DESIGN AND TRAINING SHAPE COMPONENT RELIANCE IN SMALL TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Architectural choices can change which components a Transformer learns to rely on. We compare dense and mixture-of-experts (MoE) feedforward networks (FFNs) on three algorithmic tasks and character-level language modeling. MoE models retain higher accuracy after FFN removal on carry-based addition and TinyStories. This advantage persists in deeper models on both tasks and in wider TinyStories models. Controlled one-layer addition experiments show that increasing expert count at fixed total FFN width improves no-FFN performance and increases relative attention attribution, while intact accuracy remains near perfect. We identify restricted active capacity and evolving expert assignments as contributing factors. Critically, allowing assignments to evolve increases no-FFN performance even when router weights remain frozen. Dropout and shuffled assignments further show that training conditions can change reliance. The effect varies across tasks and training stages: the gap grows with continued training in one-layer modular addition, while histogram counting shows little separation. Fitted-readout comparisons distinguish recoverable answer information from its use by the trained model, helping interpret these differences. Together, these findings suggest ways to steer component reliance through FFN design and training.

1 INTRODUCTION

Designing an efficient Transformer requires deciding how to allocate capacity between attention and feedforward networks (FFNs). These components learn together, so changing one may alter what the other learns to support. Understanding this interaction could help guide their joint design—for example, when combining sparse FFNs with memory-limited attention. We study a basic question behind this possibility: how does changing the FFN reshape the surrounding model's learned solution?

Prior work identifies counting strategies that divide work differently between attention and FFNs (Behrens et al., 2025) and studies their coupled learning dynamics (Tian et al., 2024). Evidence of self-repair, where later components compensate for ablations, further highlights the need to study components together (McGrath et al., 2023; Rushing & Nanda, 2024). We investigate how this interaction changes with FFN design. Our main comparison replaces a dense FFN with a sparse mixture-of-experts (MoE) layer while leaving the attention architecture unchanged. Dividing a fixed total FFN hidden width among experts restricts the width available to each token, allowing us to test how the surrounding model adapts.

Our investigation is organized around three questions:

1. Does sparse FFN partitioning change learned component reliance, and does the change persist beyond one layer?
2. Which properties of the FFN pathway contribute: its active capacity, its expert assignments, or the conditions under which it learns?
3. How do task and training stage affect the observed reliance pattern?

We use four tasks for different purposes. Add-7 lets us test specific FFN changes under controlled conditions. Character-level TinyStories asks whether the pattern also appears in language modeling. Modular addition and histogram counting show how it varies across tasks and training stages. We

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.

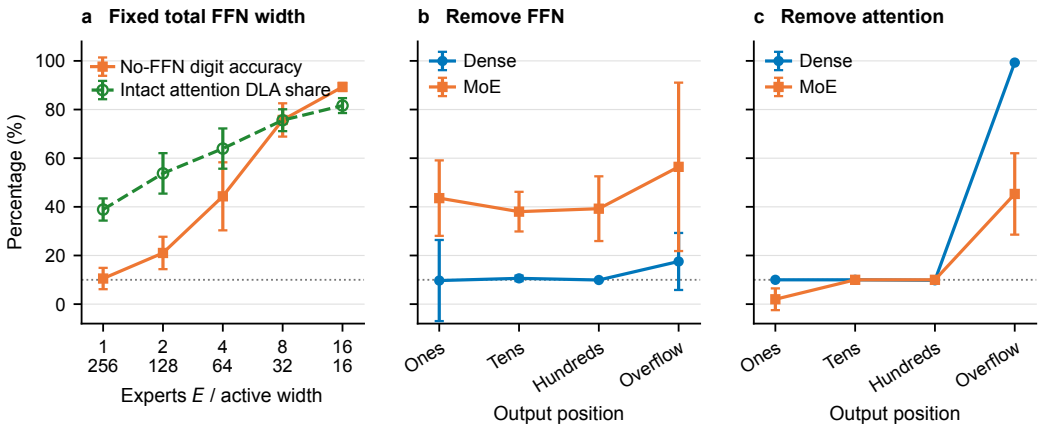


Figure 1: **Add-7: ablated and intact views show the same architectural trend. (a)** At fixed total expert hidden width, both no-FFN digit accuracy and intact attention DLA share rise with expert count; intact accuracy stays near 100%. The two curves are different metrics on a percentage axis: DLA is the mean attention share of absolute attention/FFN correct-logit contributions, excluding the embedding/positional residual. **(b–c)** Position-resolved ablations for the main dense/four-expert comparison. The no-FFN advantage spans positions; the reverse no-attention difference is mainly at overflow. Values are mean $\pm$ sample s.d. over five seeds; $E = 4$ reuses the main MoE runs. The 10% dotted line is a uniform-digit accuracy reference, not a DLA baseline.

measure performance after component interventions, with additional attribution and attention-head tests on add-7.

On add-7, increasing expert count at fixed total FFN width raises performance after FFN removal while intact accuracy stays near perfect. On TinyStories, MoE remains more accurate without its FFNs, and its prediction loss increases less. Both tasks show a no-FFN advantage through six layers, and the TinyStories advantage also persists at greater width (Section 3). The FFN is absent in each removal test: its training-time design changes what the remaining model can do.

We then use add-7 to test which FFN properties matter: active capacity, expert assignments, and variation during training (Section 4). Finally, we examine the different patterns in modular addition and histogram, and ask what a newly fitted output head can recover after FFN removal (Section 5). These comparisons distinguish information present in a representation from the original model's ability to use it.

2 Experimental Setup

We compare Transformers that differ in FFN design and measure how that choice affects the solution learned by the full model. We use one-layer models for controlled comparisons and deeper models to test whether the resulting patterns persist.

Architectures and matching. Each Transformer block applies attention followed by an FFN, with a residual connection around each component. The dense FFN has hidden width $4d_{\text{model}}$. Our primary MoE replaces it with E experts whose hidden widths sum to the dense width. Under top-1 routing, each token accesses one expert: $F(x) = F_{e^*(x)}(x)$, where $e^*(x) = \arg\max_e (W_r x)_e$. This approximately matches total FFN parameters while reducing active hidden width per token by a factor of E. The main comparison uses four experts; the expert-count sweep uses $E \in \{1, 2, 4, 8, 16\}$.

The selected top-1 gate is renormalized to one, so the task loss has no differentiable path to router weights W_r. The auxiliary balancing loss updates these weights and can also train the upstream representations. Our paired replay studies omit this loss. Gated top-1 and weighted top-2 permit router task gradients. Primary models use four attention heads; GLU, MoE-GLU, and protocol details are in the appendix.

Tasks and model scale. We study add-7, modular addition, histogram counting, and character-level TinyStories. Add-7 maps a reversed three-digit input to the four output digits of $x + 7$. Training samples uniformly from the same 1,000 inputs used for exhaustive evaluation. Modular addition predicts $(a + b) \bmod 113$ using a 30/70 train/test split. Histogram counting predicts each token's frequency in a length-10 sequence over 32 symbols. TinyStories models predict the next character using a 92-character vocabulary and context length 128 (Eldan & Li, 2023).

Attention is causal for add-7 and TinyStories, and bidirectional for modular addition and histogram. Primary widths are $d_{\text{model}} = 64$ for add-7 and 128 otherwise. We test up to six layers for add-7 and TinyStories, and three for the other tasks. The primary TinyStories width comparison uses two layers and $d_{\text{model}} \in \{128, 256\}$. Appendix B gives training and checkpoint-selection details.

Intact models and component reliance. An *intact model* is evaluated with its original trained parameters and native inference procedure, without component ablation or activation replacement. We measure *component reliance* operationally through changes in task performance under specified interventions. For a given intervention, let $R_C(A)$ denote accuracy for architecture A after intervening on every instance of component family C. Our primary architecture contrast is

$$\Delta_{\text{FFN}} = R_{\text{FFN}}(\text{MoE}) - R_{\text{FFN}}(\text{Dense}). \tag{1}$$

A positive value means that MoE retains higher performance under the same FFN intervention. We report intact performance alongside this contrast. In multilayer models, "no FFN" removes the FFN output in every layer unless stated otherwise. The remaining pathway includes embeddings, attention, residual connections, and the output head.

Interventions and complementary measurements. Zero ablation removes a component's output immediately before residual addition. Mean replacement substitutes its checkpoint-, position-, and layer-specific average output. Add-7 means are estimated over the exhaustive 1,000-input evaluation set. Graded ablations remove a fraction of output dimensions. These evaluations keep all trained parameters fixed, including the original output head. We complement them with intact-model direct logit attribution, attention-head interventions, and activation patching.

For one-layer readout diagnostics, we freeze the Transformer and fit a new affine output head to its post-attention, pre-FFN residual, using separate fitting, validation, and test groups. This tests what a new head can recover, rather than what the original head predicts after FFN removal.

3 FFN DESIGN CHANGES LEARNED COMPONENT RELIANCE

We first ask whether models trained with different FFNs behave differently when those FFNs are removed. We study this closely on add-7, then test deeper models and character-level language modeling on TinyStories.

3.1 ADD-7: THE SAME TASK, A DIFFERENT LEARNED SOLUTION

On add-7, we hold total expert hidden width at 256 and vary $E \in \{1, 2, 4, 8, 16\}$. More experts mean less FFN width per token. At the accuracy-selected checkpoints, intact accuracy remains near 100%, but accuracy after complete FFN removal rises from 10.5% to 89.3% (Fig. 1a).

The FFN is absent in every evaluation; its training-time design changes what the remaining model can support with its original readout. With sixteen experts, the surviving attention–residual–readout pathway predicts most output digits correctly even though every expert output has been removed.

Does the trend depend on balancing loss or checkpoint timing? In a separate five-seed sweep, we remove balancing, freeze router weights, and evaluate every model after 3,000 updates. Mean no-FFN accuracy again rises with expert count, under both zero and mean replacement, while intact accuracy stays near perfect (Appendix D.1).

The intact model shows a corresponding attribution trend. Attention's relative logit-magnitude contribution rises at every expert-count transition in all five seeds. The trend remains when we compare the correct answer with competing classes and separate attention's output bias (Appendix F.1.1). These measurements describe attribution; component interventions test reliance.

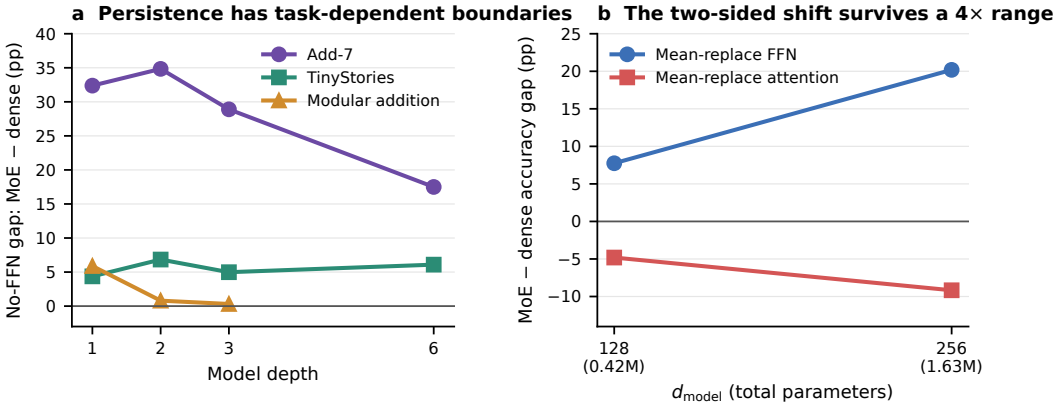


Figure 2: **The no-FFN advantage persists on add-7 and TinyStories. (a)** MoE–dense no-FFN gaps across depth. Add-7 and TinyStories remain positive through six layers, while modular addition falls below one point by depth two among converged checkpoints. **(b)** At two-layer TinyStories width, mean replacement preserves the positive no-FFN and negative no-attention orderings across 0.42M–1.64M parameters. Points show differences of seed means; per-architecture uncertainty and sample sizes are in Appendix E.

In the main dense/four-expert comparison, no-FFN accuracy is 11.9% for dense and 44.3% for MoE. The advantage spans all four output positions (Fig. 1b). The sweep's $E = 1$ models are separately trained single-expert baselines, explaining their slightly different score.

3.2 Beyond one-layer arithmetic

Deeper add-7 models. We train dense and MoE models with one, two, three, and six layers, then remove every FFN. The add-7 advantage remains positive at all four depths (Fig. 2a). At six layers it is 17.5 percentage points across five seeds, with near-perfect intact accuracy in both architectures. Removing a single FFN leaves high accuracy in both; removing the whole family reveals their different reliance.

Language modeling on TinyStories. To test the pattern beyond arithmetic, we train character-level TinyStories models. MoE remains more accurate after FFN removal through six layers. Because MoE and dense have slightly different intact accuracy, we compare loss increases relative to each model's own baseline. At six layers, temperature-calibrated cross-entropy increases by 2.60 nats per character in MoE versus 3.55 in dense after FFN removal. Mean replacement preserves the accuracy ordering as well.

Increasing width gives a further check. In two-layer TinyStories models, doubling d_{model} from 128 to 256 increases total parameters roughly fourfold, from 0.42M to 1.64M. The no-FFN advantage grows rather than disappearing (Fig. 2b). At width 256, mean FFN replacement leaves 25.04% character accuracy in MoE versus 4.85% in dense. MoE's intact accuracy is lower by approximately four points, but its calibrated loss increase is also smaller: 1.53 versus 3.05 nats per character. Thus, the pattern extends beyond one-layer arithmetic in both retained accuracy and baseline-relative loss. Full scores, uncertainty, and the wider fixed-budget diagnostic are in Appendix E.

3.3 Evidence beyond complete FFN removal

Complete component removal changes the model's internal states substantially. We therefore return to one-layer add-7 and ask whether the difference also appears under other interventions, and whether attention matters when the FFN is still present.

The gap survives mean and partial replacement. Replacing each FFN output with its position-specific mean leaves a 28.6-point MoE–dense gap, so preserving this constant contribution does not remove the difference. Partial deletion gives a consistent pattern: removing 25%, 50%, or 75% of

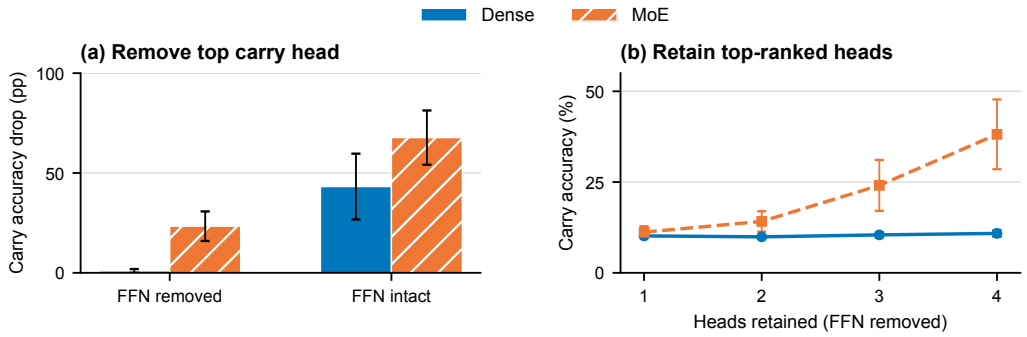


Figure 3: **Attention heads contribute to the learned add-7 solution. (a)** Removing the seed-wise top carry head has a larger effect in MoE both with the FFN absent and with it intact. **(b)** MoE carry accuracy rises as additional heads are restored. Values show mean ± s.d. over five seeds.

FFN-output dimensions favors MoE, whereas removing attention-output dimensions favors dense (Appendix Fig. 5).

Attention contributes with the FFN present. Within each seed, we select the attention head whose removal most reduces carry accuracy without the FFN. Removing that head with the FFN intact reduces carry accuracy by 67.7 points in MoE and 43.2 points in dense (Fig. 3a). Restoring additional heads also improves no-FFN carry accuracy (Fig. 3b).

Holding expert assignments at their intact values still yields a 63.8-point MoE drop. The large effect therefore does not require rerouting, although the selected experts receive altered inputs (Appendix F.5).

Aggregate no-attention accuracy is less informative: dense achieves approximately 10% at each ordinary digit position but 99.3% at overflow, producing its 32.3% average. The overflow score comes from predicting the majority class, zero, while missing all seven true overflow cases. Mean attention replacement nearly erases the reverse dense–MoE ordering, making the targeted head interventions the clearer attention-side evidence.

Together, these add-7 measurements support a change in learned reliance: the gap survives alternative FFN interventions, and selected attention heads affect predictions with the FFN present. We next use this controlled setting to investigate contributing FFN properties.

4 What shapes the learned division of labor?

We now ask which FFN choices contribute to the difference. We use one-layer add-7, where both architectures solve the task nearly perfectly, and vary active width, expert assignments, and the FFN's contribution during training.

4.1 Restricted active capacity contributes

If reduced per-token FFN capacity contributes, narrowing a dense FFN should reproduce part of the effect, while restoring active capacity in MoE should weaken it. Both predictions hold. Reducing dense hidden width from 256 to 64 raises no-FFN accuracy from 11.9% to 23.7%, still below four-expert MoE (Fig. 4a). Activating two experts or adding an always-active shared expert reduces the advantage. Within the MoE-GLU family, widening each expert to the dense per-token width lowers no-FFN accuracy from 41.7% to 18.3% (Fig. 4c).

These interventions point in a common direction: restoring the active FFN pathway weakens the advantage. Top-2 and shared experts also change mixing or connectivity, so they are complementary controls rather than isolated width manipulations. Crucially, narrow dense still falls below MoE. This leaves a question that capacity alone does not answer: does the way tokens reach experts also matter?

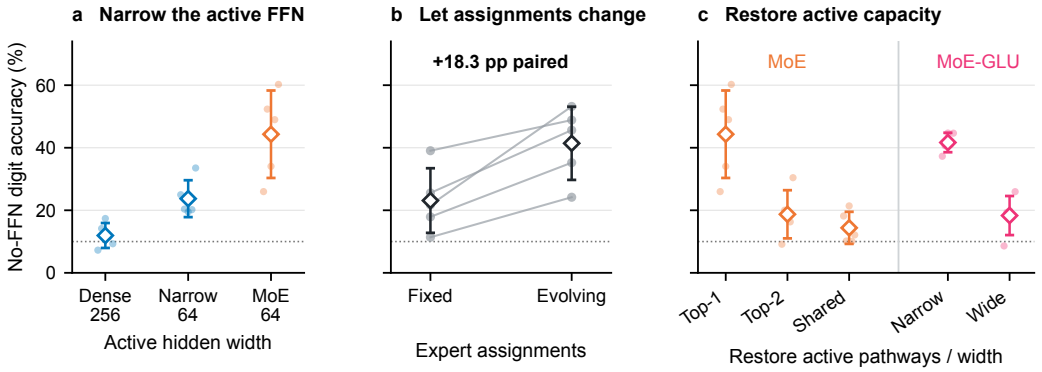


Figure 4: **Active capacity and evolving assignments contribute to learned reliance.** **(a)** Narrowing dense reproduces part of the MoE effect. **(b)** With router weights frozen, evolving top-1 assignments increase no-FFN accuracy in every paired seed at accuracy-selected checkpoints. **(c)** Restoring active pathways weakens the effect; MoE-GLU width restoration is a separate within-family comparison. Dots show five seeds, diamonds means, and bars sample s.d. Only panel (b) joins paired runs. The controls do not form an additive decomposition.

4.2 Evolving assignments contribute beyond fixed routing

Frozen router weights do not imply fixed expert assignments: as embeddings and attention learn, changing representations can send a token to a different expert.

We compare paired hard-top-1 models with identical initial parameters and training batches, frozen router weights, and no auxiliary balancing loss. The replay model keeps the initial expert choice for each example and position throughout training; the dynamic model recomputes assignments as representations change.

We find that dynamic routing gives higher no-FFN accuracy in all five pairs (Fig. 4b). A separate fixed-budget study confirms the direction when both arms are evaluated after exactly the same number of updates, including under mean replacement (Table 1; Appendix D.2.1). Assignment evolution can therefore change learned reliance even when router weights and training time are held fixed. Additional partition and operation-alignment controls are discussed in the appendix (Appendix D.5); the corresponding weighted top-2 comparison remains inconclusive (Appendix D.3).

Separately, gated top-1 weights the selected expert's output by its routing probability, giving the router a task gradient. All final models solve the intact task perfectly. The MoE–dense advantage survives zero and mean FFN replacement in all five seeds at both $E = 4$ and $E = 16$. Mean replacement leaves gaps of 15.4 and 32.5 points, respectively, smaller than in matched hard-top-1 runs (Appendix D.6). Thus the advantage also appears with task-trained routing.

4.3 Training-time FFN variability changes reliance

The replay result suggests looking beyond nominal capacity to how the FFN contributes while the model learns. Could variability in that contribution itself change later reliance? We test this by varying FFN outputs and expert assignments directly.

Varying FFN outputs at fixed hidden width. In a separate exploratory study, we cross dense hidden widths $\{64, 256\}$ with FFN-output dropout probabilities $\{0, 0.3\}$. Paired models share initial parameters and training batches. Dropout acts after the FFN and before residual addition; it is disabled at evaluation. There is no attention dropout or training loss on an FFN-ablated model.

All final models solve add-7 perfectly, yet dropout substantially increases performance after FFN removal at both widths (Table 1). The increase is larger in the narrow model. Mean replacement and

earliest-best intact checkpoints preserve these directions. Thus, changing the FFN's training-time contribution changes later reliance even without experts.

Shuffled replay provides the complementary test within MoE: changing which examples reach each expert raises no-FFN accuracy relative to fixed replay, without representation-dependent reassignment. Together, these interventions motivate studying the consistency of the FFN contribution as a possible link between architecture and reliance. They establish effects of introducing variability, rather than showing that variability mediates the ordinary MoE effect.

Table 1: **Training-time variability changes later component reliance.** Separate 3,000-update add-7 study; mean ± sample s.d. over five seeds. All final intact digit and autoregressive exact scores are 100%. Scores below are teacher-forced digit accuracies after zero or mean FFN replacement. Dropout is disabled during evaluation. The three MoE arms use four width-64 experts and frozen router weights.

Training condition	Zero FFN	Mean FFN
Dense, width 256	10.68 ± 3.46	40.03 ± 0.77
Dense, width 256, dropout	46.11 ± 6.28	64.53 ± 6.22
Dense, width 64	19.89 ± 7.22	52.15 ± 4.55
Dense, width 64, dropout	79.19 ± 10.28	85.55 ± 6.60
MoE, fixed replay	24.99 ± 9.77	52.26 ± 6.57
MoE, shuffled replay	35.32 ± 9.76	63.69 ± 6.08
MoE, dynamic	49.96 ± 8.29	68.53 ± 5.22

4.4 An idealized view of joint learning

A simple model illustrates how learning conditions can favor a shared pathway even when either pathway can represent the target. The shared pathway learns from every example, while each expert learns from only one partition. Let their scalar contributions be a and b_e, predicting the same target $y > 0$ across E equally frequent, fixed partitions. Under zero initialization and equal-rate, full-batch ordinary gradient flow on unregularized squared error, symmetry gives $b_e = b$ and

$$\dot{a} = -(a + b - y), \qquad \dot{b} = -\frac{a + b - y}{E} \quad \Longrightarrow \quad a = Eb, \qquad \frac{a}{y} \longrightarrow \frac{E}{E+1}. \tag{2}$$

The expert's update is smaller because it sees fewer examples. The shared pathway consequently supplies more of the converged prediction as expert count increases. This gives an intuition for the direction of the sweep: the observed DLA log-odds slope is 0.49 (95% CI $[0.43, 0.55]$), below the predicted $\log 2 \approx 0.69$, and fixed partitions do not explain the dynamic–replay difference. Further specifics are in Appendices F.2 and F.3.

These add-7 comparisons establish that active capacity, assignment evolution, and training-time variability can each influence reliance. We next examine modular addition and histogram to understand where the observed pattern changes.

5 When does the effect appear, and what does it measure?

Does every task show the same pattern? We examine modular addition and histogram to see where it changes. We then fit new output heads on add-7 and histogram to ask whether poor performance after FFN removal means answer information is missing.

5.1 Different patterns in modular addition and histogram

In the primary depth grid, modular addition's no-FFN gap falls from 5.9 points at one layer to below one point at two and three layers. Histogram shows little separation at all three tested depths (Appendix E). These are differences after removing every FFN, not a claim that all component roles become identical. The grid also uses different checkpoint-selection conventions across depths.

Training stage matters even when depth is fixed. In a separate one-layer modular-addition study, we compare the first saved checkpoints above 99% test accuracy with checkpoints at epoch 40,000. On

Table 2: **Accessible information and native ablation survival can differ.** Accuracy after FFN removal, averaged over five checkpoint seeds per architecture. Both readouts are evaluated on the same held-out readout-test examples. These examples are held out from fitting the new head, not necessarily from Transformer training. Add-7 reports teacher-forced digit accuracy; histogram reports count accuracy.

	Original head		New affine head	
Task	Dense	MoE	Dense	MoE
Add-7	11.38	43.77	99.83	99.38
Histogram	12.17	11.95	78.25	92.90

the same evaluation examples, the no-FFN gap grows from 6.3 to 18.2 points across three seed pairs, increasing in every pair while intact accuracy remains near perfect. The later gap also persists on inputs held out from Transformer training. Modular addition shows that reliance can keep changing after a task is solved, while histogram shows little gap with the original readout. Solving a task therefore does not determine which components the trained model will depend on.

5.2 Recoverable information differs from learned use

On add-7 and histogram, we compare the original output head with a new affine head fitted to the pre-FFN representation of one-layer models, keeping the Transformer frozen. This asks whether poor no-FFN performance reflects missing answer information or difficulty using it with the original head.

The original head learned to read the representation after the FFN. Removing the FFN changes the head's input, but leaves the pre-FFN representation unchanged for the same input tokens. A new head can therefore reveal information that the original head fails to recover after removal.

On add-7, the new head recovers nearly perfect answers from both architectures despite their large original-head gap (Table 2). Dense therefore retains linearly recoverable answer information. One possible explanation for its poor original-head performance is that the FFN transforms this information into a form suited to that head. In a separate three-seed training study, giving an untied output head a fresh random initialization raises no-FFN accuracy in both architectures and narrows their gap. Untying the head while keeping its initial weights unchanged preserves a large gap. These models are trained from scratch, showing that output-head initialization also shapes learned reliance (Appendix H.4).

Histogram gives the complementary result: the original heads perform similarly poorly without the FFN, yet a new head recovers more accurate counts from MoE representations. During training, fitted decoding improves while original-head no-FFN accuracy stays near 12%. Recoverable information and original-head performance can therefore change differently. The new head learns from answer labels and reads both the embedding residual and attention output. Together, these results show why reliance concerns how the representation, FFN, and readout work together, not simply whether answer information is present.

6 Related work

Mechanistic studies identify structured solutions for modular addition, multi-digit addition, and counting (Nanda et al., 2023; Quirke & Barez, 2023; Behrens et al., 2025; Zhong et al., 2023). Counting strategies can assign different work to attention and FFNs. JoMA studies their coupled learning dynamics (Tian et al., 2024), while Nguyen & Reddy (2025) distinguish available capacity from the rates at which solutions are learned. These studies motivate our question: how does changing one pathway influence what the surrounding model learns?

FFNs have been interpreted as key–value memories that transform residual features (Geva et al., 2021), and factual association studies examine interactions between representations and downstream projections (Meng et al., 2022). Our interventions and fitted-readout comparisons likewise distinguish a component's contribution within the trained model from information that can be recovered with another decoder. Work with frozen random attention shows that surrounding components can adapt

to fixed transformations (Dong et al., 2025); route replay tests the related distinction between fixed router weights and changing expert assignments.

Studies of the Hydra effect and self-repair show that downstream components can partly compensate for ablations (McGrath et al., 2023; Rushing & Nanda, 2024). The resulting logit change can therefore be smaller than the component's direct contribution. Our primary one-layer FFN-removal tests instead measure an already-learned pathway: with fixed input tokens, attention precedes the FFN and is unchanged by its removal. In deeper models, downstream compensation may also contribute.

MoE research studies scaling, efficiency, load balance, and specialization (Shazeer et al., 2017; Fedus et al., 2022; Jiang et al., 2024; Dai et al., 2024). Routing may respond to context (Arnold et al., 2024), learn jointly with experts through task gradients (Ahrac et al., 2026), or activate experts at random (Zuo et al., 2021; Chen et al., 2023). Our focus is the effect on reliance outside the FFN. The primary hard-top-1 setup separates router-weight updates from assignment changes; its active-width controls examine the bottleneck rather than system efficiency, since equal active FLOPs need not imply equal communication costs (Du et al., 2024).

7 Discussion and conclusion

Changing the FFN changes how the trained model uses its components. We see this in both add-7 and character-level TinyStories, including six-layer models and wider TinyStories models. The add-7 controls show that active capacity, changing expert assignments, and training conditions each matter. The effect varies across tasks: the modular-addition gap grows with continued training, whereas histogram shows little separation. The readout tests show that poor performance after FFN removal need not mean answer information is missing.

Implications for model design. Expert count, top-k, shared experts, and active width may offer controls over learned reliance alongside their familiar effects on parameter count and inference cost. A concrete prediction for larger models is that widening an always-active FFN pathway will weaken the no-FFN advantage under comparable training conditions. Testing this would determine whether the controls studied here can inform larger-model design.

This also motivates studying attention memory and FFN design together. Recurrent linear attention compresses history into a fixed-size state (Katharopoulos et al., 2020), and much of its design concerns how that memory retains and retrieves information. If the FFN changes what attention learns to support, it may also change what such a state needs to retain. We believe this is a fruitful and testable design question.

Implications for interpretability. Component roles should be reassessed after architectural or training changes, even when task accuracy stays similar. A fitted decoder can reveal recoverable information, while interventions test how the original model uses its components. Treating these as complementary questions helps distinguish information present in a representation from the solution actually learned.

Limitations. We study small models and evaluate add-7 on its finite training support; persistence through six layers does not establish transfer to large pretrained models. The FFN-removal advantage survives gated top-1, but the paired assignment effect is established only for renormalized hard top-1. Gated top-1 changes output scale as well as router gradients; top-2 also changes active width and expert mixing.

Ablations alter internal states, while fitted readouts change the decoder; neither alone identifies a complete circuit. Variability interventions change reliance but do not establish that variability mediates the ordinary MoE effect. Cross-task differences also mix task, architecture, and training effects.

References

Sagi Ahrac, Noya Hochwald, and Mor Geva. Routers learn the geometry of their experts: Geometric coupling in sparse mixture-of-experts, 2026.

Stefan Arnold, Marian Fietta, and Dilara Yesilbas. Routing in sparsely-gated language models responds to context, 2024.

Freya Behrens, Luca Biggio, and Lenka Zdeborová. Counting in small transformers: The delicate interplay between attention and feed-forward layers. In *International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 3500–3532, 2025.

Tianlong Chen, Zhenyu Zhang, Ajay Jaiswal, Shiwei Liu, and Zhangyang Wang. Sparse MoE as the new dropout: Scaling dense and self-slimmable transformers, 2023.

Damai Dai, Chengqi Deng, Chenggang Zhao, R. X. Xu, Huazuo Gao, Deli Chen, Jiashi Li, et al. DeepSeekMoE: Towards ultimate expert specialization in mixture-of-experts language models, 2024.

Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *International Conference on Machine Learning*, pp. 933–941, 2017.

Yihe Dong, Lorenzo Noci, Mikhail Khodak, and Mufan Li. Is random attention sufficient for sequence modeling? disentangling trainable components in the transformer, 2025.

Xianzhi Du, Tom Gunter, Xiang Kong, Mark Lee, Zirui Wang, Aonan Zhang, Nan Du, and Ruoming Pang. Revisiting MoE and dense speed-accuracy comparisons for LLM training, 2024.

Ronen Eldan and Yuanzhi Li. Tinystories: How small can language models be and still speak coherent english?, 2023.

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.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, 2021. doi: 10.18653/v1/2021.emnlp-main.446.

Hakan Inan, Khashayar Khosravi, and Richard Socher. Tying word vectors and word classifiers: A loss framework for language modeling. In *International Conference on Learning Representations*, 2017.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, et al. Mixtral of experts, 2024.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5156–5165. PMLR, 2020. URL https://proceedings.mlr.press/v119/katharopoulos20a.html.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. URL https://arxiv.org/abs/1412.6980.

Thomas McGrath, Matthew Rahtz, János Kramár, Vladimir Mikulik, and Shane Legg. The Hydra effect: Emergent self-repair in language model computations, 2023. URL https://arxiv.org/abs/2307.15771.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, volume 35, pp. 17359–17372, 2022.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *International Conference on Learning Representations*, 2023.

Alex Nguyen and Gautam Reddy. Differential learning kinetics govern the transition from memorization to generalization during in-context learning. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/187f9ba4cd5a59477305ac712282a1cd-Paper-Conference.pdf.

Ofir Press and Lior Wolf. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 157–163, 2017.

Philip Quirke and Fazl Barez. Understanding addition in transformers, 2023.

Cody Rushing and Neel Nanda. Explorations of self-repair in language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42836–42855. PMLR, 2024. URL https://proceedings.mlr.press/v235/rushing24a.html.

Noam Shazeer. GLU variants improve transformer, 2020.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

Yuandong Tian, Yiping Wang, Zhenyu Zhang, Beidi Chen, and Simon Du. JoMA: Demystifying multilayer transformers via joint dynamics of MLP and attention. In *International Conference on Learning Representations*, 2024.

Ziqian Zhong, Ziming Liu, Max Tegmark, and Jacob Andreas. The clock and the pizza: Two stories in mechanistic explanation of neural networks. In *Advances in Neural Information Processing Systems*, volume 36, pp. 27223–27250, 2023.

Simiao Zuo, Xiaodong Liu, Jian Jiao, Young Jin Kim, Hany Hassan, Ruofei Zhang, Tuo Zhao, and Jianfeng Gao. Taming sparsely activated transformer with stochastic experts, 2021.

REPRODUCIBILITY STATEMENT

Appendix B reports architectures, matching conventions, training budgets, seeds, and intervention procedures. The supplementary material includes code, checkpoints, and saved per-seed result summaries for every reported experiment.

AI USE STATEMENT

Generative AI tools assisted with language editing, some research code, and consistency checks. The authors chose the research questions and design, inspected generated code, verified any number reported by the models used, adjudicated the interpretation, and take responsibility for all claims and text.

A APPENDIX GUIDE AND CLAIM–EVIDENCE MAP

The main paper contains the evidence needed to evaluate the central claim; this appendix provides complete protocols, uncertainty, and secondary controls. Table 3 maps each claim to the detailed record.

Table 3: Claim–evidence map.

Claim	Main evidence	Appendix record
Changing the FFN changes learned component reliance	No-FFN and intact-DLA sweeps; mean replacement and fixed-route head tests	Apps. C, F
The tested expert-count configurations give a dose response	Primary sweep and no-balancing, equal-update replication	Apps. D, D.1
In hard top-1, evolving assignments increase no-FFN survival with frozen router weights	Paired replay and its fixed-budget replication	Apps. D.2, D.2.1
The FFN-removal advantage survives gated top-1 with router task gradients	Matched hard/gated comparison at $E = 4, 16$	App. D.6
The effect persists under modest depth/width increases but is not universal	Add-7, modular addition, histogram, TinyStories	App. E
Restoring per-token capacity attenuates the effect	Total-parameter vs. active-width matching	App. G
Learned FFN transformations differ across task/model settings	Frozen-FFN replacement with shared and query-conditioned affine maps	App. H

B TASKS, MODELS, AND TRAINING

B.1 TASK DEFINITIONS AND EVALUATION

Add-7. Given a three-digit number $x \in \{000, \ldots, 999\}$, the task produces $x + 7$ digit by digit. Digits are represented in reversed order (ones first), followed by EOS; loss is applied only to output tokens. For example, $593 + 7 = 600$ maps $[3, 9, 5, \text{EOS}]$ to $[0, 0, 6, 0, \text{EOS}]$. Each output token uses one of three operations: $+7$ at the ones position, $+1$ when carry arrives, and $+0$ otherwise. The exhaustive evaluation set contains 1,000 examples and 4,000 output tokens: 1,000 $+7$, 777 $+1$, and 2,223 $+0$. Training draws online uniform samples from this same finite support. We report digit accuracy; exact match is used as a supplementary solvability check. Carry-chain strata contain 300, 630, 63, and 7 examples for lengths $L = 0, 1, 2, 3$, so the final stratum is descriptive only.

Modular addition. For $a, b \in \mathbb{Z}_{113}$, the model predicts $(a + b) \bmod 113$ from $[a, b, =]$. We use a fixed 30/70 train/test split of all 113^2 pairs and apply loss at the answer position. Depth summaries evaluate all input pairs and include checkpoints with at least 98% intact accuracy on this full support, rather than on the held-out split alone; tables state the surviving n.

Histogram counting. Given a length-10 sequence over 32 symbols, the model predicts each token's frequency in the sequence. Backward sampling balances partition structures following Behrens et al. (2025). Attention is bidirectional. We use 10,000 training and 3,000 test sequences and report per-position count accuracy.

TinyStories. We use the first eight million characters of TinyStories validation text (Eldan & Li, 2023), split contiguously into 90% train and 10% validation segments. Models predict the next character at each position of a length-128 context over a 92-character vocabulary. Evaluation uses fixed validation batches shared by every architecture. We report top-1 character accuracy and temperature-calibrated cross-entropy.

B.2 ARCHITECTURE AND MATCHING

Input embeddings include learned positional embeddings. For attention head h,

$$\text{Attn}_h(X) = \text{softmax}(Q_h K_h^\top / \sqrt{d_k} + M)V_h, \quad \text{MHA}(X) = [\text{Attn}_1; \ldots; \text{Attn}_H]W_O.$$

For add-7 and TinyStories, $M_{ij} = 0$ when $j \le i$ and $-\infty$ otherwise: a position cannot attend to future targets during teacher forcing. Modular addition and histogram use $M = 0$ (bidirectional attention). Modular-addition inputs contain only $[a, b, =]$, not the answer. Every block applies attention and

then an FFN, with a residual connection around each. Add-7 and TinyStories tie input and output embeddings (Press & Wolf, 2017; Inan et al., 2017); modular addition and histogram use untied output heads.

The dense FFN is $W_{\text{down}}\sigma(W_{\text{up}}x)$. GLU uses $W_{\text{down}}(\sigma(W_{\text{gate}}x) \odot W_{\text{up}}x)$ (Dauphin et al., 2017; Shazeer, 2020). The top-1 MoE selects $e^*(x) = \arg\max_e(W_r x)_e$ and applies expert $F_{e^*(x)}$. After top-1 selection, its gate is divided by the sum of selected gates and therefore equals one. The task loss consequently has no differentiable path to W_r in this implementation; when W_r is unfrozen, it is updated only through the auxiliary load-balancing objective. The main MoE grid uses four experts and balancing coefficient 0.01. This implementation fact limits our boundary-update claim to hard top-1 routing and does not apply automatically to weighted top-k MoEs. GLU and MoE-GLU omit linear biases; dense FFN and plain MoE include them. Primary runs omit RMSNorm; the norm control is reported in Appendix G.

Where the balancing loss sends gradients. For top-1 routing over N batch positions, let $p_{ie} = \text{softmax}(W_r h_i)_e$, $f_e = N^{-1}\sum_i \mathbf{1}[e^*(h_i) = e]$, and $P_e = N^{-1}\sum_i p_{ie}$. Training adds $\lambda E \sum_e f_e P_e$, with $\lambda = 0.01$ in the primary sweep. The hard counts f_e have no gradient, but P_e is differentiable through both W_r and its input h_i. The input is not detached, so this objective can update embeddings and attention as well as the router. Counts and probabilities include all input sequence positions, not only loss-scored output digits. At $E = 1$ the objective is constant. Freezing W_r alone does not remove upstream auxiliary gradients; the paired dynamic/replay studies set $\lambda = 0$ in both arms. The ordinary sweep thus tests active width, routing, and balancing together, rather than an isolated partitioning effect.

Table 4: FFN widths and parameter counts for the primary matched grid. MoE entries give per-expert width. Total FFN parameters match dense within 1–2%, while top-1 active width is one quarter of dense.

Task (d_{model})	Dense FFN	GLU	MoE	MoE-GLU
Add-7 (64)	$h = 256$; 33,088	$h = 170$; 32,640	$h_E = 64$; 33,536	$h_E = 42$; 32,512
Other grid (128)	$h = 512$; 131,712	$h = 340$; 130,560	$h_E = 128$; 132,608	$h_E = 85$; 131,072

Table 5: Training hyperparameters. Algorithmic tasks use seeds $\{42, 137, 256, 512, 1024\}$; TinyStories uses $\{42, 137, 256\}$.

	Add-7	Modular addition	Histogram	TinyStories
Optimizer	AdamW	AdamW	AdamW	AdamW
Learning rate	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Weight decay	0.01	1.0	0.1	0.01
Batch size	128 online	full batch	full batch	64
Duration	10k steps	40k epochs	1k epochs	8k steps
Train/test	online/exhaustive	30%/70%	10k/3k	90%/10%

Primary add-7 checkpoint selection. The primary add-7 grid and expert sweep allow up to 10,000 updates. Every 200 updates, the training script evaluates greedy autoregressive exact match, including final EOS, on 256 newly sampled inputs from the finite support. A checkpoint replaces the saved state only on a strict accuracy improvement; ties keep the earlier state. The patience limit is 50 evaluations without improvement. The reported ablations use the saved best-accuracy state, not the last training update, and evaluate all 1,000 inputs. Table 12 records the actual sweep endpoints. Because those endpoints differ, this protocol compares accuracy-selected solutions, not equal training exposure. No reliance measurement is used for selection.

B.3 Interventions, metrics, and statistics

Zero ablation replaces a component output with zero before residual addition. Mean replacement estimates a separate mean at each position and layer from intact-model activations, then substitutes that

constant for the component output. Partial ablation zeros a uniformly sampled $f \in \{0.25, 0.50, 0.75\}$ fraction of output dimensions; eight masks are averaged within checkpoint. In multi-layer models, "no FFN" or "no attention" removes every block of that family unless a layer is named explicitly. Unless stated otherwise, accuracy summaries report mean $\pm$ population standard deviation over seeds. Figures 1, 4, 7, and 8, and Tables 18, 30, and 26 use sample s.d. computed from individual seed results. Confidence intervals use sample variability. Route-replay contrasts are paired by seed and use two-sided t-based inference. No single p value is treated as the main evidence.

For add-7, means are estimated separately for each checkpoint and component from teacher-forced sequences covering all 1,000 three-digit inputs, then evaluated on that same exhaustive set. At each position, every example receives the same mean vector; no model parameters are updated during estimation. This is an exhaustive finite-task intervention analysis, not an independently calibrated held-out evaluation.

For TinyStories calibrated CE, one positive temperature is fitted separately for each checkpoint and intervention on held-out training batches, then evaluated on fixed validation batches. Temperature scaling changes neither predictions nor top-1 accuracy. Component means, temperature fitting, and validation use distinct generator seeds.

C Complete primary add-7 evidence

Table 6: One-layer zero ablations on add-7 (digit accuracy percentage; mean $\pm$ s.d., five seeds).

Architecture	Intact	No attention	No FFN
Dense FFN	100.00	32.27 ± 0.10	11.94 ± 3.57
GLU	99.95 ± 0.06	30.72 ± 2.95	9.70 ± 2.42
MoE	100.00	16.83 ± 4.30	44.32 ± 12.51
MoE-GLU	99.95 ± 0.06	21.82 ± 5.34	41.68 ± 2.78

Table 7: Add-7 FFN-output interventions (digit accuracy percentage; mean $\pm$ s.d., five seeds). Partial rows average eight masks per checkpoint before aggregating seeds.

Architecture	Zero	Mean	25% zeroed	50% zeroed	75% zeroed
Dense	11.94 ± 3.57	37.38 ± 4.81	74.58 ± 3.50	36.59 ± 7.29	19.22 ± 5.34
GLU	9.70 ± 2.42	48.25 ± 2.54	77.54 ± 3.39	45.89 ± 4.51	22.57 ± 2.17
MoE	44.32 ± 12.51	65.96 ± 4.82	93.47 ± 2.17	78.51 ± 4.61	60.32 ± 8.49
MoE-GLU	41.68 ± 2.78	57.35 ± 1.01	93.50 ± 1.32	78.78 ± 2.49	61.40 ± 1.99

Table 8: Add-7 attention-output interventions, in the same format as Table 7. Mean replacement nearly removes the reverse no-attention contrast; graded interventions preserve the two-sided ordering.

Architecture	Zero	Mean	25% zeroed	50% zeroed	75% zeroed
Dense	32.27 ± 0.10	32.32 ± 0.00	94.05 ± 1.88	71.08 ± 3.76	44.78 ± 3.33
GLU	30.72 ± 2.95	31.04 ± 1.05	90.52 ± 1.65	63.04 ± 2.40	42.40 ± 1.58
MoE	16.83 ± 4.30	31.82 ± 1.00	77.30 ± 2.81	47.32 ± 4.43	28.64 ± 2.04
MoE-GLU	21.82 ± 5.34	32.32 ± 0.00	79.69 ± 1.83	49.73 ± 3.76	30.61 ± 6.33

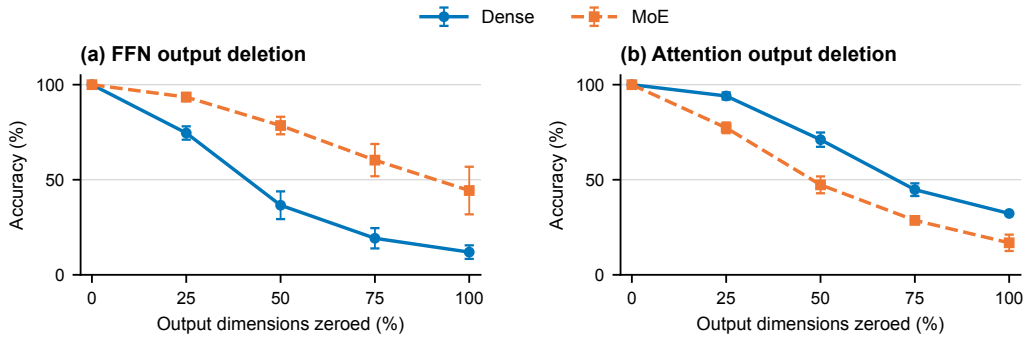


Figure 5: **Graded interventions favor opposite architectures.** Partial deletion of FFN-output dimensions favors MoE, while partial deletion of attention-output dimensions favors dense. Points show mean ± s.d. over five seeds; partial interventions average eight masks per checkpoint. Endpoints show intact and complete zero-ablation accuracy.

C.1 Carry and operation strata

Figure 1b–c separates output positions for the main matched dense/MoE models. No-FFN accuracy at ones, tens, hundreds, and overflow is $9.70/10.60/9.92/17.54\%$ for dense and $43.58/38.02/39.24/56.44\%$ for MoE. No-attention accuracy is $10.00/10.00/9.80/99.30\%$ for dense and $2.00/10.00/10.00/45.30\%$ for MoE. The reverse aggregate contrast is therefore largely an overflow effect. Every dense no-attention model predicts zero at overflow on all 1,000 inputs: it matches 993 majority-class targets and misses all seven genuine overflows. The aggregate is not evidence that dense retains carry computation. The ones output is predicted at sequence position 3 (EOS), after all three input digits, not at the first causal input position.

Table 9: No-FFN add-7 accuracy by carry-chain length. The $L = 3$ row has only seven examples and should not be used inferentially.

Architecture	$L = 0$ ($n = 300$)	$L = 1$ ($n = 630$)	$L = 2$ ($n = 63$)	$L = 3$ ($n = 7$)
Dense	11.17 ± 3.1	12.41 ± 3.9	8.73 ± 3.7	31.43 ± 4.2
GLU	9.73 ± 2.4	9.75 ± 2.4	6.83 ± 3.1	29.29 ± 1.4
MoE	41.28 ± 16.0	45.75 ± 10.8	43.02 ± 18.1	57.14 ± 15.3
MoE-GLU	40.98 ± 5.2	42.22 ± 2.5	39.44 ± 8.3	42.14 ± 28.2

The effect is not confined to pass-through tokens. Dense/MoE no-FFN accuracy is $9.70/43.58\%$ on $+7$, $10.86/38.15\%$ on $+1$ carry, and $13.32/46.81\%$ on $+0$. Because class sizes differ, these strata corroborate breadth but do not replace the balanced intervention comparisons.

C.2 Constant-offset and direct-path controls

Mean replacement increases dense and MoE no-FFN accuracy to 37.4% and 66.0%, showing that both FFNs supply useful position-dependent constants. The architecture gap nevertheless remains 28.6 points. On the ones position, zero-to-mean replacement changes dense from 9.7% to 73.5% and MoE from 43.6% to 83.9%. A separate direct-path correction subtracts the constant direct contribution after FFN removal: dense ones-digit accuracy rises by 19.7 points, while MoE falls by 5.9 points. Thus dense models rely more heavily on FFN/direct-path bias cancellation; this nuisance explains part of absolute zero-ablation behavior but not the mean-replacement gap or the graded two-sided curves.

D EXPERT-COUNT AND ROUTING CONTROLS

Table 10: Restoring active FFN capacity weakens the no-FFN contrast. All conditions solve add-7; values are digit accuracy, mean ± population s.d. over five seeds. Comparisons change more than width alone and do not form an additive decomposition. The last two rows provide a within-MoE-GLU comparison.

Condition	No FFN	Manipulation and interpretation
Dense FFN	11.9 ± 3.6	Always-active full width
Narrow dense	23.7 ± 5.3	Matches one expert's active width without partitioning
Top-1 MoE	44.3 ± 12.5	Main narrow routed comparison
Top-2 MoE	18.7 ± 6.9	Doubles active width; adds mixing and router task gradients
Shared expert	14.4 ± 4.6	Adds an always-active width-64 path
Total-parameter MoE-GLU	41.7 ± 2.8	Narrow top-1 expert, matched total FFN parameters
Active-width MoE-GLU	18.3 ± 5.6	Restores dense per-token expert width

Table 11: Expert-count sweep at fixed total FFN width (add-7 percentage; five seeds). Per-token active expert width is $256/E$.

E	Active width	Intact	No attention	No FFN
1	256	100.00	32.27 ± 0.10	10.52 ± 3.92
2	128	100.00	28.32 ± 4.06	21.04 ± 5.95
4	64	100.00	16.83 ± 4.30	44.32 ± 12.51
8	32	99.98	10.33 ± 1.87	75.72 ± 6.12
16	16	100.00	8.83 ± 3.00	89.28 ± 1.48

Increasing E changes partition count, active expert width, and the balancing objective together. The $E = 4$ point reuses the main MoE runs; $E = 1$ is a separately trained single-expert model, not the main dense baseline. The narrow-dense and top-2 controls in Table 10 test related architectural changes, but do not provide complete factorial identification. In particular, top-2 changes expert mixing and restores a differentiable task-loss path through the relative selected gate weights, in addition to doubling active width.

Table 12: Actual saved update counts in the primary expert sweep, in seed order. All runs use the same strict-improvement selection rule and balancing coefficient 0.01. These are not fixed-budget endpoints.

E	42	137	256	512	1024
1	400	200	400	400	400
2	400	400	600	600	600
4	400	400	600	400	600
8	600	400	600	600	800
16	600	600	1000	800	800

Higher expert counts are generally selected later. The observed dose response therefore describes the saved solutions under this protocol; it does not separate architecture from the possible contribution of different evaluation times.

D.1 EXPERT SWEEP WITHOUT BALANCING AT A COMMON ENDPOINT

We train a separate $E \in \{1, 2, 4, 8, 16\}$ sweep without the auxiliary loss and evaluate every model at update 3,000. Router weights remain frozen, but assignments are recomputed from the evolving residual stream. The five seeds are $42, 137, 256, 512, 1024$. Each model has one layer, four causal heads, $d_{\text{model}} = 64$, SiLU experts with total hidden width 256, tied embeddings/readout, and no

normalization or dropout. Training uses AdamW with learning rate 10^{-3}, weight decay 0.01, batch size 128, and norm-one gradient clipping.

Within each seed, all expert counts share identical initial embeddings, attention, and positional parameters, and the same pregenerated stream of uniformly sampled inputs. Expert and router shapes vary with E. The endpoint and analysis rules are recorded before inspecting outcomes; every run is included, without early stopping or an accuracy threshold. We save checkpoints every 500 updates and evaluate intact, zero-FFN, and mean-FFN accuracy on the full 1,000-input support. FFN means are position-specific and estimated from that checkpoint's intact teacher-forced activations on the same support.

Table 13: No-balancing sweep at the common 3,000-update endpoint. Digit accuracy is mean $\pm$ sample s.d. over five seeds. The primary balanced, accuracy-selected sweep remains a separate experiment.

E	Active width	Intact	Zero FFN	Mean FFN
1	256	100.0 ± 0.0	10.6 ± 4.2	40.8 ± 2.2
2	128	100.0 ± 0.0	24.2 ± 2.7	51.8 ± 4.1
4	64	100.0 ± 0.0	58.4 ± 17.3	73.0 ± 9.7
8	32	100.0 ± 0.0	67.3 ± 14.9	78.5 ± 9.0
16	16	99.8 ± 0.4	80.9 ± 8.9	88.2 ± 5.0

Mean retained accuracy increases at each expert-count transition under both interventions. This ordering of means also appears at every saved positive update, from 500 to 3,000. At the final endpoint, every seed increases under mean replacement; zero-FFN accuracy dips from $E = 4$ to 8 in two seeds and ties from $E = 8$ to 16 in one. All but one model have perfect intact digit and autoregressive exact-match accuracy; the $E = 16$, seed-256 model scores 99.2% and 96.8%, respectively, and remains included. Thus the average dose response does not require balancing or unequal training durations, although individual zero-ablation curves vary. This replication jointly removes the auxiliary objective and matches training exposure; its difference from the original sweep is not an isolated estimate of the balancing loss's effect. Active width and partitioning still vary together.

All 25 final checkpoints are independently rescored. Validation also checks matched initial upstream tensors and batch streams, unchanged router weights, source hashes, and the causal mask. Protocols, per-seed endpoints, and common-update trajectories are retained in the artifact manifest.

D.2 Matched top-1 route replay

The paired top-1 replay experiment initializes two models identically, freezes W_r in both, uses identical batches, and removes the auxiliary load-balancing loss. In the dynamic condition, the frozen router is evaluated on the evolving residual stream. In replay, a table indexed by the complete input example and sequence position stores routes computed from the initial post-attention representations. These context-derived assignments stay fixed throughout training; replay is not a context-independent token-identity partition. Both runs train for 10,000 updates, but the reported checkpoints are separately selected by sampled autoregressive accuracy and need not occur at the same update. This comparison estimates the assignment effect under that selection rule; the fixed-budget follow-up below compares equal update counts.

Table 14: Per-seed top-1 route replay at accuracy-selected checkpoints, not a common update. Agreement is with initialization-time assignments.

Condition	42	137	256	512	1024	Mean $\pm$ s.d.
Dynamic no FFN	24.18	53.20	35.23	45.63	48.85	41.42 ± 10.46
Replay no FFN	11.38	21.70	17.90	25.55	39.05	23.12 ± 9.24
Difference	12.80	31.50	17.33	20.08	9.80	18.30 ± 7.49
Dynamic agreement	71.25	49.96	66.78	60.53	57.94	61.29 ± 7.34

All five paired differences are positive. Using sample variability for inference gives paired 95% CI $[7.90, 28.71]$, $t(4) = 4.88$, two-sided $p = .0081$. Replay reaches 100% intact accuracy in every seed; dynamic reaches $99.95 \pm 0.09\%$. No-attention accuracy is $19.32 \pm 6.94\%$ for replay and $18.83 \pm 7.00\%$ for dynamic: the large observed contrast is on the no-FFN side, without establishing equivalence on the other side.

The separately trained main-regime comparison is not paired and should not be interpreted as an equivalence test. Frozen W_r gives $49.1 \pm 9.5\%$ no-FFN accuracy and auxiliary-trained W_r gives $44.3 \pm 12.5\%$, an estimated frozen-minus-trained difference of $+4.8$ pp with approximate Welch 95% CI $[-11.7, 21.2]$ pp. Thus the effect clearly exists without boundary updates, but this comparison is too imprecise to determine whether those updates alter its magnitude.

Assignment trajectories confirm that frozen parameters do not produce static behavior. With W_r frozen, assignments change by 2.3–4.2% per 250-step snapshot over the latter half of training across five seeds, and midpoint-to-final agreement ranges from 65.2 to 77.6%. Auxiliary-trained-W_r trajectories change more rapidly in the three recorded seeds (10.4–14.8% late-half churn). These trajectory summaries are descriptive because the regimes and seed counts differ; the fixed-budget replay follow-up also matches evaluation time.

Fixed top-1 replay and narrow dense have nearly identical point estimates (23.1% and 23.7%), but they are not an equivalence result. Replay retains four separately parameterized experts and a static initialization-derived partition, whereas narrow dense has one always-active width-64 FFN. The top-1 $+18.3$ pp contrast concerns paired runs under accuracy-based checkpoint selection; proximity to narrow dense does not establish an additive decomposition.

D.2.1 Fixed-budget replication

The separate study underlying Table 1 tests the assignment contrast at a common endpoint. It uses the same five seeds, four width-64 GELU experts, one layer, $d_{\text{model}} = 64$, four causal heads, tied embeddings/readout, and no normalization. Dynamic and replay share initial parameters and a pregenerated minibatch stream within each seed; both freeze W_r and omit balancing. Each trains for exactly 3,000 AdamW updates with batch size 128, learning rate 10^{-3}, weight decay 0.01, and norm-one gradient clipping. The primary endpoint is the final update, fixed before inspecting outcomes. All models reach 100% intact digit and autoregressive exact-match accuracy on the full support.

Table 15: Fixed-budget top-1 replication already included in the variability study. All five dynamic-minus-replay differences are positive for both FFN interventions. Scores are percentages; spread is sample s.d. across five seeds, and intervals use paired t inference. The secondary best-accuracy rows concern this new study, not rescoring of the original 10,000-update runs.

Endpoint	Intervention	Dynamic	Replay	Difference [95% CI]
Update 3,000	Zero FFN	49.96 ± 8.29	24.99 ± 9.77	$24.98\ [12.48, 37.47]$
Update 3,000	Mean FFN	68.53 ± 5.22	52.26 ± 6.57	$16.27\ [9.76, 22.79]$
Selected best	Zero FFN	41.41 ± 11.08	23.24 ± 10.44	$18.17\ [11.20, 25.14]$
Selected best	Mean FFN	62.15 ± 7.21	51.25 ± 4.91	$10.90\ [4.51, 17.30]$

At update 3,000, the five zero-FFN differences are $21.30, 32.30, 36.90, 23.28, 11.10$ points, in seed order $42, 137, 256, 512, 1024$. The equal-update result supports an assignment effect without different training durations or an auxiliary objective. It is a short-horizon replication within frozen-router hard top-1, not a test of which factor mediates the primary balanced sweep.

As a secondary endpoint, the study retains the earliest checkpoint attaining its highest exhaustive autoregressive exact-match accuracy, evaluated every 250 updates. Both endpoint conventions preserve the direction. Evaluation uses all 1,000 inputs and four output digits; position-specific FFN means are estimated on the same full support. Complete trajectories, per-seed records, and independent checkpoint-rescoring checks are in the artifact manifest.

D.3 WEIGHTED TOP-2 MEMBERSHIP REPLAY

This follow-up asks whether changing expert identities affects FFN-removal performance when both models have task-trained mixture weights. Both arms have four width-64 GELU experts, $k = 2$, one layer, $d_{\text{model}} = 64$, four heads, tied embeddings/readout, and no normalization or dropout (58,944 parameters). For router logits $r_e(h)$ and selected expert set S, the FFN is

$$F(h; S) = \sum_{e \in S} \frac{\exp r_e(h)}{\sum_{j \in S} \exp r_j(h)} F_e(h). \tag{3}$$

The dynamic arm reselects the current top two experts. Replay fixes their initialization-time identities separately for every full input example and sequence position. It still recomputes the current normalized weights of those same experts. Both arms therefore train router weights through the task loss; neither uses a balancing loss. The intervention fixes expert membership, not all routing adaptivity.

Five seed pairs $(42, 137, 256, 512, 1024)$ share initial parameters and actual minibatch streams. Each runs 10,000 updates with batch size 128, AdamW learning rate 10^{-3}, weight decay 0.01, and gradient clipping at norm one. The primary endpoint is the final fixed-budget checkpoint. As a secondary check, we retain the first checkpoint attaining the best exhaustive autoregressive exact-match score, including final EOS, evaluated every 200 updates. This monitor differs from the earlier top-1 experiment's sampled 256-example selection, so cross-k score differences are not an isolated routing comparison. Ablations evaluate all 4,000 output digits on the finite add-7 support. Mean replacement uses each checkpoint's own position-specific intact FFN means on that support.

Preflight checks on seed 42 reproduce native logits, task gradients, and the first optimizer update exactly in both implementations. In all five pairs, both routers receive nonzero task gradients and update during training. Initial-parameter and batch-stream hashes match within pairs; replay membership stays fixed, and membership-change statistics ignore swaps in pair order. Final and selected-best checkpoints are saved separately and independently rescored.

Table 16: Weighted top-2 replay at the prespecified final 10,000-update endpoint. Accuracies are percentages; Δ is dynamic minus replay in percentage points. All ten runs reach 100% intact digit and autoregressive exact-match accuracy. Summary rows use sample s.d. across five seeds.

	Zero FFN			Mean FFN replacement		
Seed	Dynamic	Replay	Δ	Dynamic	Replay	Δ
42	13.03	9.60	$+3.43$	57.28	53.88	$+3.40$
137	21.75	24.80	-3.05	49.23	48.25	$+0.98$
256	31.60	18.95	$+12.65$	58.28	52.28	$+6.00$
512	19.30	19.80	-0.50	46.30	46.75	-0.45
1024	58.98	27.05	$+31.93$	69.98	58.20	$+11.78$
Mean	28.93	20.04	$+8.89$	56.21	51.87	$+4.34$
Sample s.d.	18.08	6.75	14.19	9.24	4.57	4.83

The primary no-FFN contrast has paired 95% CI $[-8.73, 26.51]$ pp, with three positive and two negative seed differences. Mean FFN replacement gives $+4.34$ pp (CI $[-1.65, 10.33]$). The result is imprecise, not evidence of equivalence or a robust positive effect. No-attention accuracy is $27.83 \pm 5.42\%$ for dynamic and $22.17 \pm 8.50\%$ for replay (paired difference $+5.66$ pp, CI $[-0.73, 12.05]$). Uncalibrated no-FFN cross-entropy is 15.69 ± 6.65 versus 17.58 ± 3.21 nats per digit; its paired difference is also uncertain (-1.89, CI $[-6.35, 2.58]$).

Table 17: Secondary selected-best weighted top-2 comparison (mean ± sample s.d., five seeds). The selected checkpoints occur at 200–800 updates and all attain 100% intact digit and autoregressive exact-match accuracy. Intervals are paired 95% t intervals.

Intervention	Dynamic	Replay	Difference [95% CI]
Zero FFN	18.75 ± 7.74	17.96 ± 6.36	$+0.79\ [-2.09, 3.66]$
Mean FFN	50.44 ± 2.99	51.48 ± 3.38	$-1.04\ [-3.79, 1.71]$
Zero attention	27.18 ± 7.73	23.95 ± 7.63	$+3.23\ [0.67, 5.78]$

At the final endpoint, the mean largest selected gate over output-digit positions is 0.829 ± 0.021 for dynamic and 0.841 ± 0.013 for replay. The mean entropy-based effective expert count, $\exp(-\sum_e w_e \log w_e)$, is 1.495 ± 0.045 and 1.468 ± 0.038, respectively; these are weighted mixtures, not unit-gate top-1 models. Replay's weights remain input-dependent even though its expert identities are fixed. This test does not establish that the earlier top-1 replay advantage generalizes to weighted top-2. It also does not isolate which change from top-1 matters: active width, mixing, router task gradients, and checkpoint selection differ across the two experiments. The final-versus-selected estimates show why the endpoint must be stated explicitly.

D.4 Stream-independent, shared-path, and routing-association controls

Token hashing gives $11.90 \pm 2.54\%$ no-FFN and $26.17 \pm 5.45\%$ no-attention accuracy. Because hashing also changes load and grouping, it is weaker than paired replay. In the simplified shared-expert model, removing every FFN gives $14.39 \pm 4.59\%$. Removing only the routed path leaves $52.82 \pm 11.33\%$, while removing only the shared path leaves $43.59 \pm 8.08\%$: both paths contribute, but an always-active path suppresses the global reliance shift.

The shared-expert control uses one always-active width-64 expert plus top-1 selection among three width-64 routed experts, with GELU activations. Its routed output is multiplied by the selected softmax probability, without renormalizing that single gate to one. Thus it changes gating as well as connectivity and active width relative to the primary hard top-1 model; it does not isolate the shared path alone.

Operation–expert normalized mutual information is 0.236 ± 0.083 for auxiliary-trained-W_r MoE and 0.183 ± 0.106 for frozen-W_r MoE, versus a chance simulation of 0.0008 ± 0.0005. The corresponding MoE-GLU values are 0.253 ± 0.122 and 0.224 ± 0.079. These pooled associations can reflect output position as well as operation: $+7$ occurs only at the ones position. They neither establish that association was learned nor identify a specialization mechanism. The paired, position-conditioned comparison below addresses these limitations more directly.

D.5 Operation alignment in the paired replay comparison

We evaluate the same five dynamic/replay checkpoint pairs used in Table 14, without retraining. Replay assignments are read from each saved route table, not from a newly evaluated router. Reconstructing the initial model reproduces this table exactly in every seed; the dynamic router weights also remain identical to initialization. Both arms reproduce their saved intact and no-FFN scores. Checkpoints are the existing selected best-accuracy states, not necessarily equal training steps.

On all 1,000 inputs and four output positions, let $O \in \{+7, +1, +0\}$ denote operation, E expert identity, and P output position. We report pooled arithmetic-normalized mutual information, $2I(O;E)/(H(O) + H(E))$, and the position-conditioned quantity

$$I(O; E \mid P) = \frac{1}{4} \sum_{p=1}^{4} I(O; E \mid P = p).$$

Information is measured in bits. The ones-position operation is constant and contributes zero conditional information. For each seed, 2,000 permutations shuffle operation labels independently within positions, preserving operation frequencies and expert loads; the two arms share the same shuffled labels. The generator seed is $20260914+$ training seed. This is an association check on the exhaustive finite support, not a mediation test or a held-out generalization result.

Table 18: Operation alignment in the actual paired route-replay checkpoints. Mean ± sample s.d. across five seeds. Both arms exceed the within-position permutation baselines, but the higher-survival dynamic arm has lower position-conditioned alignment in every observed pair.

Arm	Pooled NMI	$I(O;E \mid P)$	Permuted MI	No-FFN accuracy
Fixed replay	0.316 ± 0.113	0.0545 ± 0.0176	0.00123	23.12 ± 10.33
Dynamic	0.181 ± 0.072	0.0346 ± 0.0168	0.00127	41.42 ± 11.70

In seed order $\{42, 137, 256, 512, 1024\}$, dynamic-minus-replay conditional information is $\{-0.00057, -0.02636, -0.00265, -0.04212, -0.02789\}$ bits, while every no-FFN accuracy difference is positive. The mean information difference is -0.0199 bits with paired t-based 95% CI $[-0.0420, 0.0022]$; five seeds do not precisely estimate a population decrease. The observed survival gain nevertheless does not require an observed increase in final operation alignment. This limits the simple explanation that dynamic routing helps by separating these three operation classes more strongly. It does not rule out other expert roles, transient alignment during learning, or optimization effects. Pooled NMI falls in four of five pairs, not all five. Overflow includes only seven carry cases, so its separate MI is descriptive.

D.6 Gated top-1 with task-trained routing

Does the FFN-removal advantage survive when the router receives task gradients? We compare renormalized hard top-1 with gated top-1,

$$F(h) = p_{e^*(h)}(h) F_{e^*(h)}(h), \qquad p(h) = \mathrm{softmax}(W_r h), \quad e^*(h) = \arg\max_e p_e(h).$$

The selected probability is not renormalized to one. Each token still uses exactly one expert, but the gate now provides a task-gradient path to W_r.

Matched study. We train dense and both routing variants at $E \in \{4, 16\}$ with five seeds $(42, 137, 256, 512, 1024)$: 25 new runs. All models have one layer, $d_{\mathrm{model}} = 64$, four causal heads, SiLU, tied embeddings/readout, and no normalization or dropout. Total FFN hidden width is 256, giving expert widths 64 and 16. Hard and gated models share identical initial parameters within each seed and expert count; all five arms share upstream initialization and pregenerated training batches within each seed. Both MoE variants use balancing coefficient 0.01 over all sequence positions. There is no token dropping or expert capacity limit.

The protocol fixes 3,000 AdamW updates, learning rate 10^{-3}, weight decay 0.01, batch size 128, and norm-one gradient clipping. Training loss covers the four output digits and final EOS. Every run is included at the fixed endpoint, without early stopping. Evaluation uses all 1,000 add-7 inputs from training support. Mean replacement averages each checkpoint's intact FFN output, including gate scaling, separately by position on these same inputs. All 25 runs reach 100% intact teacher-forced digit and autoregressive exact-match accuracy.

Table 19: **The advantage survives gated top-1 but is smaller than under renormalized hard top-1.** Final-update teacher-forced digit accuracy (%, mean ± sample s.d., five seeds). The last column is complete-answer autoregressive accuracy after mean FFN replacement, including final EOS; it is not digit accuracy. These runs use a different training and selection protocol from the original expert sweep.

Model	Zero FFN	Mean FFN	Mean-FFN AR exact
Dense	11.3 ± 3.5	40.7 ± 3.5	0.5 ± 0.2
Hard top-1, $E = 4$	54.4 ± 16.3	72.2 ± 6.9	22.1 ± 14.6
Gated top-1, $E = 4$	39.0 ± 5.3	56.2 ± 3.9	5.4 ± 4.5
Hard top-1, $E = 16$	96.4 ± 2.6	96.6 ± 2.4	87.7 ± 7.9
Gated top-1, $E = 16$	70.2 ± 7.5	73.2 ± 9.2	26.9 ± 14.8

Table 20: Paired teacher-forced digit-accuracy differences at update 3,000, in percentage points with unadjusted 95% t intervals over five seeds.

Contrast	Zero FFN	Mean FFN
Gated $E = 4$ minus dense	$+27.7$ $[18.2, 37.2]$	$+15.4$ $[10.1, 20.8]$
Gated $E = 16$ minus dense	$+58.9$ $[49.0, 68.8]$	$+32.5$ $[17.5, 47.5]$
Gated minus hard, $E = 4$	-15.4 $[-34.5, 3.7]$	-16.0 $[-20.0, -12.0]$
Gated minus hard, $E = 16$	-26.2 $[-34.5, -17.8]$	-23.4 $[-33.2, -13.5]$

Gated MoE exceeds dense in every seed at both expert counts under zero and mean replacement. Under mean replacement, it scores below matched hard top-1 in every seed, retaining about half to 60% of the hard-top-1 advantage over dense. At $E = 4$, the zero-FFN gated-minus-hard interval includes zero. Gated $E = 16$ also exceeds gated $E = 4$ on average: $+31.2$ points under zero replacement (CI $[20.0, 42.4]$) and $+17.1$ under mean replacement (CI $[2.8, 31.3]$). The seed-mean mean-replacement ordering, dense $<$ gated $<$ hard, holds at every recorded evaluation from update 250 onward.

Checks and interpretation. All experts receive at least 1% of all-position assignments in every MoE run; the largest expert load is below 27% at $E = 4$ and 9% at $E = 16$. Mean selected probabilities are 0.77 and 0.51 for gated routing, versus 0.54 and 0.105 for hard routing, where these probabilities are renormalized to one before weighting expert outputs. Checks confirm the intended task-gradient paths, matched initial parameters and batch streams, causal and loss masks, and evaluation RNG neutrality. Independent rescoring of all 25 final checkpoints reproduces the accuracies exactly. Cross-entropy, per-seed values, and trajectories are included in the saved summary (Appendix L).

Thus the advantage does not require a task-gradient-free router. Gating also changes FFN output magnitude and gradients into experts and upstream representations, so the attenuation does not isolate router learning alone. This is a one-layer add-7 check at two expert counts, not a new scaling law or a gated dynamic-versus-replay test.

E Depth, width, and task boundaries

E.1 Add-7 depth

Table 21: Add-7 depth grid under family-wide zero ablation (percentage; five seeds).

Depth	Architecture	Intact	No attention	No FFN
1	Dense	100	32.27 ± 0.10	11.94 ± 3.57
	GLU	99.95 ± 0.06	30.72 ± 2.95	9.70 ± 2.42
	MoE	100	16.83 ± 4.30	44.32 ± 12.51
	MoE-GLU	99.95 ± 0.06	21.82 ± 5.34	41.68 ± 2.78
2	Dense	100	32.27 ± 0.10	10.65 ± 2.61
	GLU	100	32.17 ± 0.30	8.56 ± 1.08
	MoE	99.98 ± 0.04	18.78 ± 7.20	45.49 ± 6.12
	MoE-GLU	99.92 ± 0.11	22.23 ± 2.74	36.44 ± 4.57
3	Dense	100	31.7 ± 1.2	9.3 ± 1.5
	GLU	100	32.3 ± 0.1	8.8 ± 1.8
	MoE	100	25.3 ± 5.6	38.2 ± 12.1
	MoE-GLU	100	26.3 ± 3.7	28.5 ± 5.4
6	Dense	99.9	32.3 ± 0.1	9.2 ± 2.2
	MoE	100.0	30.3 ± 1.9	26.8 ± 9.9

At two layers, removing only MoE FFN layer 0 or 1 leaves $77.81 \pm 7.55\%$ or $89.13 \pm 5.31\%$, whereas removing both leaves $45.49 \pm 6.12\%$. At three layers, the corresponding single-layer removals leave

85.8%, 95.2%, and 98.1%, while removing all leaves 38.2%. At six layers, deleting any single FFN leaves 95.3–99.8% for dense and 96.9–99.7% for MoE, whereas deleting every FFN leaves 9.2% and 26.8%, respectively. The family-wide no-FFN gap therefore remains $+17.5$ pp at six layers, while the no-attention gap attenuates to -2.0 pp.

E.2 Cross-task depth

For modular addition, the one-layer grid uses the first saved checkpoint above 99% held-out accuracy; the two- and three-layer grids use the best held-out-accuracy checkpoint from each completed run. Reported scores and the 98% inclusion threshold use all 113^2 input pairs. The depth grid therefore compares selected trained models, not matched stages of learning.

Table 22: Modular-addition depth results conditioned on intact accuracy $\geq 98\%$. Unequal n reflects different convergence rates.

Depth	Dense n	MoE n	Intact D/M	No FFN D/M	Gap
1	3	5	99.668/99.635	1.590/7.477	+5.888
2	5	4	99.392/99.346	1.043/1.844	+0.801
3	2	2	99.252/99.326	0.670/0.983	+0.313

No-attention accuracy is $1/113 = 0.885\%$ in every included modular-addition cell.

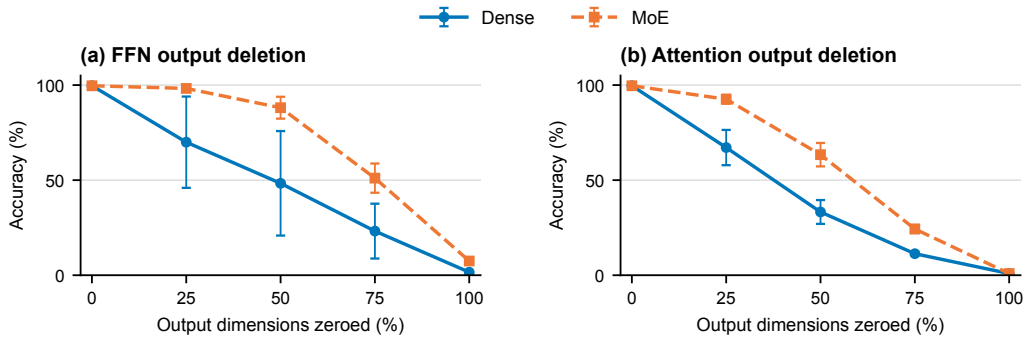


Figure 6: Graded component interventions on one-layer modular addition. Partial FFN deletion separates dense and MoE more clearly than full deletion; partial attention deletion does not give a comparably clear reverse ordering. The curves are reported separately from the add-7 evidence and use five seeds (mean ± s.d.).

Table 23: Histogram depth results for dense and MoE models (percentage; five seeds). No-FFN accuracy remains near the 10% floor and the two-sided redistribution signature is absent.

Depth	Intact D/M	No FFN D/M	No attention D/M
1	99.93±0.0/99.64±0.1	12.06±0.5/12.75±1.6	18.74±1.4/16.69±1.3
2	99.91±0.1/98.87±0.2	11.11±0.8/11.16±0.5	19.79±0.0/19.43±0.2
3	99.97±0.0/99.40±0.2	10.81±0.3/10.89±0.5	19.79±0.0/19.78±0.0

Histogram can be learned with either component family held at random initialization (Table 31), yet ordinary joint training shows little architecture-dependent reliance shift. The frozen block remains in these models, so this does not establish solvability by either component alone. It does show that success under either training constraint is insufficient to predict the reliance contrast under ordinary joint training.

E.3 TINYSTORIES DEPTH AND WIDTH

Table 24: TinyStories zero-ablation architecture gaps across depth at $d_{\text{model}} = 128$. Positive no-FFN and negative no-attention values are the two-sided signature.

Depth	Parameters D/M	No-FFN gap	No-attention gap
1	0.23/0.23M	+4.37	−3.31
2	0.423/0.425M	+6.83	−2.78
3	0.62/0.62M	+4.98	−3.54
6	1.212/1.218M	+6.09	−2.14

Table 25: Two-layer TinyStories width sweep (three seeds). The primary width comparison uses $d \in \{128, 256\}$; $d = 512$ is retained only as a fixed-budget diagnostic. Zero and mean interventions are kept separate; CE is temperature-calibrated mean-no-FFN cross-entropy.

d	Arch.	Params	Intact	Zero FFN	Mean FFN	Zero attn.	Mean attn.	Cal. CE	CE degradation
128	Dense	422,912	68.77	2.13 ± 0.1	2.15 ± 0.1	16.76 ± 0.7	20.95 ± 2.0	4.3097	3.308
	MoE	424,704	64.92	8.97 ± 3.0	9.89 ± 3.5	13.97 ± 1.9	16.14 ± 3.9	3.6371	2.498
256	Dense	1,632,256	70.18	4.61 ± 0.9	4.85 ± 1.0	10.93 ± 0.7	20.24 ± 0.6	4.0067	3.048
	MoE	1,635,840	66.50	25.31 ± 0.8	25.04 ± 0.3	9.14 ± 0.1	11.07 ± 0.8	2.6162	1.533
512	Dense	6,410,240	69.02	8.67 ± 1.3	9.14 ± 1.7	9.88 ± 0.5	18.67 ± 1.0	3.4896	2.486
	MoE	6,417,408	64.67	25.58 ± 1.3	27.89 ± 1.3	9.41 ± 0.5	11.92 ± 0.9	2.4901	1.341

The uniform 92-character CE is $\ln 92 = 4.522$. We report $d = 512$ only as a matched-budget diagnostic and exclude it from the primary scaling claim because MoE intact accuracy is 4.35 points lower; the primary width comparison uses $d \in \{128, 256\}$.

At six layers and $d = 128$, dense/MoE parameters are 1,212,416/1,217,792; intact accuracy is 72.44/70.13. Zero no-FFN accuracy is $1.94/8.03\%$ and zero no-attention is $16.75/14.61\%$. Mean replacement gives $1.92/7.61\%$ and $22.06/18.30\%$, respectively. Calibrated no-FFN degradation is $3.553/2.604$ nats per character.

F MECHANISTIC CORROBORATION

F.1 INTACT LOGIT ATTRIBUTION ACROSS EXPERT COUNTS

We analyze the same 25 checkpoints as the expert-count sweep, without retraining. For output digit i, let a_i and f_i be the attention and FFN contributions to its correct-token logit in the intact model. Following the existing DLA definition, each checkpoint's attention share is

$$S_A = \frac{1}{4000} \sum_{i=1}^{4000} \frac{|a_i|}{|a_i| + |f_i| + 10^{-10}}. \tag{4}$$

This is a mean of tokenwise ratios, not a ratio of mean magnitudes. It excludes the embedding and positional residual from its denominator and includes the attention output-projection bias in a_i. It is neither FFN-removal accuracy nor a fraction of total computation. Checkpoint hashes and intact/no-FFN accuracies reproduce the sweep; the additive logit reconstruction preserves every prediction.

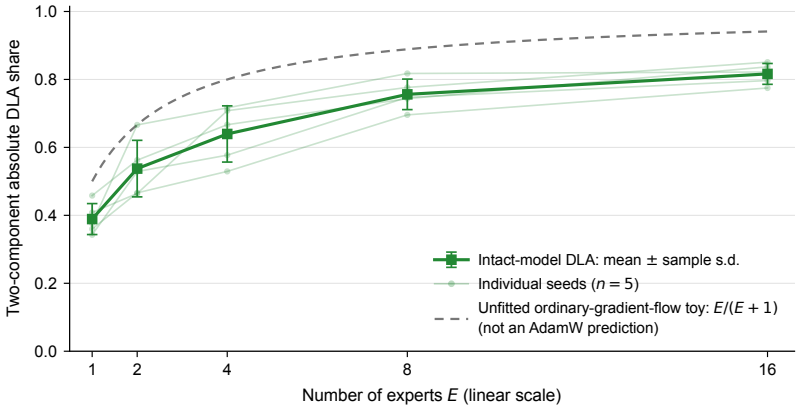


Figure 7: **Intact attention attribution rises with expert count.** Points show the mean and sample s.d. of S_A over five seeds; thin lines show individual seeds. Every observed seed increases at every transition. The dashed curve is the unfitted ordinary-gradient-flow toy in Appendix F.2, not a prediction derived for AdamW or for this DLA metric. The embedding/positional residual is excluded from the two-component share.

Table 26: Intact DLA over the expert sweep (mean ± sample s.d., five seeds). Signed columns are mean correct-token logit contributions, not shares.

E	Attention share S_A	Attention, signed	FFN, signed	Embedding/position, signed
1	0.389 ± 0.046	7.85 ± 3.05	18.29 ± 2.90	5.29 ± 3.00
2	0.538 ± 0.083	15.82 ± 5.39	13.60 ± 6.08	5.28 ± 2.98
4	0.640 ± 0.083	22.44 ± 9.01	10.29 ± 4.47	5.45 ± 2.98
8	0.756 ± 0.045	29.98 ± 7.48	7.42 ± 2.24	5.53 ± 2.90
16	0.816 ± 0.031	32.27 ± 6.39	5.43 ± 2.30	5.56 ± 2.90

The mean and every seed's S_A increase with E. The seedwise $E = 16$ minus $E = 1$ contrast is 0.427 (95% t-based CI $[0.389, 0.465]$). The mean increments per added expert decrease on a linear E axis, but only one of five individual curves is concave throughout. Five seeds also leave uncertainty about adjacent contrasts: the $E = 2$ to $E = 4$ CI narrowly spans zero. Thus the evidence supports a graded intact-attribution shift, not an established saturation law. The alternative ratio of mean magnitudes also increases in the means, but falls from $E = 8$ to 16 in one seed. The all-seed monotonicity statement applies to the existing tokenwise definition above.

F.1.1 Prediction-contrast and output-bias checks

A large correct-token logit contribution need not distinguish the correct answer from its competitors. We therefore repeat the attribution analysis using two contrasts. Let z^C_{ic} be component C's contribution to class c at output position i, and let y_i be the correct class. We measure

$$q^C_i = z^C_{iy_i} - \frac{1}{V}\sum_{c=1}^{V} z^C_{ic}, \qquad m^C_i = z^C_{iy_i} - z^C_{ic^*_i}, \quad c^*_i = \arg\max_{c \neq y_i} z^{\text{intact}}_{ic}, \tag{5}$$

where $V = 12$ includes the full model vocabulary. The first contrast centers logits across classes; the second compares the correct answer with the intact model's strongest incorrect class. The competitor is the same for each component. Both contrasts are unchanged by a class-common logit offset. We substitute q^A_i, q^F_i, or m^A_i, m^F_i, for a_i, f_i in S_A above.

We also subtract the attention output-projection bias, projected through the tied readout, before computing each contrast. This separates a constant attention contribution in the attribution calculation; it does not modify the model's forward pass or expert assignments.

Table 27: Attention attribution after removing prediction-neutral offsets and separating output bias. Entries are tokenwise magnitude shares, mean ± sample s.d. over five seeds. Every column increases at every expert-count transition in every seed.

	Class-centered contrast		Correct–competitor margin	
E	With attention bias	Bias removed	With attention bias	Bias removed
1	0.399 ± 0.043	0.400 ± 0.042	0.460 ± 0.036	0.462 ± 0.036
2	0.542 ± 0.082	0.543 ± 0.081	0.512 ± 0.056	0.511 ± 0.056
4	0.653 ± 0.080	0.653 ± 0.080	0.582 ± 0.040	0.582 ± 0.041
8	0.764 ± 0.049	0.765 ± 0.048	0.686 ± 0.019	0.685 ± 0.020
16	0.822 ± 0.031	0.822 ± 0.031	0.725 ± 0.014	0.724 ± 0.014

The expert-count trend survives all four checks (Table 27). It is therefore present in attribution to prediction contrasts, not only raw correct-token magnitude. These remain absolute attention/FFN attribution shares, excluding the embedding residual; signed contributions can be negative, so the shares are not fractions of beneficial or causal contribution. The analysis uses the same 25 accuracy-selected checkpoints, verifies their hashes, reproduces the original scores, and checks logit reconstruction and offset invariance. Per-seed values and signed contributions are retained in the artifact manifest.

F.2 An idealized learning-rate hypothesis

Motivated by work on relative learning rates (Nguyen & Reddy, 2025), consider one scalar target feature $y > 0$ shared across E equally frequent, fixed partitions. A shared parameter a and a partition-specific parameter b_e predict $a + b_e$, with loss

$$\mathcal{L} = \frac{1}{2E} \sum_{e=1}^{E} (a + b_e - y)^2. \tag{6}$$

Assume zero initialization, equal parameter learning rates, full-batch ordinary gradient flow, and no regularization. Symmetry keeps every $b_e = b$. Then $\dot{a} = -(a + b - y)$ and $\dot{b} = -(a + b - y)/E$, so $a = Eb$ throughout learning. At interpolation,

$$\frac{a}{y} = \frac{E}{E+1}, \qquad \frac{b}{y} = \frac{1}{E+1}. \tag{7}$$

The shared-path fraction is therefore increasing, concave, and saturating in this toy. It is also the minimum-Euclidean-norm interpolating solution: expressing the feature in the shared coordinate costs less parameter norm than duplicating it across experts. These are consequences of this parameterization and optimizer, not a capacity theorem for Transformers. The toy adds one coordinate per expert and does not literally preserve the experiments' total-width budget. With effective output learning rates κ_A and κ_F, the corresponding fraction is $E\kappa_A/(E\kappa_A + \kappa_F)$. Its log odds equal $\log(\kappa_A/\kappa_F) + (\log 2) \log_2 E$: a constant rate ratio changes the intercept, not the slope. If shrinking expert width also changed κ_F, a different curve could result; hidden width alone does not specify that rate.

Figure 7 provides a post-hoc consistency check using a different intact-model measurement. Its mean curve has the qualitative shape, but all five mean DLA shares lie below the unfitted toy fractions. We do not identify the scalar solution share with absolute DLA, or treat shape agreement as independent confirmation of a learning mechanism. Actual attention and FFN pathways are coupled and their feature-learning rates need not be equal. Moreover, Adam's moment normalization can cancel constant gradient rescaling (Kingma & Ba, 2014). In the balanced scalar toy, proportional gradient histories give equal Adam updates to a and b when the numerical stabilizer is zero, yielding a half share from zero initialization; equal decoupled weight decay preserves that symmetry. Thus adaptive optimization need not merely compress the gradient-flow curve. A stochastic signal-to-noise or non-stationarity explanation for the measured curve would require additional measurements.

The toy also has no distinction between fixed and evolving assignments and cannot explain the paired top-1 replay gap. That residual is a useful constraint: a fuller learning account must explain why partitioning changes intact attribution and why changing the partition adds an effect beyond

fixed replay. Neither stronger final alignment to the three operation labels nor this scalar calculation supplies that missing link.

A conditional width-dependent extension. To connect active width to the scalar learning rates, suppose the effective FFN rate is $\kappa_F(w) = cw$ for hidden width w, with $c > 0$ and a fixed shared-path rate κ_A. Let H be the full dense hidden width. The full dense model has one partition and width H; narrow dense has one partition and width H/E; MoE has E partitions of width H/E. The resulting shared-path fractions are

$$S_{\text{dense}} = \frac{\kappa_A}{\kappa_A + cH} < S_{\text{narrow}} = \frac{\kappa_A}{\kappa_A + cH/E} < S_{\text{MoE}} = \frac{\kappa_A}{\kappa_A + cH/E^2}, \qquad E > 1. \tag{8}$$

This additional rate assumption produces the same qualitative ordering as the dense–narrow–MoE no-FFN results. It also gives a MoE log-odds slope of $2 \log 2$, compared with the measured DLA slope of 0.490. The two comparisons jointly constrain a quantitative account of how hidden width affects learning rates; the proposed proportionality remains an assumption to test.

F.3 A DESCRIPTIVE LOG-ODDS FIT

To check more than monotonicity, we re-express each checkpoint's attribution share as $z_{s,E} = \log(S_{A,s,E}/(1 - S_{A,s,E}))$ and fit its seed mean against $x = \log_2 E$. This averages logits across seeds, rather than taking the logit of the mean share. Ordinary least squares with a free intercept and slope gives

$$\bar{z}_E \approx -0.393 + 0.490 \log_2 E. \tag{9}$$

The mean curve is close to linear over the five observed expert counts: $R^2 = 0.993$ and residual RMSE 0.057 natural-logit units. This is a two-parameter empirical fit, not a one-parameter consequence of the toy. Fitting each seed separately gives slope 0.490 ± 0.046 (sample s.d.; 95% t-based CI $[0.433, 0.547]$) and intercept -0.393 ± 0.215 (CI $[-0.660, -0.127]$). The five repeated seeds, not the 25 checkpoint measurements, determine these uncertainty estimates.

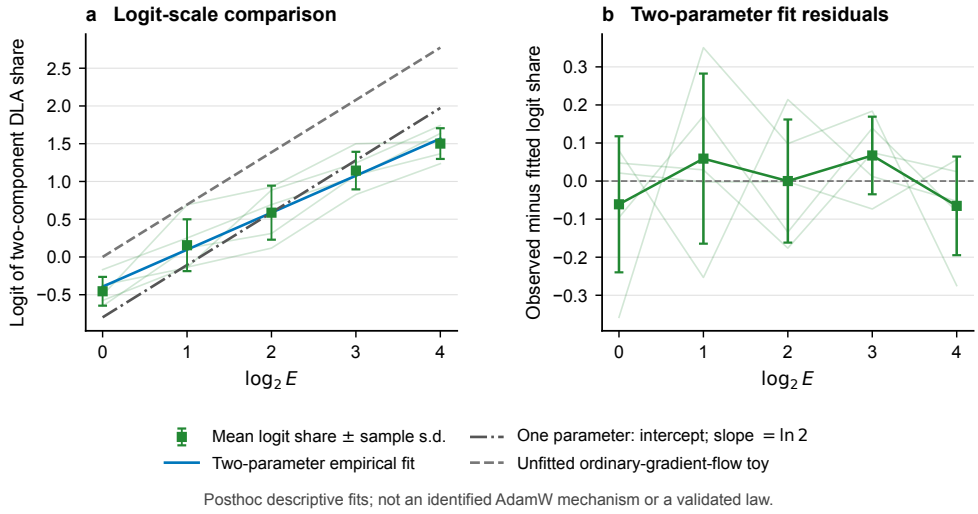


Figure 8: **Log-odds attribution has a compact empirical description.** Points show mean ± sample s.d. across five seeds after transforming each checkpoint's share. Thin lines show seeds; the residual panel subtracts each seed's own two-parameter fit. The ideal toy's slope is $\log 2$ with zero intercept; fitting both coefficients is a descriptive extension. This plot re-expresses existing checkpoints and is not an independent experiment or evidence identifying an optimizer mechanism.

Table 28: Post-hoc fits to mean log-odds attribution. Errors are in natural-logit units. Leave-one-E-out refits use four existing expert counts to predict the fifth; they do not test a new seed or a new scale.

Fit	Free parameters	Intercept	Slope	RMSE / leave-one-E-out RMSE
Unfitted toy	0	0	0.693	0.851/0.851
Slope only	1	0	0.359	0.234/0.246
Intercept only, ideal slope	1	−0.800	0.693	0.293/0.366
Intercept and slope	2	−0.393	0.490	0.057/0.115

The fitted slope is below $\log 2$; a constant relative-rate adjustment in the scalar gradient-flow model changes only its intercept and cannot explain that slope difference. Calling it "Adam compression" would therefore add an untested mechanism. Curvature is not precisely resolved: a seedwise quadratic diagnostic gives a mean coefficient of -0.027 with 95% CI $[-0.112, 0.058]$. This neither establishes exact linearity nor a decisive bend. Interventions on expert learning rates or training noise could test stronger accounts, but these fits alone do not predict how dropout changes the slope or intercept. The empirical log-odds relation and the unexplained top-1 replay gap are constraints for such an account, not its confirmation.

F.4 DLA AND HEAD COUNTERFACTUALS

DLA projects each additive residual component through the unembedding and compares absolute correct-token logit contribution. The shared output-projection bias is excluded from head-level decomposition. Heads are sorted within seed rather than matched by index. At the tens/hundreds positions, the rank-1 share is approximately 0.29 for dense, 0.57 for narrow dense, and 0.63 for both frozen- and learned-routing MoE (uniform share $= 0.25$). This is an intact-model magnitude diagnostic, not a necessity claim.

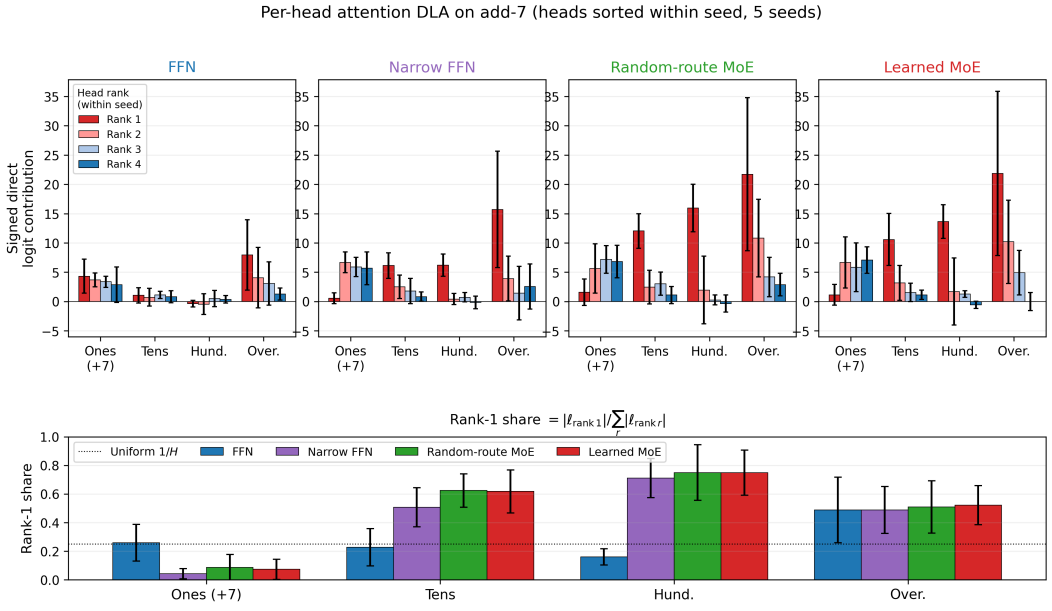


Figure 9: Per-head DLA by position. Heads are ranked within seed. The dominant-head shift occurs at intermediate carry positions; overflow is already concentrated in dense models. Error bars show one s.d. across five seeds.

Table 29: Carry-head counterfactuals (five seeds). The top head is selected independently in each seed under no-FFN carry accuracy.

Arch.	No-FFN carry	Top-head drop	Remaining	Keep 1	Keep 2	Keep 3 / 4
Dense	10.86 ± 0.95	0.95 ± 0.95	9.91	10.17	9.91	10.45 / 10.86
GLU	10.42 ± 1.03	0.51 ± 1.03	9.91	9.91	9.91	10.42 / 10.42
MoE	38.15 ± 9.61	23.35 ± 7.40	14.80	11.20	14.16	24.09 / 38.15
MoE-GLU	30.01 ± 10.45	16.50 ± 8.29	13.51	10.42	14.70	20.03 / 30.01

Removing the same selected head from intact models drops carry accuracy by 43.19 ± 16.49 points for dense, 25.82 ± 21.05 for GLU, 67.72 ± 13.59 for MoE, and 47.26 ± 19.72 for MoE-GLU. The selected head is therefore used before FFN deletion, but the retention curve rules out single-head sufficiency. With intact FFNs, head removal can change both downstream features and expert assignments; these are total intervention effects, not an isolation of direct attention computation. The no-FFN head tests avoid the routing-mediated contribution while characterizing the surviving pathway.

F.5 Head removal with intact expert assignments fixed

This evaluation-time control differs from training-time route replay. For each of the five main MoE checkpoints, we record intact expert assignments for every example and sequence position. We then remove the same head selected by the existing no-FFN carry ranking, while forcing the FFN to use those intact assignments. Experts still receive the head-perturbed representations and recompute their outputs; only assignment changes are prevented. With no head removed, this fixed-assignment implementation reproduces the native logits exactly in all five seeds (maximum absolute error zero). Ordinary head removal reproduces the published dense and MoE effects.

Table 30: Carry-accuracy drop after selected-head removal with the FFN intact (percentage points). Each architecture uses its own within-seed selected head. Summary uncertainties are sample s.d.

Seed	Dense	MoE, routes recomputed	MoE, intact routes fixed
42	30.63	78.51	79.79
137	48.39	62.55	51.09
256	58.82	72.59	72.33
512	17.89	43.76	36.94
1024	60.23	81.21	78.64
Mean $\pm$ s.d.	43.19 ± 18.44	67.72 ± 15.19	63.76 ± 18.92

Ordinary deletion changes expert assignments at 24.19% of output positions and 24.04% of carry positions on average. Preventing these changes still leaves a 63.76-point mean carry drop (95% t interval $[40.27, 87.25]$). The paired recomputed-minus-fixed difference is 3.96 points, with 95% CI $[-2.47, 10.40]$; this does not establish equivalence or zero routing contribution. It shows that rerouting is not required for the large selected-head effect. The fixed-route effect can still pass through downstream FFN transformations. Heads are selected and evaluated on the same exhaustive finite task, so this is not an independently validated circuit identification.

F.6 Activation patching and interpretation

For each non-ones position, all valid matched pairs requiring different operations are constructed (typically 200–400 pairs). One component output at that position is patched from the source into the destination, and flip rate is averaged across five seeds.

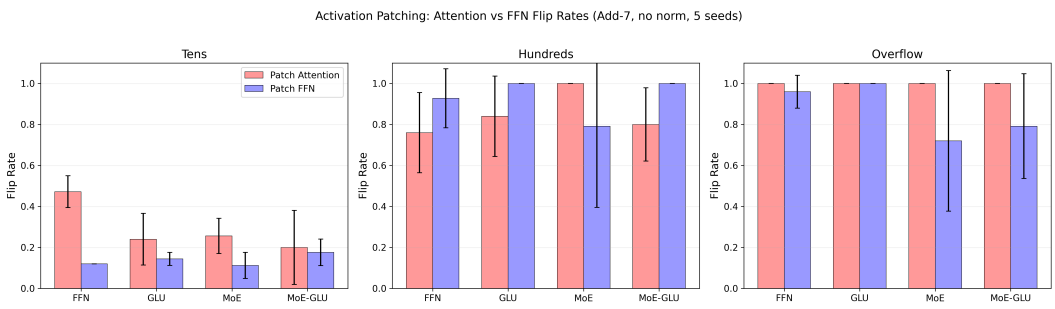


Figure 10: Activation-patching flip rates. DLA and patching need not order cells identically: DLA measures additive correct-logit magnitude; patching measures argmax sensitivity to a source–destination counterfactual.

The tens position illustrates the distinction. Dense DLA assigns more absolute magnitude to the FFN, while attention patching flips predictions more often. Attention can carry the discriminative difference between competing outputs even when the FFN contributes more shared logit magnitude. We therefore report the methods as complementary, not uniformly convergent.

G Matching and additional robustness

G.1 Frozen-component and attention-only training

Table 31: Intact accuracy when one component family is frozen at random initialization during training (five seeds). "Frozen attention" leaves the trainable FFN path; "frozen FFN" leaves trainable attention.

	Frozen attention			Frozen FFN		
Arch.	Add-7	ModAdd	Hist.	Add-7	ModAdd	Hist.
Dense	100	100	98.3 ± 0.6	100	3.9 ± 3.8	96.6 ± 3.4
GLU	100	100	98.9 ± 0.5	100	58.5 ± 46.4	95.6 ± 0.7
MoE	100	99.8	99.3 ± 0.1	100	1.6 ± 1.2	96.0 ± 2.4
MoE-GLU	100	83.5 ± 23.4	99.4 ± 0.1	100	40.5 ± 44.7	94.1 ± 3.1

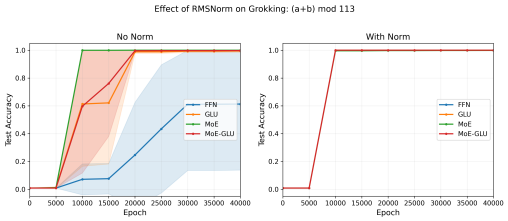
Separately, one- and two-layer attention-only Transformers trained with no FFN reach 100% add-7 digit and exact-match accuracy in all five seeds. These experiments establish solvability, not the organization selected during ordinary joint training.

G.2 Active-FLOP matching

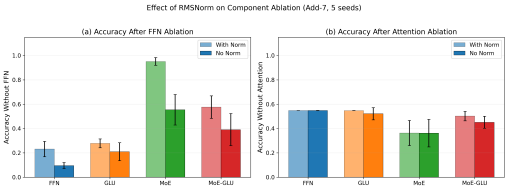
Per-active matching gives every MoE-GLU expert the dense per-token width and approximately four times the dense total FFN parameters. On add-7, dense no-FFN/no-attention accuracy is $11.9 \pm 3.6/32.3 \pm 0.1\%$; total-parameter-matched MoE-GLU gives $41.7 \pm 2.8/21.8 \pm 5.3\%$; and active-width-matched MoE-GLU gives $18.3 \pm 5.6/29.3 \pm 2.9\%$. Thus the two architecture gaps compress from $+29.7/-10.5\,\mathrm{pp}$ to $+6.4/-3.0\,\mathrm{pp}$ when dense-scale per-token expert width is restored. An equivalent GELU active-width replica gives $16.8 \pm 4.2/30.3 \pm 2.0\%$. This is a MoE-GLU capacity-restoration control rather than a systematic plain-MoE active-width sweep, but both intervention sides move in the bottleneck account's predicted direction. Modular addition and histogram remain near their no-FFN floors under both matching conventions.

G.3 Normalization, activation, and massive activations

Primary mechanistic runs omit RMSNorm, following common small-circuit practice. RMSNorm compresses the quantitative gaps but preserves the qualitative no-FFN architecture ordering.



(a) Training with and without RMSNorm.



(b) Add-7 component interventions.

Figure 11: Normalization controls. No-norm is the primary analysis setting; RMSNorm preserves the qualitative comparison with compressed separation.

SiLU and GELU variants reproduce the main GLU/MoE-GLU ordering (Fig. 12). No add-7 checkpoint crosses the $100\times$ component-median massive-activation threshold among 512,000 measurements per component and seed. The largest observed ratio is 16.2 (MoE-GLU FFN output). FFN-output excess kurtosis is 0.83, 2.19, 3.46, and 6.19 for dense, GLU, MoE, and MoE-GLU, respectively.

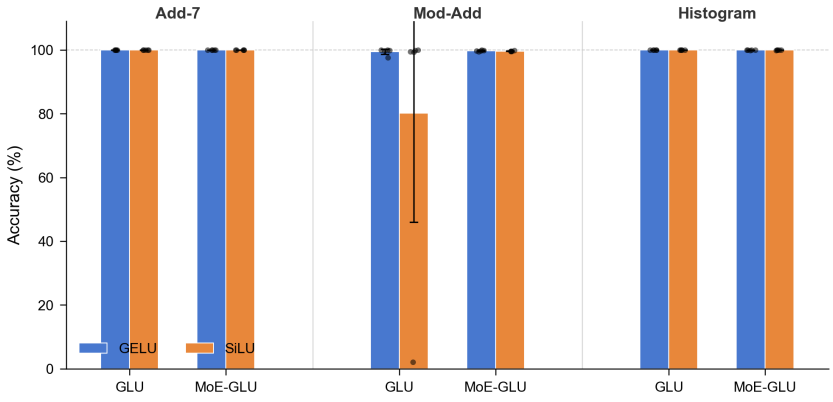


Figure 12: Activation-function control for GLU-family experiments.

H Task-dependent FFN roles: what simple replacements preserve

H.1 Question and connection to prior work

Histogram shows little architecture gap after FFN removal. This does not tell us which other component arrangements could solve counting. Our frozen-component controls show successful alternative training arrangements; Behrens et al. (2025) give explicit counting algorithms that assign different work to attention and the FFN. In their inventory construction, the model aggregates token information and then extracts the count for the queried token. This motivates a possible FFN role, but does not identify the circuit in our multi-head residual models.

Which available solution is learned is a separate question. Nguyen & Reddy (2025) explain competition between memorization and in-context generalization through how quickly each is learned in a simplified attention–MLP model. That mechanism has not been established for sparse FFNs or our sequential residual architecture. We first ask a more direct question: *what transformation does the trained FFN supply under the model's original output head?* Figure 13 tests this without fitting a new task decoder.

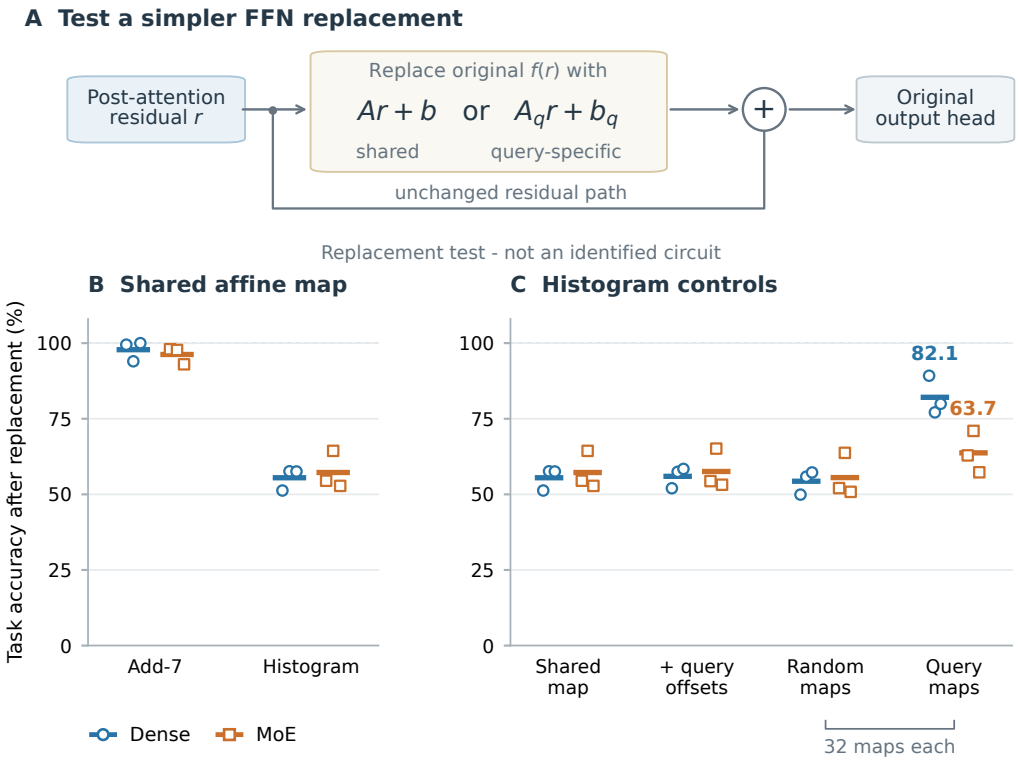


Figure 13: **The trained FFNs differ in what a simple replacement preserves. (a)** Fit a replacement to FFN outputs without changing attention, residual addition, or the original head. Query identity q is supplied externally. **(b)** A shared affine map preserves high overall digit accuracy on add-7 but much less count accuracy on histogram. **(c)** Query-specific maps improve histogram recovery; offsets alone and equally sized random groups do not reproduce the gain. Points show three checkpoint seeds per architecture, with mean bars. This characterizes FFN transformations, not the cause of the architecture gap.

H.2 Replacement protocol

Let r denote the residual after attention, $f(r)$ the original FFN output, and g the native output head. The intact logits are $g(r + f(r))$. We freeze every Transformer parameter and fit an approximation $\hat{f}$ to the teacher's hidden FFN outputs. Evaluation uses $g(r + \hat{f}(r))$: attention, the direct residual connection, and the original output classes are unchanged. Task labels score the final predictions but are not used for fitting or model selection. Thus this is teacher-output distillation, not a label-supervised readout probe.

We use three existing checkpoint seeds per architecture, $\{42, 137, 256\}$: add-7 replicas at a fixed 2,000 updates and histogram checkpoints at epoch 1,000. These exploratory endpoints are distinct from the best-accuracy-selected add-7 checkpoints in the primary ablation grid. The add-7 models have width 64, SiLU, causal attention, and a tied 12-class output head; histogram uses width 128, GELU, bidirectional attention, and an untied 10-class head. Both use four heads and no normalization. The contrast therefore concerns two task/model settings, not a task-only manipulation.

A fixed sequence-level 60/20/20 split (seed 20260908) assigns the 1,000 add-7 inputs to 600/200/200 fit/validation/test examples, giving 2,400/800/800 teacher-forced digit rows. For histogram, 2,993 unique sampled sequences yield 1,795/598/600 sequences, or 17,950/5,980/6,000 token rows. These holdouts concern approximation fitting, not necessarily the original Transformer's training support; different permutations of the same histogram bag can cross splits. All histogram replacement arms share checkpoint, data, and split hashes. Native intact and zero-FFN logits are checked against direct evaluation.

For each affine fit, we select ridge strength by validation hidden-output reconstruction MSE, never test scores. The candidate strengths are $\{10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}, 1\}$. The shared family uses $\hat{f}(r) = Ar + b$. Histogram additionally uses:

- *Shared slope, query-specific offsets:* $Ar + b_q$, where q is the input-token identity at the prediction position.
- *Query-specific maps:* $A_q r + b_q$, fitted separately for each of the 32 identities, with per-group validation selection.
- *Random-group maps:* 32 affine maps with the same nominal parameter count. Query labels are independently permuted within each split (one fixed draw, seed 20260929), preserving every group's size. This is a size-matched uninformative partition, not a stable input-dependent routing rule.

Conditional surrogates receive group identity externally. The query-specific and random-group families each have 32 times as many affine parameters as the global family; the shared-slope control tests whether query-specific offsets alone reproduce the improvement.

H.3 Results and interpretation

One shared affine map preserves 97.83% overall add-7 digit accuracy for dense and 96.25% for MoE. This does not mean the dense FFN is dispensable: its learned transformation is being approximated, not removed. Nor does high aggregate recovery establish uniform recovery across carry operations. Histogram gives a different result (Table 32): query-specific slopes improve on shared slopes plus query offsets by $26.13\,\mathrm{pp}$ for dense and $6.16\,\mathrm{pp}$ for MoE, with positive differences in all three seeds. Random groups do not reproduce this gain.

Table 32: Histogram FFN-replacement accuracy (%, mean ± population s.d. across three checkpoint seeds; 6,000 common test token rows). All approximations fit the teacher's FFN outputs and retain the native head.

FFN output used at evaluation	Dense	MoE
Original FFN	99.89 ± 0.06	99.78 ± 0.09
Zero	12.07 ± 0.63	12.26 ± 2.56
Shared affine map	55.49 ± 2.99	57.23 ± 5.11
Shared slope + query offsets	55.97 ± 2.82	57.57 ± 5.37
Random-group affine maps	54.34 ± 3.18	55.54 ± 5.81
Query-specific affine maps	82.09 ± 5.18	63.72 ± 5.62

Knowing the query identity helps the fitted replacement preserve histogram predictions, especially for dense models. This does not identify an inventory-counting circuit or show that query-specific maps are necessary. Recovery is incomplete, particularly for MoE and high counts. The global affine fit already reconstructs hidden outputs with R^2 of 0.978 for dense and 0.967 for MoE despite much poorer task accuracy. Small reconstruction errors can therefore matter greatly for predictions; the result should not be summarized as "histogram FFNs are less linear." Failure of our fitted approximation also does not show that every possible affine fit would fail.

These tests describe the learned FFN transformation, not why training produced it. The replacements are applied to frozen models after training; they do not hold other pathways fixed while changing how the FFN learns its role. They therefore do not establish that the measured role differences cause the dense–MoE ablation gap, or that different learning speeds explain it. Our working hypothesis is that the same active FFN restriction has different effects because training learns different ways to divide the work across tasks.

H.4 Output-head tying and initialization during training

The fitted-readout results raise a separate training-time question: does the add-7 reliance gap depend on tying the output head to the input embeddings, or on how that head is initialized? We address

this with a matched-data study and a subsequent exploratory scale/orientation follow-up. These experiments train the whole model; they do not fit a new decoder after freezing a checkpoint.

Matched-data protocol. The initial study uses three seeds, $\{42, 137, 256\}$, and six arms per seed: dense or four-expert MoE crossed with tied, untied-copy, or untied-random heads. Models have one layer, $d_{\text{model}} = 64$, four causal attention heads, SiLU FFNs, no normalization or dropout, and a maximum-sequence embedding table of length 128. Dense hidden width is 256; MoE uses four width-64 experts with renormalized hard top-1 gating and auxiliary balancing coefficient 0.01. The router receives auxiliary-loss updates but no task-loss gradient.

All arms receive the same pregenerated minibatches within each seed. We train for exactly 2,000 AdamW updates with batch size 128, learning rate 10^{-3}, weight decay 0.01, $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$, and gradient clipping at norm one. Task loss covers the four output digits and final EOS. We retain the final checkpoint without performance-based stopping or extension. This is a separate protocol from the primary selected-best ablation grid.

Head interventions. The tied head uses the input embedding matrix as its output weights. Untied-copy creates an independent, bias-free output matrix initialized to an exact copy of those embeddings. Its initial predictions and all non-head parameters match the tied arm. Untied-random instead uses the seeded standard initialization of a bias-free linear head. It changes both the head's initial orientation and scale. Shared embedding, attention, and head initializations are matched across architectures where applicable. Untying adds 768 parameters: dense/MoE totals are 58,496/58,944 when tied and 59,264/59,712 when untied.

Evaluation and checks. Tables 33 and 34 use the same 800 teacher-forced digit rows from 200 inputs in the existing fixed 600/200/200 fit/validation/test split. Position-specific mean FFN outputs are estimated from the 600 fit inputs. All 1,000 inputs belong to Transformer training support; the split separates post-training diagnostic fitting from evaluation, not original-model training from testing. No new head is fitted for the zero- or mean-FFN scores below. All 30 checkpoints achieve 100% intact accuracy on these 800 rows. On exhaustive autoregressive evaluation, including EOS, 28 checkpoints achieve 100%; tied MoE seed 137 achieves 99.5% and untied-random MoE seed 256 achieves 99.9%.

Saved records verify identical consumed minibatch hashes within each seed, matching shared initial parameters, and exact reconstruction of intact and zero-FFN logits. We report sample standard deviations over three seeds and expose the individual scores rather than treating diagnostic token rows as independent replications.

Table 33: **Releasing weight tying preserves a large reliance gap; changing head initialization shifts both architectures' scores.** Teacher-forced digit accuracy (%, mean ± sample s.d., three seeds), using the original trained head after zero or mean FFN replacement. Intact accuracy is 100% on the common 800 evaluation rows. The last two conditions were chosen after inspecting the initial study.

	Zero FFN		Mean FFN	
Head used during training	Dense	MoE	Dense	MoE
Tied	12.13 ± 4.77	41.63 ± 8.45	39.29 ± 4.44	66.79 ± 5.60
Untied copy	12.29 ± 4.92	44.21 ± 12.42	39.00 ± 4.11	69.92 ± 7.57
Untied random	78.75 ± 9.33	90.46 ± 7.01	76.67 ± 8.43	92.33 ± 5.31
Exploratory scale/orientation follow-up				
Copy at random-head RMS	46.13 ± 6.96	83.21 ± 10.09	54.42 ± 2.25	82.96 ± 8.53
Random at copied-head RMS	39.71 ± 2.11	80.46 ± 9.32	83.38 ± 2.83	93.96 ± 2.07

Tying and initialization give different answers. Simply untying the head at matched initialization leaves a large no-FFN gap: dense/MoE scores are $12.3/44.2\%$, compared with $12.1/41.6\%$ in the tied arms. Mean replacement gives the same architecture ordering. Weight tying alone is therefore not required for the gap in this setting. By contrast, independently initializing the untied head raises

no-FFN accuracy to $78.8/90.5\%$. Because that comparison changes both initialization scale and orientation, we next separate those two changes.

Exploratory scale/orientation follow-up. After inspecting the initial study, we train 12 additional arms: both architectures, the same three seeds, and two positive rescalings of the initial untied head. Let U_c be the copied embedding matrix, U_r the independently initialized head, and $\rho(U)$ the root-mean-square matrix entry. We initialize

$$U_{c,\text{low}} = \frac{\rho(U_r)}{\rho(U_c)} U_c, \qquad U_{r,\text{high}} = \frac{\rho(U_c)}{\rho(U_r)} U_r.$$

The copied head's RMS is approximately 0.95–1.00, whereas the standard random head's is approximately 0.072. These interventions exchange their initial scales while preserving each matrix's direction. Only the initial output matrix is rescaled; input embeddings and all other initial weights are unchanged. Both heads remain untied and fully trainable. We reuse the exact first-stage minibatches, optimizer settings, and 2,000-update endpoint.

Reducing the copied head's initial scale raises dense/MoE no-FFN accuracy to $46.1/83.2\%$. Increasing the random head's initial scale lowers it to $39.7/80.5\%$. Thus initial output scale affects later reliance, while copied and random orientations also yield different scores at matched initial RMS. Neither weight tying nor initial head RMS alone accounts for all observed differences.

Table 34: Per-seed output-head control results. Zero and mean FFN columns use the common 800 digit rows; intact AR exact uses all 1,000 inputs and requires all four output digits and EOS. Every paired entry is dense/MoE (%). These endpoints are distinct from the primary five-seed ablation grid and the 3,000-update variability study.

Head	Seed	Zero FFN D/M	Mean FFN D/M	Intact AR D/M
Tied	42	6.750/37.125	34.375/73.000	100.0/100.0
	137	13.750/51.375	40.500/65.250	100.0/99.5
	256	15.875/36.375	43.000/62.125	100.0/100.0
Untied copy	42	6.750/33.625	34.375/73.250	100.0/100.0
	137	14.000/57.875	40.375/75.250	100.0/100.0
	256	16.125/41.125	42.250/61.250	100.0/100.0
Untied random	42	71.875/82.625	67.750/86.625	100.0/100.0
	137	89.375/96.125	77.750/97.125	100.0/100.0
	256	75.000/92.625	84.500/93.250	100.0/99.9
Copy at random-head RMS	42	45.500/84.000	52.875/74.875	100.0/100.0
	137	53.375/92.875	53.375/91.875	100.0/100.0
	256	39.500/72.750	57.000/82.125	100.0/100.0
Random at copied-head RMS	42	38.250/70.375	83.750/96.125	100.0/100.0
	137	42.125/88.750	86.000/93.750	100.0/100.0
	256	38.750/82.250	80.375/92.000	100.0/100.0

These controls establish sensitivity to output-head training conditions while preserving a substantial dense–MoE contrast after untying at matched initialization. They concern the jointly learned representation–FFN–readout arrangement. They are not FFN-output rescaling experiments, post-hoc calibration, or evidence that the architectures learn identical attention representations. The follow-up is exploratory, and three seeds do not establish equivalence between conditions or a universal initialization law.

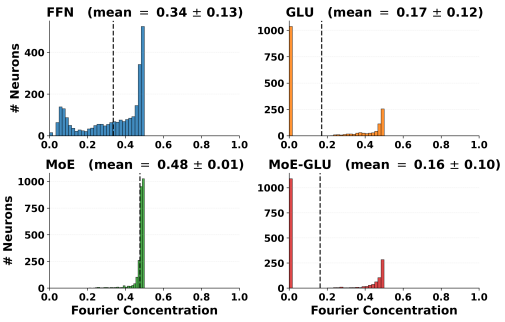
I GLU AS A SECONDARY REPRESENTATION RESULT

GLU does not reproduce the MoE component-reliance signature, but it changes how modular-addition structure aligns with individual neuron coordinates. At matched widths, mean per-neuron Fourier concentration falls from 0.34 ± 0.13 for dense FFNs to 0.17 ± 0.12 for GLUs, while the strongest principal component is at least as Fourier-aligned (0.41 versus 0.49) and linear probes recover the

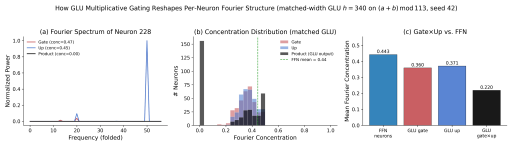
task perfectly from both activation vectors. This is consistent with preserved structure distributed in a different coordinate basis rather than absent task information. It establishes architecture-dependent basis alignment on modular addition; it does not causally prove that multiplicative gating produces the change or establish generality beyond this task.

Table 35: Secondary modular-addition representation summary at total-parameter-matched widths (five seeds). Probe accuracy is from the full activation vector.

Architecture	Per-neuron Fourier concentration	Top-PC concentration	Linear probe
Dense FFN	0.34 ± 0.13	0.41	100%
GLU	0.17 ± 0.12	0.49	100%



(a) Per-neuron Fourier concentration.



(b) GLU gate/product decomposition.

Figure 14: Secondary GLU representation evidence. These analyses concern basis alignment, not the primary MoE reliance claim.

J GROKKING AND OPTIMIZATION-SIDE CONTROLS

The modular-addition runs also show that FFN design changes whole-model learning dynamics. We recover the first saved checkpoint exceeding 99% held-out accuracy, not the epoch of the best-accuracy checkpoint. The budget is 40,000 epochs. Models use full-batch AdamW training, no normalization, and no dropout except in the named dropout control. Baseline and narrow runs are evaluated every 100 epochs; top-2 and dropout runs every 500. Times therefore have this resolution and summarize only observed crossings.

Table 36: Archived modular-addition threshold crossings. The expanded dense/MoE groups include the original five seeds; other controls use those five seeds. "No crossing" requires a 40,000-epoch archive confirming that the historical best remains at or below 99%. Incomplete archives are kept separate. Medians condition on observed crossings.

Variant	n	Crossed	No crossing	Unverified	Median epoch
Dense FFN	20	12	6	2	23,150
MoE top-1	20	20	0	0	8,400
MoE top-2	5	0	5	0	—
Narrow dense	5	4	1	0	28,900
Dense, dropout 0.3	5	5	0	0	2,000

The original five-seed dense and top-1 groups have three and five recorded crossings, with conditional median epochs 22,500 and 8,500. Narrow dense reaches the threshold later among its successful seeds, whereas top-1 MoE reaches it earlier; dropout is faster still. Two expanded dense runs lack evidence of completion to 40,000 epochs, so their status at the cutoff remains unknown. Conditional medians do not estimate the time to generalization across all seeds, and non-crossings within the stated budget do not establish permanent failure to generalize.

These observations support an effect of architecture on optimization, not a measured race between attention and FFN learning. Top-2 changes active width, expert mixing, and router task gradients together; dropout supplies another route to faster grokking. Modular addition shows faster grokking but weaker redistribution than add-7, whose shared training/evaluation support does not test delayed held-out generalization. A component-learning mechanism requires measurements and interventions beyond these whole-model timing comparisons.

K VISUALIZING RELIANCE DURING TRAINING

These figures follow saved one-layer dense and four-expert MoE checkpoints through training. They use one illustrative seed pair (seed 42), chosen for its relatively dense checkpoint archive. They complement the multi-seed results rather than estimate variability across runs. An intact model uses its original components and readout; no-FFN and no-attention accuracy use that same readout after zeroing the indicated component output. No models or readouts are trained for these visualizations.

K.1 MODULAR ADDITION: EARLY, MIDDLE, AND LATE CHECKPOINTS

Figure 15 separates learning the task from the performance that survives component removal. At 7,400 epochs, MoE has begun to generalize, but little performance survives FFN removal. At 10,000 epochs, its intact accuracy is nearly perfect and no-FFN accuracy reaches 19.7%. By 40,000 epochs, both architectures achieve nearly perfect intact accuracy, yet MoE retains 17.3% without its FFN versus 2.4% for dense. Thus, similar final task performance can accompany different learned reliance. The MoE no-FFN trajectory fluctuates after generalization; it does not increase monotonically.

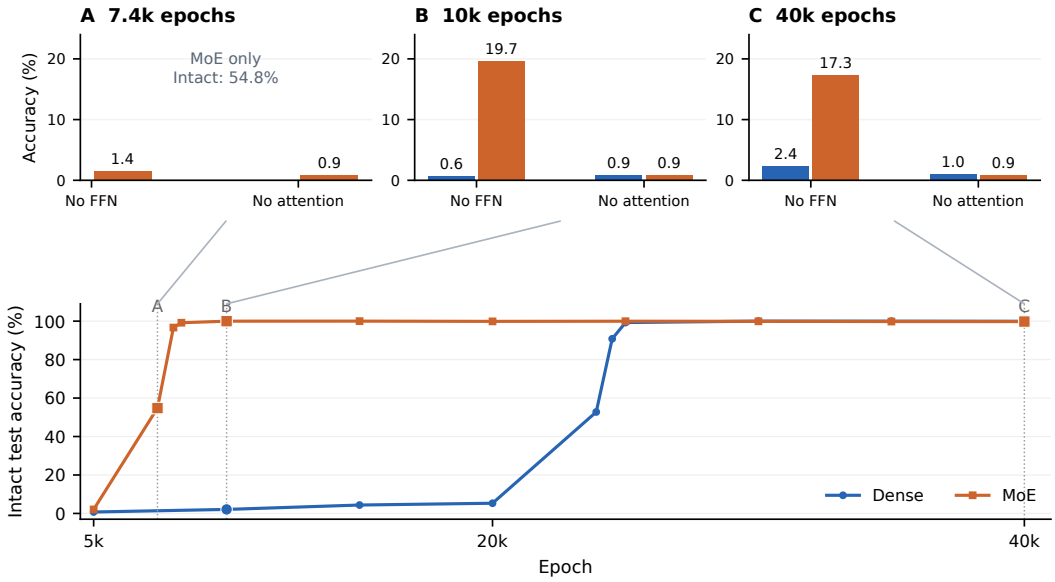


Figure 15: **Task learning and component reliance at three training times.** The lower panel shows intact modular-addition test accuracy; the upper panels show accuracy after FFN or attention removal at the marked checkpoints. Colors identify dense and MoE throughout. The first inset contains MoE only: no dense checkpoint is available at 7,400 epochs, so no matched dense value is inferred. All bars share the same scale and report accuracy, not additive shares of a solution. One seed pair; lines connect saved observations.

The comparison at 10,000 epochs includes models with different intact accuracy; the late comparison holds intact performance approximately matched. No-attention accuracy stays near 1%: in this one-layer task, the prediction position contains the same separator token for every example. Removing attention removes its access to the operands, leaving a constant prediction. This floor is a task-structure check, not evidence of a dense–MoE difference in attention's role.

Evaluation protocol. The task is $(a+b) \bmod 113$, with inputs $[a, b, =]$. We reconstruct the original seeded split of all 12,769 ordered pairs into 3,830 training and 8,939 test examples and retain all 114 output classes, including the separator. Both models have width 128, four attention heads, GELU, untied readouts, noncausal attention, and no normalization. Dense hidden width is 512; MoE has four width-128 experts with hard top-1 routing and selected gate renormalized to one. Original training uses full-batch AdamW, learning rate 10^{-3}, weight decay 1.0, and a balancing-loss coefficient of 0.01 for MoE. The archive supplies ten distinct dense and eleven MoE snapshots, beginning at epoch

5,000. We use fixed-time snapshots and saved accuracy milestones, not selected-best substitutes. The observed trajectories therefore describe this training and regularization regime.

K.2 Comparing the three algorithmic tasks

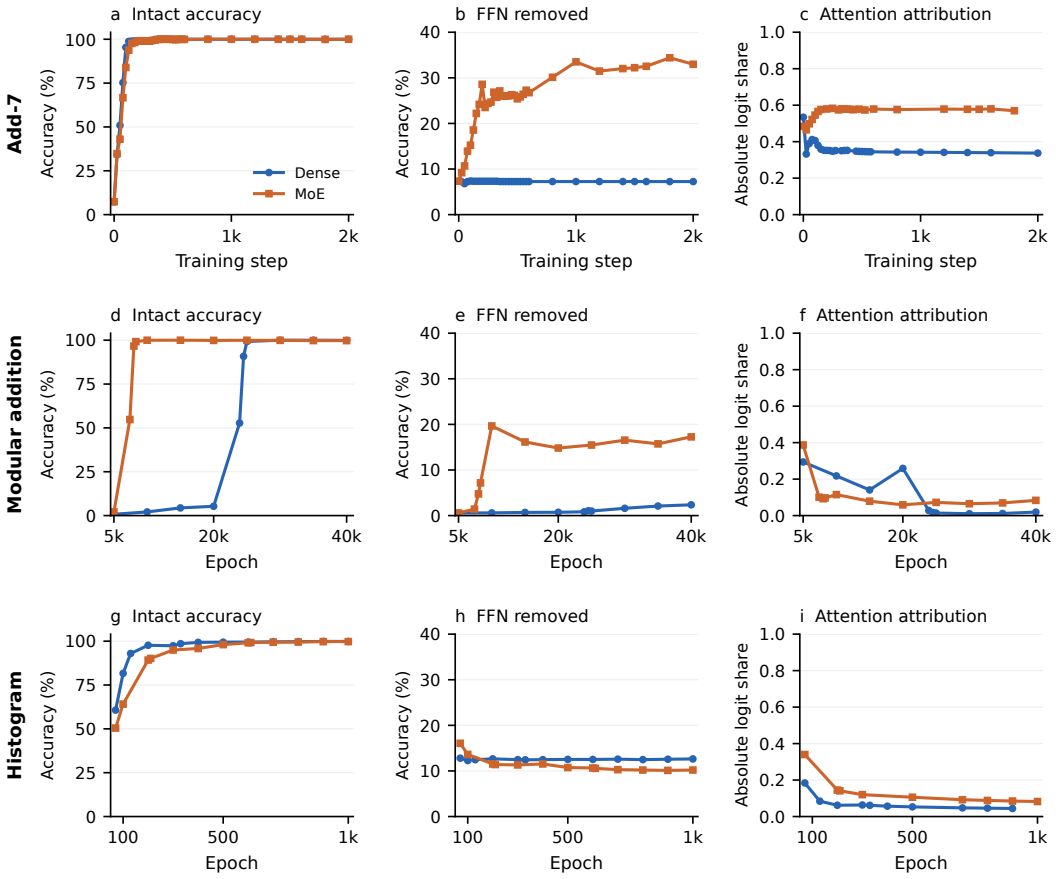


Figure 16: **Task success, FFN-removal survival, and intact attribution follow different trajectories.** Rows show add-7, modular addition, and histogram; columns show intact accuracy, no-FFN accuracy, and attention's absolute correct-logit share in the intact model. Each row uses one dense/MoE seed pair and a fixed evaluation set across training. Add-7 uses exhaustive finite training support; modular addition uses its original held-out split; histogram uses a fixed diagnostic sample. Markers are observations, not independent seeds. Attribution uses available checkpoint weights, so its coverage can be shorter than the cached accuracy curves.

On add-7, a no-FFN gap emerges while both models learn the task and remains after intact accuracy reaches 100%. In modular addition, the saved checkpoints show delayed generalization and a later no-FFN advantage. Histogram reaches high intact accuracy without a sustained MoE no-FFN advantage in this pair. At the final displayed times, dense/MoE no-FFN accuracies are $7.25/32.98\%$ on add-7, $2.36/17.26\%$ on modular addition, and $12.63/10.20\%$ on histogram.

The attribution curves provide a complementary measurement. Attention's absolute share falls during modular-addition generalization even as no-FFN accuracy improves. Histogram also shows that a larger relative attention attribution need not produce better FFN-removal performance. These measures answer different questions: one describes relative logit contributions with both components present; the other tests predictions after removing a component.

Measurement and coverage. For each correct-answer logit, we compute $|\ell_{\text{attn}}|/(|\ell_{\text{attn}}| + |\ell_{\text{FFN}}|)$ using the original unembedding, then average over evaluated predictions. This excludes the direct

embedding/position contribution and is an absolute attribution ratio, not a fraction of computation. Add-7 accuracy uses all 1,000 inputs and 4,000 teacher-forced output digits. Histogram uses the existing diagnostic split of 600 generated sequences and 6,000 predictions; overlap with all refreshed training samples has not been exhaustively audited. Add-7 has 33 accuracy observations per architecture and histogram has 13. Newly evaluated attribution uses 27/24 dense/MoE add-7 checkpoints and 10/9 histogram checkpoints with locally available weights. In particular, MoE add-7 attribution ends at step 1,800 and dense histogram attribution at epoch 900; missing values are not extrapolated. Within each task, reconstructed logits and zero-FFN outputs are checked against the model forward pass and an output-ablation hook. Across tasks, width, normalization, readout tying, training data, and regularization differ, so these comparisons do not isolate task identity.

K.3 How these trajectories relate to depth

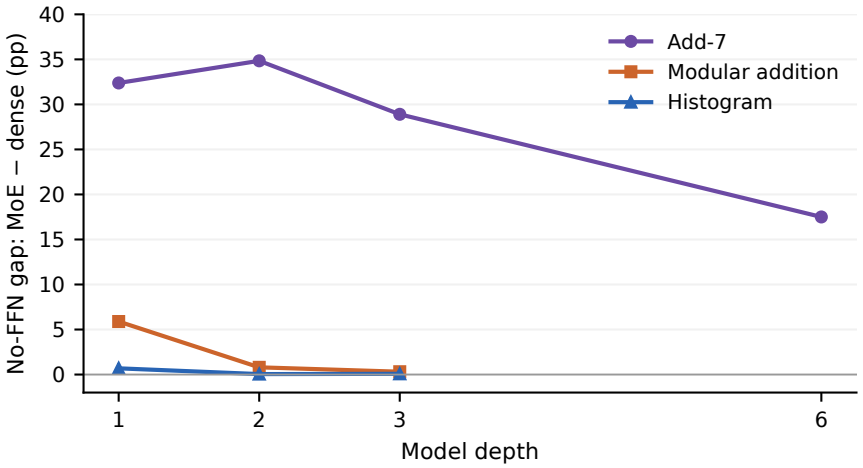


Figure 17: **The depth pattern differs across the algorithmic tasks.** Points reproduce the multi-seed mean MoE–dense no-FFN gaps in Tables 21, 22, and 23, after removing every FFN. Add-7 retains a positive gap through six layers; modular addition's gap is below one percentage point at two and three layers; histogram's is small at all three tested depths. This is a summary of existing endpoint experiments, not the single-seed training trajectories above. Add-7 and histogram use five seeds per cell. Modular addition conditions on convergence and has unequal counts and checkpoint conventions, specified with the depth results. Uncertainty and per-architecture scores are reported in the referenced tables. No six-layer modular-addition or histogram point is implied.

The clear empirical distinction is persistence, not universal transfer: the add-7 advantage remains substantial with depth, whereas the tested deeper modular-addition models and histogram models have little separation after complete FFN removal. The one-layer modular-addition trajectories additionally show that the endpoint matters. Its late-training gap should therefore not be compared to the depth grid as if training stage were matched across all cells.

A useful interpretation is that solving a task does not determine how the jointly trained representation, FFN, and original readout support that solution. The fitted-readout experiments provide a separate check of this distinction: useful information can be recovered upstream of the FFN even when the original model performs poorly without it. The figures make the task, training-time, and depth dependence visible; establishing why greater depth specifically attenuates the modular-addition gap would require matched training trajectories and interventions in those deeper models.

K.4 Fourier progress measures for modular addition

Following the progress-measure approach of Nanda et al. (2023), we examine restricted and excluded Fourier losses, Fourier concentration, and squared parameter norm over the same modular-addition checkpoints (Fig. 18). Both runs already fit the training split at the first available snapshot. Later generalization accompanies greater Fourier concentration, and the selected frequency terms become

important for training predictions. This transition occurs earlier in the MoE run. The dotted lines mark the first saved checkpoints above 99% test accuracy, not inferred mechanistic phase boundaries.

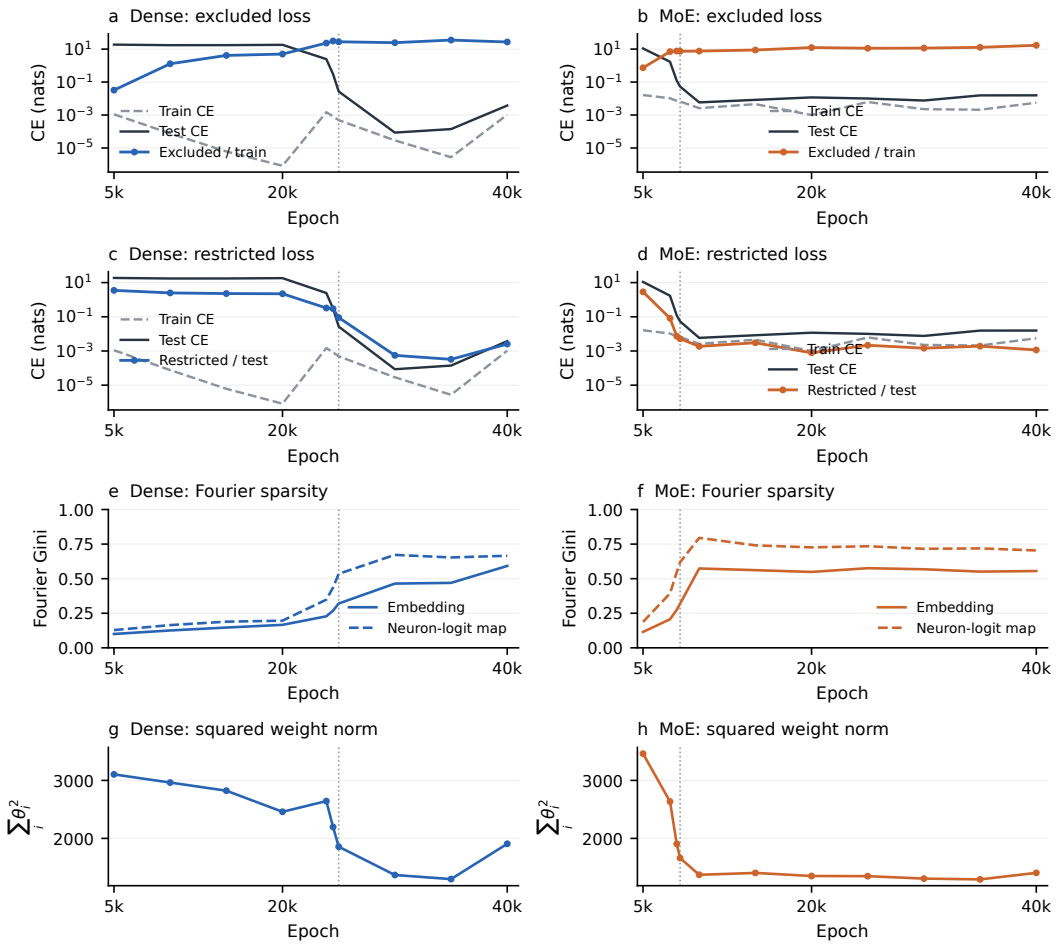


Figure 18: **Fourier progress measures, adapted for dense and MoE modular addition.** Columns show dense and MoE (seed 42). The top rows compare ordinary train/test cross-entropy with excluded loss on training examples and restricted loss on test examples. The lower rows show concentration in a real Fourier basis and total squared parameter norm. Each architecture's frequency set is selected from its final checkpoint and held fixed across time. Restricted and excluded losses are retrospective projections of the full input-grid logits, not independently trained models. Vertical dotted lines mark the first saved $> 99\%$ test-accuracy checkpoint: 25,000 epochs for dense and 8,300 for MoE.

Selecting frequencies. At epoch 40,000, we project the neuron-to-logit map onto an orthonormal real Fourier basis over the 113 answer tokens, group sine/cosine squared energy by frequency, and select the smallest ranked set covering at least 95% of nonconstant energy. The sets are $\{1, 15, 19, 37, 38, 39, 56\}$ for dense and $\{7, 20, 41\}$ for MoE. For MoE, the map concatenates the experts' down-projections through the unembedding; it describes all expert neurons, not their input-dependent routing. Biases and router parameters are excluded from this map. Selecting frequencies from final weights deliberately uses future information for retrospective analysis; the different set sizes also prevent a capacity-matched interpretation of the filtered losses.

Restricted and excluded losses. We evaluate logits on the full (a, b) grid and transform the two input axes. For each selected frequency k, the restricted logits retain the four $(\pm k, \pm k)$ coefficients, together with the constant term; excluded logits remove these nonconstant blocks while retaining the constant. We score restricted logits on the original test split and excluded logits on the training split. This projection pools predictions across the input grid. For example, at epoch 7,400, MoE

restricted-logit test accuracy is 99.92% while ordinary accuracy is 54.78%; dense shows 90.33% versus 52.76% at epoch 23,900. These comparisons expose structure in the model's outputs, not performance available through its ordinary pointwise inference rule.

Concentration, norm, and checks. Fourier Gini is the Gini coefficient of real Fourier coefficient-row norms, including the constant, for the residue-token embeddings and neuron-to-logit map. Squared weight norm sums all parameter squares, including biases and the router. The analysis checks basis orthogonality, energy conservation, synthetic filtering, and logit reconstruction; saved results also include 90% and 99% frequency-selection thresholds, which preserve the qualitative early-to-late loss changes. These are an adaptation of the progress measures, rather than a claim that the same circuit or discrete training stages have been identified in both architectures.

L Artifact and reproducibility manifest

The supplementary archive contains the exact configurations and saved per-seed summaries used for the main results. The result-file mapping is:

- expert-count sweep: `analysis/add7_esweep_results.txt`;
- no-balancing, common-update sweep: `analysis/refine_noaux_esweep_20260920/`, including the recorded protocol, training and independent validation scripts, all 25 final checkpoints, 500-update snapshots, and per-seed summaries. The summary, protocol, and independent rescoring record are also included as `assets_iclr2027/noaux_matched_sweep_20260920*.json`;
- expert-sweep checkpoint and attribution audit: `analysis/refine_dla_checks_20260920.py` and `analysis/refine_dla_checks_20260920/`, including selected update counts, checkpoint hashes, centered and margin attribution with and without attention output bias, reconstruction tests, and an auxiliary-only upstream-gradient check. The complete result is also included as `assets_iclr2027/dla_contrast_checks_20260920.json`;
- route replay, hashing, shared expert, and trajectories: `analysis/rebuttal_route_replay_results_5seed.txt`, `analysis/rebuttal_hash_routing_results.txt`, `analysis/rebuttal_shared_expert_results.txt`, and `analysis/rebuttal_route_stability_*_results.txt`;
- fixed-budget replay replication and FFN-variability study: `analysis/ffn_reliability_training_20260914_protocol.md`, `analysis/ffn_reliability_training_20260914_results/`, and `analysis/ffn_reliability_report_20260914_results/`. The report includes per-seed final and earliest-best endpoints, trajectories, initialization and batch-stream checks, and independent rescoring of all 70 retained checkpoints;
- weighted top-2 membership replay: `analysis/top2_route_replay_20260914_protocol.md`, `analysis/top2_route_replay_20260914.py`, and `analysis/top2_route_replay_20260914_results/`; independent checkpoint rescoring and paired summaries are in `analysis/top2_route_replay_20260914_validation/verification.json` and `analysis/top2_route_replay_20260914_validation/results.txt`;
- gated top-1: `analysis/gated_top1_20260924/`, including the recorded protocol, all 25 runs, training and independent rescoring scripts, per-seed results, and trajectories. The summary, protocol, and verification record are also included as `assets_iclr2027/gated_top1_20260924*.json`;
- graded and mean interventions: `analysis/softer_ablation_*_results.txt`;
- depth and width: `analysis/add7_2layer_results.txt`, `analysis/add7_3layer_results.txt`, `analysis/add7_6layer_results.txt`, `analysis/cross_task_depth_results.json`, and `analysis/ablation_width_sweep_*_results.txt`;

- active-width matching: `analysis/flop_matched_ablation_add7_results.txt`;
- head counterfactuals and routing association: `analysis/head_counterfactual_add7_results.txt` and `analysis/routing_mi_randroute_results.txt`;
- position-resolved headline: `analysis/sweep_position_headline_20260914/fig_sweep_position_headline.json`, including per-seed scores and checkpoint hashes;
- intact expert-sweep DLA: `analysis/dla_expert_sweep_20260914/results.json`, including the existing tokenwise attribution definition, per-seed values, signed contributions, shape checks, and reconstruction validation; the headline overlay uses `analysis/make_sweep_position_dla_headline_20260914.py`;
- descriptive log-odds fits: `analysis/fitted_dla_scaling_20260914/results.json`, including parameter-count comparisons, seedwise coefficients, residuals, curvature diagnostics, and leave-one-expert-count-out errors;
- control-ladder visualization: `analysis/control_ladder_20260914.py`, with per-checkpoint scores and sample s.d. in `assets_iclr2027/fig_control_ladder.json`;
- grokking controls: `analysis/grokking_controls_20260914/results.json`, including per-seed threshold milestones, archive-completeness status, and conditional timing summaries;
- training-trajectory visualizations: `analysis/modadd_nanda_progress_s42_20260920/`, `analysis/algorithmic_training_comparison_s42_20260920/`, and `analysis/modadd_snapshot_story_s42_20260920/`, each with measurements and evaluation notes. The corresponding analysis scripts are `analysis/modadd_nanda_progress.py`, `analysis/compare_algorithmic_training.py`, and `analysis/make_modadd_snapshot_story.py`; appendix layouts are generated by `analysis/make_training_paper_figures.py`;
- output-head training controls: `analysis/add7_readout_constraint_20260909/` and `analysis/add7_readout_constraint_scale_20260909/`; the common-split analyses are in `analysis/add7_readout_constraint_analysis_v2_20260909/` and `analysis/add7_readout_constraint_scale_analysis_v2_20260909/`. A compact extraction with source and checkpoint hashes, per-seed scores, and recomputed sample s.d. is included as `assets_iclr2027/readout_initialization_results.json`;
- paired operation alignment: `analysis/paired_route_alignment_20260914/results.json`; evaluation-time fixed-route head removal: `analysis/fixed_routes_head_20260914/results.json`. Both directories include protocols, per-seed outputs, and saved assignments;
- frozen-FFN replacement: per-checkpoint records in `analysis/ffn_readout_mechanism_20260908/`, `analysis/histogram_token_affine_results_20260909/`, and `analysis/histogram_shared_slope_results_20260909/`. Figure 13 uses the checked extraction in `analysis/make_task_ffn_replacement_20260914.py`.

Before submission, all absolute paths, tracking-service identifiers, checkpoint metadata, and repository history in the archive should be checked for author identity. Standard-deviation conventions and the explicitly sample-based summaries are specified in Appendix B and the corresponding captions.