpha > 0$ and the same tie rule, leaves top-k selections and B_2 unchanged but sends MMR to MMR^{α}. MMR measures separation from the median rank and depends on logit scale; it does not measure the margin between the selected and unselected experts.

Joint interpretation. Load concentration and weak router preference are distinct routing phenomena. They also differ from the collapse of hidden representations studied by Chi et al. (2022). Neither B_2 nor MMR observes expert outputs: even identical expert functions can coexist with balanced, confident routing. Consequently, the scores describe traffic and router preference, while the masking test in Appendix I measures sensitivity to expert removal. Table 10 reports the diagnostics; comparisons use the same probe and aggregation, and absolute comparisons in the main text are restricted to equal shapes because both scores depend on E.

Related metrics used in prior work. Besides the three metrics of Section 2.3, the literature uses further load-balance and routing-confidence metrics; Table 2 lists the ones we also computed, with their ranges under top-k routing. For one routing decision, p_i are the router's softmax probabilities over the E experts (temperature 1), sorted as $p_{(1)} \geq \cdots \geq p_{(E)}$; S is the selected set and $a_i = p_i / \sum_{j \in S} p_j$ the renormalised weight of $i \in S$; q_i are the load shares of Equation (8). Our models use no capacity limit, so no assignment is ever dropped; Table 11 gives the values of the other related metrics for every small-tier model.

Table 2: Related metrics used in prior work: load balance (top four rows) and routing confidence (bottom five rows). Ranges are for top-k routing with $k < E$. Cited works show prior use of each quantity; the exact formulas and normalisations are as stated here.

Metric	Definition	Range, direction	Source
normalised load entropy B_H	$-\sum_i q_i \ln q_i / \ln E$	$[\ln k / \ln E, 1]$, higher more even	(Nguyen et al., 2026)
MaxVio	$E \max_i q_i - 1$	$[0, E/k - 1]$, lower less overload	(Wang et al., 2024)
normalised Gini G^*	$\sum_{i,j} \lvert q_i - q_j \rvert / [2(E-1)]$	$[0, (E-k)/(E-1)]$, lower more even	(Chen et al., 2026a)
dropped-assignment rate	dropped / requested assignments	$[0, 1]$; not applicable (no drops)	(Fedus et al., 2022; Lepikhin et al., 2021)
selected-expert probability	$p_i, i \in S$	$[0, 1]$, higher more support	(Fedus et al., 2022; Riquelme et al., 2021)
selected-set mass	$\sum_{i \in S} p_i$	$[k/E, 1]$, higher more concentrated	(Shahout et al., 2025)
boundary margin	$\ln p_{(k)} - \ln p_{(k+1)}$	$[0, \infty)$, higher clearer selection	(Thaman, 2025)
full-pool entropy concentration	$1 - H(p) / \ln E$	$[0, 1]$, higher sharper	(Shahout et al., 2025)
selected-weight concentration	$1 - H(a) / \ln k$	$[0, 1]$, higher one expert dominates	(Nguyen et al., 2026)

C Experimental configuration

Data. All models are trained on the `sample-100BT` subset of FineWeb-Edu (Penedo et al., 2024), tokenised with the SmolLM2 tokenizer (Ben Allal et al., 2025) (vocabulary 49,152), which yields 101.7B tokens. Sequences are packed to a context length of 4,096 tokens. Every run reads the data in the same fixed order, so that at every optimiser step all runs have seen the same tokens; this makes the step-wise paired comparisons possible.

Optimisation. We use AdamW (Loshchilov & Hutter, 2019) with $\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay 0.1 and gradient clipping at norm 1.0, in bfloat16. The global batch is 96 sequences, i.e. 393,216 tokens per step. The learning rate follows a warmup–stable–decay schedule (Hu et al., 2024): a linear warmup over 1,000 steps to a peak of 3×10^{-4}, a constant phase, and a decay over the last 10% of the run with a $1 - \sqrt{\cdot}$ shape down to 5% of the peak. All models share this schedule, the initialisation seed (42) and the data order; the seed replicate changes only the initialisation seed, to 43.

Auxiliary losses. Every model adds to the language-modelling loss a load-balancing loss of the Switch Transformer form (Fedus et al., 2022) with coefficient 0.01 and a router z-loss (Zoph et al., 2022) with coefficient 0.001. Both are computed for every MoE layer on every pass and averaged over the unrolled depth. The losses we report are the language-modelling loss alone.

Table 3: Per-configuration hyperparameters and parameter counts. (E, D, L): experts per MoE layer, layers in the loop block, loop passes. $E_{\text{real}} = E \cdot D$ (distinct experts), $E_{\text{comp}} = D \cdot L \cdot k$ (expert calls per token), $E_{\text{eq}} = E \cdot D \cdot L$. Attention: *shared* = one set of attention weights reused by every pass; *untied* = one set per pass. Parameter columns in millions, rounded independently: total; experts (including their routers) and attention inside the loop block; the non-looped head and tail layers; "other" = token embeddings and output projection. Run ids are the identifiers used in our logs. Base width: $d_{\text{model}} = 1024$, 16 attention heads, expert hidden width 1536, top-$k = 2$, one head and one tail layer with 8 experts each, vocabulary 49,152. [a] $E = 4$ with $k = 2$: half of the experts in a layer are active for every token. [b] Two head and two tail layers. [c] Not looped ($L = 1$).

Model	Run id	(E, D, L)	Attention	E_{real}	E_{comp}	E_{eq}	Total	Loop experts	Loop attn.	Head/ tail	Other
Base width ($d_{model} = 1024$): Foil family											
Base-tied	S1	(8,8,2)	shared	64	32	128	520.2	302.1	33.6	83.9	100.7
proto-Foil-3	S2	(16,4,4)	shared	64	32	256	503.4	302.1	16.8	83.9	100.7
proto-Foil-2	S3	(32,2,8)	shared	64	32	512	495.0	302.1	8.4	83.9	100.7
proto-Foil-1	S4	(64,1,16)	shared	64	32	1024	490.8	302.1	4.2	83.9	100.7
Base	U1	(8,8,2)	untied	64	32	128	553.7	302.1	67.1	83.9	100.7
Foil-3	U2	(16,4,4)	untied	64	32	256	553.7	302.1	67.1	83.9	100.7
Foil-2	U3	(32,2,8)	untied	64	32	512	553.7	302.1	67.1	83.9	100.7
Foil-1	U4	(64,1,16)	untied	64	32	1024	553.7	302.1	67.1	83.9	100.7
Base width ($d_{model} = 1024$): other grid models											
–	S6[c]	(8,8,1)	shared	64	16	64	520.2	302.1	33.6	83.9	100.7
–	S8	(8,8,3)	shared	64	48	192	520.2	302.1	33.6	83.9	100.7
–	S9	(8,8,4)	shared	64	64	256	520.2	302.1	33.6	83.9	100.7
–	N4	(8,8,8)	shared	64	128	512	520.2	302.1	33.6	83.9	100.7
–	S5	(16,4,2)	shared	64	16	128	503.4	302.1	16.8	83.9	100.7
–	N3	(16,4,6)	shared	64	48	384	503.4	302.1	16.8	83.9	100.7
–	N2	(16,4,8)	shared	64	64	512	503.4	302.1	16.8	83.9	100.7
–	S7	(8,4,2)	shared	32	16	64	352.4	151.0	16.8	83.9	100.7
–	S12	(8,4,4)	shared	32	32	128	352.4	151.0	16.8	83.9	100.7
–	N1	(8,4,8)	shared	32	64	256	352.4	151.0	16.8	83.9	100.7
–	S11[a]	(4,8,2)	shared	32	32	64	369.1	151.0	33.6	83.9	100.7
–	S13[a]	(4,8,4)	shared	32	64	128	369.1	151.0	33.6	83.9	100.7
–	S10	(16,8,2)	shared	128	32	256	822.2	604.1	33.6	83.9	100.7
–	S14[c]	(8,16,1)	shared	128	32	128	855.8	604.1	67.1	83.9	100.7
–	S15[b]	(8,8,2)	shared	64	32	128	604.1	302.1	33.6	167.8	100.7

Training length. The 20B-token stage runs 50,000 steps (19.66B tokens); the constant phase ends at step 45,000 and the decay occupies the remaining 5,000 steps. The 100B-token continuation starts from the step-45,000 checkpoint, taken before the decay, resumes the optimiser state and the data order, and lengthens the schedule to 254,313 steps (100.0B tokens), with the learning rate at its peak until step 228,882 and the same decay afterwards.

Reported loss. The reported loss of a model is its mean language-modelling loss over the last 2,000 training steps, and two models are compared by the mean of their step-wise loss difference over this window. Loss standard errors treat consecutive training steps as independent and are therefore understated.

D Experimental results

Seed replicate. Base-tied was retrained with only the initialisation seed changed (seed 43 instead of 42; both runs are in Table 4). The paired difference of the final-window loss is -0.0003 ± 0.0009 nat, and every difference we interpret as a result is at least 0.002 nat; smaller differences are reported as on par.

Table 4 collects the configuration, parameter count and final-window loss of every small-tier model. The three expert counts are $E_{\text{real}} = ED$ (distinct experts in the core), $E_{\text{comp}} = DLk$ (expert calls per token) and $E_{\text{eq}} = EDL$ (experts of the equivalent non-looped model).

Table 4: Main results of all small-tier models ($d_{\text{model}} = 1024$) at 20B tokens, and at 100B tokens where the model was continued. Loss: mean task loss over the final window (nat); perplexity is its exponential. Parameters in millions. [§]Repeated from an earlier group for comparison. Single seed except the seed-43 repeat of Base-tied.

Model	(E, D, L)	attention	params (M)	E_{real}	E_{comp}	E_{eq}	loss 20B	ppl 20B	loss 100B
Flattening, tied attention									
Base-tied	$(8, 8, 2)$	tied	520.2	64	32	128	2.688	14.71	2.522
proto-Foil-3	$(16, 4, 4)$	tied	503.4	64	32	256	2.679	14.58	2.520
proto-Foil-2	$(32, 2, 8)$	tied	495.0	64	32	512	2.684	14.64	2.531
proto-Foil-1	$(64, 1, 16)$	tied	490.8	64	32	1024	2.700	14.89	2.548
Flattening, untied attention									
Base	$(8, 8, 2)$	untied	553.7	64	32	128	2.666	14.38	2.511
Foil-3	$(16, 4, 4)$	untied	553.7	64	32	256	2.663	14.33	2.506
Foil-2	$(32, 2, 8)$	untied	553.7	64	32	512	2.657	14.25	2.501
Foil-1	$(64, 1, 16)$	untied	553.7	64	32	1024	2.659	14.28	2.499
Loop passes, $(E, D) = (8, 8)$									
–	$(8, 8, 1)$	tied	520.2	64	16	64	2.752	15.68	2.582
Base-tied[§]	$(8, 8, 2)$	tied	520.2	64	32	128	2.688	14.71	2.522
–	$(8, 8, 3)$	tied	520.2	64	48	192	2.650	14.16	–
–	$(8, 8, 4)$	tied	520.2	64	64	256	2.629	13.86	–
–	$(8, 8, 8)$	tied	520.2	64	128	512	2.596	13.41	–
Loop passes, $(E, D) = (16, 4)$									
–	$(16, 4, 2)$	tied	503.4	64	16	128	2.749	15.63	–
proto-Foil-3[§]	$(16, 4, 4)$	tied	503.4	64	32	256	2.679	14.58	2.520
–	$(16, 4, 6)$	tied	503.4	64	48	384	2.649	14.14	–
–	$(16, 4, 8)$	tied	503.4	64	64	512	2.638	13.99	–
Loop passes, $(E, D) = (8, 4)$									
–	$(8, 4, 2)$	tied	352.4	32	16	64	2.787	16.24	–
–	$(8, 4, 4)$	tied	352.4	32	32	128	2.720	15.17	–
–	$(8, 4, 8)$	tied	352.4	32	64	256	2.688	14.70	–
Pool size and depth at fixed $E_{\text{comp}} = 32$									
–	$(4, 8, 2)$	tied	369.1	32	32	64	2.728	15.30	–
Base-tied[§]	$(8, 8, 2)$	tied	520.2	64	32	128	2.688	14.71	2.522
–	$(16, 8, 2)$	tied	822.2	128	32	256	2.641	14.03	–
–[§]	$(8, 4, 4)$	tied	352.4	32	32	128	2.720	15.17	–
–	$(8, 16, 1)$	tied	855.8	128	32	128	2.625	13.81	2.463
Other controls									
–	$(4, 8, 4)$	tied	369.1	32	64	128	2.675	14.51	–
Base-tied, two-layer prelude/coda	$(8, 8, 2)$	tied	604.1	64	32	128	2.656	14.24	–
Base-tied, seed 43	$(8, 8, 2)$	tied	520.2	64	32	128	2.688	14.70	–

E Downstream evaluation

Final checkpoints are evaluated on 36 downstream tasks. A task is kept as valid only if all 35 models trained for 20B tokens (all widths) score, on average, at least five standard errors above its baseline (chance level, zero for open-ended completion, or the majority class for classification tasks), in the metric and setting that lies furthest above the baseline. The appendix reports seven valid tasks: the three representative tasks of the main text, and PROST, LAMBADA (OpenAI), SWAG and BLiMP. The three representative tasks of the main text take one task from each of three categories: word prediction (LAMBADA, standard split), sentence continuation (HellaSwag) and coreference resolution (XWinograd, English). Zero-shot is the main setting (Table 5). For 5-shot (Table 6), HellaSwag is averaged over three evaluation seeds, which change the choice of in-context examples; the other tasks have seed 0 only, and BLiMP and LAMBADA (OpenAI) have no 5-shot variant. Standard errors are those of the evaluation harness; a mean over tasks treats the tasks as independent, and the standard error of a difference between two models is $\sqrt{\text{se}_a^2 + \text{se}_b^2}$, which ignores that both models answer the same questions and is therefore conservative.

F Held-out validation loss

Held-out losses are computed at the final checkpoints on a FineWeb-Edu validation slice of 8.0M tokens, disjoint from the training data, and on Wikipedia, C4 and arXiv slices of 3–4M tokens each; the evaluation reports means only (Table 7).

Table 5: Zero-shot downstream accuracy (%, ±1 standard error) of the eight models of Table 3. SWAG, HellaSwag and PROST are scored by length-normalised accuracy, the other tasks by accuracy; the mean over tasks treats tasks as independent (standard error of a difference between two models about 0.31).

Task	Base-tied	proto-Foil-3	proto-Foil-2	proto-Foil-1	Base	Foil-3	Foil-2	Foil-1
20B tokens								
PROST	31.27 ±0.34	31.00 ±0.34	29.50 ±0.33	30.73 ±0.34	30.26 ±0.34	31.71 ±0.34	31.10 ±0.34	29.80 ±0.33
LAMBADA (standard)	23.17 ±0.59	24.22 ±0.60	25.48 ±0.61	23.73 ±0.59	25.89 ±0.61	26.14 ±0.61	26.66 ±0.62	26.14 ±0.61
LAMBADA (OpenAI)	32.08 ±0.65	32.33 ±0.65	32.23 ±0.65	31.09 ±0.64	35.30 ±0.67	34.54 ±0.66	33.86 ±0.66	34.29 ±0.66
SWAG	55.34 ±0.35	55.47 ±0.35	55.36 ±0.35	54.69 ±0.35	55.74 ±0.35	55.90 ±0.35	56.73 ±0.35	56.41 ±0.35
HellaSwag	39.55 ±0.49	40.41 ±0.49	40.72 ±0.49	39.86 ±0.49	40.82 ±0.49	41.30 ±0.49	41.07 ±0.49	41.46 ±0.49
BLiMP	82.18 ±0.13	82.07 ±0.13	81.09 ±0.14	81.15 ±0.14	81.94 ±0.13	81.98 ±0.13	82.14 ±0.13	82.40 ±0.13
XWinograd (en)	63.78 ±1.00	63.01 ±1.00	63.61 ±1.00	62.37 ±1.00	64.52 ±0.99	64.09 ±1.00	64.77 ±0.99	64.56 ±0.99
Mean, seven tasks	46.77 ±0.22	46.93 ±0.22	46.86 ±0.22	46.23 ±0.21	47.78 ±0.22	47.95 ±0.22	48.05 ±0.22	47.87 ±0.22
Mean, three representative tasks	42.17 ±0.42	42.55 ±0.42	43.27 ±0.42	41.99 ±0.42	43.74 ±0.42	43.84 ±0.42	44.17 ±0.42	44.05 ±0.42
100B tokens								
PROST	29.84 ±0.33	31.21 ±0.34	31.85 ±0.34	30.34 ±0.34	29.60 ±0.33	34.52 ±0.35	29.92 ±0.33	29.45 ±0.33
LAMBADA (standard)	29.85 ±0.64	29.32 ±0.63	28.31 ±0.63	26.57 ±0.62	30.43 ±0.64	31.38 ±0.65	30.18 ±0.64	31.83 ±0.65
LAMBADA (OpenAI)	36.44 ±0.67	38.42 ±0.68	37.16 ±0.67	36.23 ±0.67	39.24 ±0.68	39.80 ±0.68	39.61 ±0.68	39.98 ±0.68
SWAG	59.13 ±0.35	59.32 ±0.35	58.77 ±0.35	58.44 ±0.35	59.17 ±0.35	59.98 ±0.35	59.90 ±0.35	59.98 ±0.35
HellaSwag	47.17 ±0.50	47.07 ±0.50	46.59 ±0.50	45.68 ±0.50	47.23 ±0.50	47.41 ±0.50	48.06 ±0.50	48.43 ±0.50
BLiMP	82.12 ±0.13	81.14 ±0.14	80.24 ±0.14	81.64 ±0.13	81.97 ±0.14	82.13 ±0.13	81.66 ±0.13	81.82 ±0.13
XWinograd (en)	67.87 ±0.97	68.26 ±0.97	66.92 ±0.98	67.01 ±0.98	68.22 ±0.97	68.95 ±0.96	70.67 ±0.94	68.82 ±0.96
Mean, seven tasks	50.35 ±0.22	50.68 ±0.22	49.98 ±0.22	49.42 ±0.22	50.84 ±0.22	52.02 ±0.22	51.43 ±0.21	51.47 ±0.22
Mean, three representative tasks	48.30 ±0.42	48.22 ±0.42	47.27 ±0.42	46.42 ±0.42	48.63 ±0.42	49.25 ±0.42	49.64 ±0.41	49.69 ±0.42

Table 6: 5-shot downstream accuracy (%, ±1 standard error). HellaSwag is the mean over three evaluation seeds; †seed 0 only; – marks tasks without a 5-shot variant. The mean over five tasks excludes these two and treats tasks as independent (standard error of a difference between two models about 0.38). Metrics as in Table 5.

Task	Base-tied	proto-Foil-3	proto-Foil-2	proto-Foil-1	Base	Foil-3	Foil-2	Foil-1
20B tokens								
PROST†	33.24 ±0.34	31.50 ±0.34	26.96 ±0.32	27.88 ±0.33	30.35 ±0.34	29.96 ±0.33	29.12 ±0.33	29.50 ±0.33
LAMBADA (standard)†	22.71 ±0.58	23.17 ±0.59	24.84 ±0.60	22.34 ±0.58	24.86 ±0.60	24.41 ±0.60	24.80 ±0.60	24.24 ±0.60
LAMBADA (OpenAI)	–	–	–	–	–	–	–	–
SWAG†	54.41 ±0.35	55.22 ±0.35	54.85 ±0.35	54.33 ±0.35	54.96 ±0.35	55.30 ±0.35	55.59 ±0.35	55.55 ±0.35
HellaSwag	40.13 ±0.49	41.12 ±0.49	41.01 ±0.49	40.17 ±0.49	40.79 ±0.49	41.15 ±0.49	41.50 ±0.49	41.38 ±0.49
BLiMP	–	–	–	–	–	–	–	–
XWinograd (en)†	63.78 ±1.00	65.20 ±0.99	65.03 ±0.99	63.57 ±1.00	66.80 ±0.98	65.08 ±0.99	64.43 ±0.99	67.18 ±0.97
Mean, five tasks	42.85 ±0.27	43.24 ±0.27	42.54 ±0.27	41.66 ±0.27	43.55 ±0.27	43.18 ±0.27	43.09 ±0.27	43.57 ±0.27
Mean, three representative tasks	42.21 ±0.42	43.16 ±0.42	43.63 ±0.42	42.03 ±0.42	44.15 ±0.42	43.55 ±0.42	43.58 ±0.42	44.27 ±0.41
100B tokens								
PROST†	27.84 ±0.33	27.87 ±0.33	29.36 ±0.33	25.24 ±0.32	29.58 ±0.33	31.61 ±0.34	28.49 ±0.33	28.25 ±0.33
LAMBADA (standard)†	28.66 ±0.63	26.49 ±0.61	27.44 ±0.62	25.25 ±0.61	30.02 ±0.64	27.87 ±0.62	30.93 ±0.64	30.91 ±0.64
LAMBADA (OpenAI)	–	–	–	–	–	–	–	–
SWAG†	58.83 ±0.35	59.02 ±0.35	58.60 ±0.35	58.13 ±0.35	58.48 ±0.35	58.96 ±0.35	59.36 ±0.35	59.33 ±0.35
HellaSwag	47.52 ±0.50	47.56 ±0.50	47.07 ±0.50	46.20 ±0.50	47.61 ±0.50	47.95 ±0.50	48.78 ±0.50	48.48 ±0.50
BLiMP	–	–	–	–	–	–	–	–
XWinograd (en)†	69.81 ±0.95	71.14 ±0.94	69.68 ±0.95	68.09 ±0.97	71.53 ±0.94	71.31 ±0.94	72.77 ±0.92	72.13 ±0.93
Mean, five tasks	46.53 ±0.27	46.42 ±0.26	46.43 ±0.27	44.58 ±0.27	47.44 ±0.27	47.54 ±0.27	48.07 ±0.26	47.82 ±0.27
Mean, three representative tasks	48.66 ±0.41	48.40 ±0.41	48.06 ±0.41	46.51 ±0.42	49.72 ±0.41	49.04 ±0.41	50.83 ±0.41	50.51 ±0.41

G Ablation results

Ablation details. Table 8 lists the widening gains and the interaction of widening and looping behind Section 4.2; Table 9 lists the routing confidence and the looping gains of the $(8, 8, L)$ series behind Section 4.4.

H Routing metrics of all models

I Expert-masking evaluation

The masking test asks whether experts that receive little traffic are also of little value. It uses two disjoint samples of the same corpus: sample A selects the experts, sample B measures the loss. On sample A we count, for every physical expert of the recurrent core, how often it is selected into the top-k, with the passes pooled onto the physical expert; the prelude and coda layers are never masked.

Table 7: Held-out loss (nat) of the Foil family on a FineWeb-Edu validation slice and three domain slices, final checkpoints. The FineWeb-Edu slice reproduces the training-stream ordering: Foil-2 lowest at 20B, Foil-1 lowest at 100B, and the untied model below the tied one at every shape. At 100B, Wikipedia and C4 agree in direction: every Foil is below Base and every untied model is below its tied counterpart. arXiv is the exception: at 20B Foil-3 and the $(16, 4, 4)$ untied model are above their comparators, and Foil-3 remains above Base at 100B. Slice sizes 3–8M tokens; the evaluation reports means only, so no standard errors. Code and book slices are absent from the training corpus and are not reported. The slices are small and no standard errors are available, so differences of about 0.001 nat or less may be within noise.

Model	(E, D, L)	Training stream	FineWeb-Edu	Wikipedia	C4	arXiv
20B tokens						
Base-tied	(8,8,2)	2.6884	2.6307	2.6840	3.1288	2.9151
proto-Foil-3	(16,4,4)	2.6793	2.6198	2.6782	3.1190	2.9147
proto-Foil-2	(32,2,8)	2.6836	2.6240	2.6933	3.1251	2.9395
proto-Foil-1	(64,1,16)	2.7005	2.6414	2.7052	3.1410	2.9733
Base	(8,8,2)	2.6659	2.6077	2.6703	3.1043	2.8997
Foil-3	(16,4,4)	2.6627	2.6035	2.6704	3.1021	2.9337
Foil-2	(32,2,8)	2.6566	2.5984	2.6651	3.0952	2.8729
Foil-1	(64,1,16)	2.6586	2.6000	2.6731	3.0981	2.8765
100B tokens						
Base-tied	(8,8,2)	2.5218	2.4711	2.5408	2.9824	2.7153
proto-Foil-3	(16,4,4)	2.5202	2.4700	2.5430	2.9809	2.7447
proto-Foil-2	(32,2,8)	2.5305	2.4800	2.5533	2.9902	2.7795
proto-Foil-1	(64,1,16)	2.5481	2.4971	2.5632	3.0078	2.7933
Base	(8,8,2)	2.5108	2.4610	2.5376	2.9721	2.7127
Foil-3	(16,4,4)	2.5061	2.4552	2.5334	2.9649	2.7231
Foil-2	(32,2,8)	2.5012	2.4506	2.5135	2.9594	2.7084
Foil-1	(64,1,16)	2.4988	2.4492	2.5311	2.9584	2.7084

Table 8: Widening and looping, 20B tokens, tied attention. Gains are decreases of the final-window loss (nat), ± 1 standard error of the paired difference. Top: widening doubles E at fixed D and L; there are no $(4, 8, 8)$, $(16, 8, 4)$ or $(16, 8, 8)$ models. Bottom: for each 2×2 block, looping doubles L and "both" does the two steps together; interaction = both $-$ widening $-$ looping, with a standard error of about 0.0014 combined from the independent paired differences.

Widening	$L = 2$	$L = 4$	$L = 8$
$(8, 4, L) \to (16, 4, L)$	0.0384 ± 0.0011	0.0402 ± 0.0010	0.0491 ± 0.0012
$(4, 8, L) \to (8, 8, L)$	0.0393 ± 0.0010	0.0460 ± 0.0009	–
$(8, 8, L) \to (16, 8, L)$	0.0470 ± 0.0009	–	–

Start	widening	looping (passes)	both	interaction
$(8, 4, 2)$	0.0384	0.0678 ($2 \to 4$)	0.1080	+0.0018
$(8, 4, 4)$	0.0402	0.0321 ($4 \to 8$)	0.0812	+0.0089
$(4, 8, 2)$	0.0393	0.0530 ($2 \to 4$)	0.0990	+0.0067

Sorting the experts by this share from lowest to highest and adding them until the cumulative share is closest to $x\%$ of all routing requests gives the least-used group $T(x)$. Masking sets the routing probabilities of the masked experts to zero before the top-k selection, so that each token chooses its k experts among the remaining ones with renormalised weights; nothing is retrained. We report the increase ΔL_T of the mean language-modelling loss on sample B, and compare it with 20 random groups drawn from the experts outside $T(x)$ and matched to its traffic share (fixed seeds). Every masked set must leave at least k experts in each layer. We use $x = 10\%$ (Table 12).

At the 10% tier, across all 16 combinations of the eight models and the two token budgets, the loss increase from masking the least-used group never falls below the range of the random groups: rarely used experts are not dispensable. On the least flattened models at 20B tokens the least-used group

Table 9: The $(8, 8, L)$ series, 20B tokens, tied attention. MMR: model-level routing confidence of the recurrent core (geometric mean). Gain: decrease of the final-window loss relative to the non-looped $(8, 8, 1)$ (nat, ± 1 standard error); per pass: gain divided by the $L - 1$ added passes.

L	1	2	3	4	8
MMR	4.06	4.15	4.02	3.73	3.36
Gain	–	0.0640 ± 0.0013	0.1022 ± 0.0013	0.1237 ± 0.0014	0.1564 ± 0.0014
Per pass	–	0.0640	0.0511	0.0412	0.0223

Table 10: Routing metrics of the recurrent core for all small-tier models at 20B tokens (step 50,000). Class 0: distinct real experts used per token, its upper bound $D \min(E, kL)$ and the ratio of the two; class 1: model-level B_2; class 2: MMR. §Repeated from an earlier group. Values depend on the pool size E; compare absolute values only at equal shape (Section 2.3).

Model	(E, D, L)	attention	distinct experts	upper bound	ratio	B_2	MMR
Flattening, tied attention							
Base-tied	$(8, 8, 2)$	tied	23.47	32	0.73	0.921	4.15
proto-Foil-3	$(16, 4, 4)$	tied	15.67	32	0.49	0.849	5.74
proto-Foil-2	$(32, 2, 8)$	tied	11.61	32	0.36	0.807	7.46
proto-Foil-1	$(64, 1, 16)$	tied	9.42	32	0.29	0.731	8.39
Flattening, untied attention							
Base	$(8, 8, 2)$	untied	24.08	32	0.75	0.947	4.21
Foil-3	$(16, 4, 4)$	untied	17.79	32	0.56	0.898	6.14
Foil-2	$(32, 2, 8)$	untied	12.48	32	0.39	0.839	8.50
Foil-1	$(64, 1, 16)$	untied	9.71	32	0.30	0.805	8.77
Loop passes, $(E, D) = (8, 8)$							
–	$(8, 8, 1)$	tied	16.00	16	–	0.952	4.06
Base-tied§	$(8, 8, 2)$	tied	23.47	32	0.73	0.921	4.15
–	$(8, 8, 3)$	tied	26.34	48	0.55	0.912	4.02
–	$(8, 8, 4)$	tied	28.44	64	0.44	0.915	3.73
–	$(8, 8, 8)$	tied	36.23	64	0.57	0.908	3.36
Loop passes, $(E, D) = (16, 4)$							
–	$(16, 4, 2)$	tied	12.22	16	0.76	0.897	6.08
proto-Foil-3§	$(16, 4, 4)$	tied	15.67	32	0.49	0.849	5.74
–	$(16, 4, 6)$	tied	18.66	48	0.39	0.830	5.34
–	$(16, 4, 8)$	tied	21.21	64	0.33	0.838	5.26
Loop passes, $(E, D) = (8, 4)$							
–	$(8, 4, 2)$	tied	11.53	16	0.72	0.941	4.19
–	$(8, 4, 4)$	tied	13.57	32	0.42	0.932	3.80
–	$(8, 4, 8)$	tied	16.99	32	0.53	0.904	3.53
Pool size and depth at fixed $E_{\text{comp}} = 32$							
–	$(4, 8, 2)$	tied	20.73	32	0.65	0.969	2.06
Base-tied§	$(8, 8, 2)$	tied	23.47	32	0.73	0.921	4.15
–	$(16, 8, 2)$	tied	24.76	32	0.77	0.903	6.07
–§	$(8, 4, 4)$	tied	13.57	32	0.42	0.932	3.80
–	$(8, 16, 1)$	tied	32.00	32	–	0.926	3.84
Other controls							
–	$(4, 8, 4)$	tied	23.27	32	0.73	0.963	1.88
Base-tied, two-layer prelude/coda	$(8, 8, 2)$	tied	22.54	32	0.70	0.912	4.26
Base-tied, seed 43	$(8, 8, 2)$	tied	23.30	32	0.73	0.923	4.34

costs more than almost every random group (19 or 20 of 20 for Base-tied, Base and Foil-3; Tables 12 and 13 and Figure 6). Low load therefore does not indicate low value, and an unbalanced load is not by itself a sign of an unhealthy router.

J LIMITATIONS

Four limitations bound our conclusions. First, we have no non-looped control at equal parameters and compute: the non-looped model with the expert parameters and expert calls of $(8, 8, 2)$ is $(4, 16, 1)$, which raises the active share k/E to $1/2$ and so confounds sharing experts across layers

Table 11: Values of related-work routing metrics for every base-width ($d_{\text{model}} = 1024$) model at the end of 20B-token training (step 50,000). Loop block only. Load metrics: B_H = normalised load entropy $H(q)/\log E$; MaxVio $= E \max_i q_i - 1$; G^* = Gini coefficient normalised by $E - 1$; each is computed per physical layer from selection counts pooled over all passes, then averaged arithmetically over physical layers. Confidence metrics (per token, then averaged over tokens): rank-1 mean probability; selected-set mass M_k (sum of the k selected full-pool probabilities); boundary margin (logit of the k-th minus the $(k{+}1)$-th expert); $C_{\text{full}} = 1 - H(p)/\log E$; C_{sel} = one minus the normalised entropy of the renormalised weights of the selected experts; each is averaged with equal weight over all (physical layer, pass) cells of the loop block (arithmetic mean). Selections are recomputed as the top-k of the stored float16 router logits. Token drop rate is 0 for every model and not applicable (no capacity limit, no dropping), so it is not listed. Probe set: 255,500 tokens from 500 documents, 50 from each of ten domains of the Pile. Model names and footnotes as in Table 3.

			Load			Confidence				
Model	Run id	(E, D, L)	B_H	MaxVio	G^*	Rank-1 prob.	Selected mass M_k	Boundary margin	C_{full}	C_{sel}
Foil family										
Base-tied	S1	(8,8,2)	0.979	0.523	0.162	0.382	0.556	0.417	0.164	0.130
proto-Foil-3	S2	(16,4,4)	0.971	1.099	0.221	0.269	0.403	0.374	0.143	0.103
proto-Foil-2	S3	(32,2,8)	0.972	1.812	0.236	0.173	0.274	0.295	0.130	0.069
proto-Foil-1	S4	(64,1,16)	0.968	3.449	0.270	0.095	0.159	0.198	0.110	0.040
Base	U1	(8,8,2)	0.987	0.451	0.138	0.386	0.559	0.409	0.168	0.133
Foil-3	U2	(16,4,4)	0.982	0.964	0.176	0.286	0.411	0.359	0.150	0.133
Foil-2	U3	(32,2,8)	0.977	1.479	0.216	0.193	0.294	0.330	0.137	0.092
Foil-1	U4	(64,1,16)	0.975	1.291	0.249	0.106	0.171	0.229	0.106	0.055
Other grid models										
–	S6[c]	(8,8,1)	0.987	0.337	0.126	0.381	0.561	0.437	0.161	0.126
–	S8	(8,8,3)	0.977	0.589	0.172	0.375	0.553	0.406	0.159	0.121
–	S9	(8,8,4)	0.979	0.644	0.178	0.361	0.543	0.391	0.148	0.106
–	N4	(8,8,8)	0.977	0.651	0.191	0.343	0.531	0.363	0.140	0.086
–	S5	(16,4,2)	0.980	0.759	0.178	0.288	0.412	0.396	0.149	0.139
–	N3	(16,4,6)	0.968	1.366	0.229	0.253	0.390	0.336	0.138	0.084
–	N2	(16,4,8)	0.971	1.339	0.214	0.250	0.389	0.315	0.141	0.078
–	S7	(8,4,2)	0.985	0.409	0.141	0.383	0.564	0.452	0.164	0.123
–	S12	(8,4,4)	0.983	0.517	0.156	0.365	0.551	0.413	0.151	0.103
–	N1	(8,4,8)	0.977	0.712	0.186	0.352	0.552	0.405	0.158	0.078
–	S11[a]	(4,8,2)	0.989	0.253	0.120	0.493	0.739	0.502	0.153	0.116
–	S13[a]	(4,8,4)	0.985	0.255	0.126	0.473	0.728	0.488	0.133	0.094
–	S10	(16,8,2)	0.981	0.779	0.175	0.288	0.410	0.355	0.152	0.141
–	S14[c]	(8,16,1)	0.981	0.489	0.160	0.368	0.549	0.411	0.151	0.113
–	S15[b]	(8,8,2)	0.977	0.583	0.172	0.388	0.564	0.418	0.170	0.132

with the sparsity of activation. Our question is how the experts should be arranged within the looped block once looping has been chosen; that sharing experts across layers is itself useful is supported by external evidence (Qiu et al., 2026; Jaggi, 2026). Second, apart from one seed replicate, every model is trained with a single seed; the replicate differs by -0.0003 ± 0.0009 nat, which sets the resolution of our comparisons, every difference interpreted in the main text is at least 0.002 nat, and the loss standard errors are optimistic because consecutive steps are not independent (Appendix C). Third, all results are at a single width, $d_{\text{model}} = 1024$. Fourth, the ablations use shared attention, under which the gain from flattening vanishes after the first step (Figure 4, a1–a2); untying the attention is what lets the gain continue, so the ablation trends may differ in size for Foil itself.

Table 12: Masking the least-used experts at the 10% traffic tier. ΔL_T: loss increase (nat) when the least-used group is masked; random: loss increase for 20 traffic-matched random groups (mean and range); last column: number of the 20 random groups whose loss increase is below ΔL_T.

Model	(E, D, L)	experts masked	ΔL_T	random mean	random range	above random
20B tokens						
Base-tied	(8,8,2)	11	0.160	0.104	[0.068, 0.165]	19/20
proto-Foil-3	(16,4,4)	11	0.173	0.137	[0.047, 0.229]	15/20
proto-Foil-2	(32,2,8)	11	0.221	0.231	[0.129, 0.386]	9/20
proto-Foil-1	(64,1,16)	12	0.309	0.271	[0.140, 0.464]	15/20
Base	(8,8,2)	9	0.141	0.087	[0.044, 0.143]	19/20
Foil-3	(16,4,4)	10	0.185	0.109	[0.066, 0.141]	20/20
Foil-2	(32,2,8)	10	0.162	0.151	[0.087, 0.235]	12/20
Foil-1	(64,1,16)	10	0.298	0.287	[0.099, 0.779]	15/20
100B tokens						
Base-tied	(8,8,2)	10	0.109	0.121	[0.048, 0.461]	15/20
proto-Foil-3	(16,4,4)	11	0.174	0.201	[0.092, 0.508]	9/20
proto-Foil-2	(32,2,8)	11	0.231	0.239	[0.091, 0.426]	11/20
proto-Foil-1	(64,1,16)	12	0.232	0.189	[0.113, 0.390]	16/20
Base	(8,8,2)	9	0.134	0.113	[0.061, 0.177]	12/20
Foil-3	(16,4,4)	10	0.131	0.196	[0.069, 0.693]	13/20
Foil-2	(32,2,8)	11	0.136	0.236	[0.096, 2.035]	11/20
Foil-1	(64,1,16)	11	0.235	0.269	[0.078, 1.223]	16/20

Table 13: Least-used group versus random groups at the 10% traffic tier, 20B tokens. Ratio: loss increase from masking the least-used group divided by the mean loss increase of the 20 traffic-matched random groups; last column: number of random groups whose loss increase is below that of the least-used group. Seven of the eight ratios exceed one; none of the least-used groups falls below the range of the random groups.

Model	(E, D, L)	attention	ratio	random groups below
Base-tied	$(8, 8, 2)$	tied	1.54	19/20
proto-Foil-3	$(16, 4, 4)$	tied	1.26	15/20
proto-Foil-2	$(32, 2, 8)$	tied	0.96	9/20
proto-Foil-1	$(64, 1, 16)$	tied	1.14	15/20
Base	$(8, 8, 2)$	untied	1.61	19/20
Foil-3	$(16, 4, 4)$	untied	1.69	20/20
Foil-2	$(32, 2, 8)$	untied	1.07	12/20
Foil-1	$(64, 1, 16)$	untied	1.04	15/20

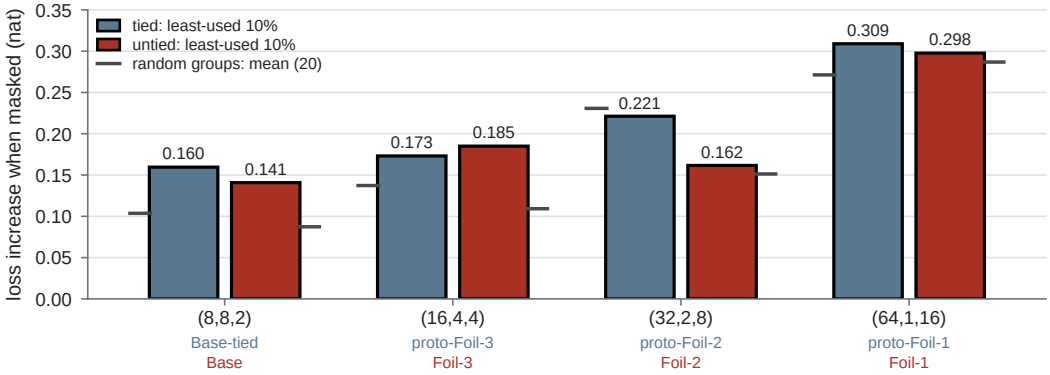

Figure 6: Masking the least-used experts at the 10% traffic tier, 20B tokens. For each shape, the left bar is the tied-attention model and the right bar the untied-attention model; bar height is the loss increase from masking the least-used group and the short dark dash beside it the mean loss increase of the 20 traffic-matched random groups. The ranges of the random groups are given in Table 12, the ratios to the random mean in Table 13.