Cross-Step Activation Mixing Has a Depth-Dependent Design Tradeoff: Preliminary Results on Multi-Digit Addition

[Author redacted]

May 4, 2026

Note on authorship and scope. This writeup was drafted by an AI language model (Claude) from numerical results, code, and design discussions provided by the author. All experiments were run by the author; the AI's role was limited to prose, formal description, and formatting. The author reviewed and approved the final text. This is a *preliminary* report on a small-scale study; it is not the resulting writeup of a completed project. Ablations are partial, all runs are single-seed, and the model and task are toy. The findings should be read accordingly.

Abstract

We study a thin cross-step side channel added to small autoregressive transformers, in which each block reads a learned, gated, content-conditioned combination of the previous generation step's layer outputs. We find that the channel's effect on multi-digit addition depends jointly on model depth and on the expressiveness of the channel's routing. At four layers, a minimal static α-weighted variant produces large gains over a vanilla baseline (3-digit accuracy $0.586 \to 0.898$), while a more expressive attention-style dynamic variant gives little benefit. At eight layers, the static channel is absorbed by depth and stops helping; the dynamic channel instead drives near-perfect in-distribution accuracy across difficulty levels (4-digit accuracy $0.088 \to 0.962$, 6-digit accuracy $0.132 \to 0.890$ versus a same-depth vanilla baseline). A two-pass training-time approximation of the channel underperforms vanilla in all conditions. Length generalization to one digit beyond the training range remains absent. We report this as a preliminary but suggestive finding: the channel and depth interact, and the right channel design at the right depth substantially shifts what a small transformer can learn on a structured task.

1 Introduction

A standard autoregressive transformer has no direct path along which hidden state can flow from one generation step to the next. When generating token t given tokens $0, \ldots, t-1$, every internal representation it computes is fresh; previous-token information enters the current step only through self-attention's read of the cached key-value tensors. There is no path by which, say, layer 3's representation of token $t-1$ can directly influence layer 1's representation of token t. Such paths are routine in recurrent networks, where the state at time t is a direct function of the state at time $t-1$.

For multi-digit addition this is restrictive. Carries propagate from least- to most-significant digit, so the answer at digit k depends on resolutions of digits $k-1, k-2, \ldots$. A vanilla transformer must perform every such resolution within the single forward pass over its current token, even though the relevant intermediate state was computed by the previous pass and then discarded.

We study a simple architectural modification that opens a thin cross-step channel: each block, in addition to its residual stream input, receives a learned, gated linear combination of cached layer outputs from the previous generation step. The combination is taken across source depth (the block chooses which depth of the previous step to read from) and is content-gated by the current token. The mechanism is one step deep — it does not generalize to a full RNN-style recurrence — and the cross-step signal is linear and detached. The point of interest is not its expressive power in isolation but how it interacts with model depth and with the choice of routing within the channel.

We compare two source approximations at training time (since teacher forcing collapses the autoregressive process into a single forward), two cache modes, two routing flavors, and two injection points. The empirical question we report on here is narrower than this design space: *does the channel help, and does its effect depend on depth and on the routing flavor?*

The headline finding is that the answer to both is yes, in opposite directions. A cheap static-routing variant of the channel produces large gains at four layers and is absorbed by depth at eight; an expressive dynamic-routing variant gives little benefit at four layers and dominates at eight. The naive prediction — that more expressiveness always helps, or that more depth obviates the channel — holds for neither.

Disclosure. A prior writeup by the same author and AI assistant described an earlier version of this work in which the single-pass "causal-in-depth" variant contained a bug: it read source layers at the *same* sequence position rather than at the previous position, so the mechanism was within-step depth mixing rather than cross-step. The bug was identified and fixed during the development of this writeup. All results reported here use the corrected implementation. We mention this for honesty about the project's history; the present results stand on their own.

2 Method

2.1 Pre-norm transformer block

We use a standard pre-norm GPT block as the baseline:

$$x \leftarrow x + \mathrm{Attn}(\mathrm{LN}_1(x)), \tag{1}$$

$$x \leftarrow x + \mathrm{MLP}(\mathrm{LN}_2(x)). \tag{2}$$

2.2 Cross-step side channel

Throughout, L is the number of blocks, D is the model dimension, T is sequence length, and $H^{(l)} \in \mathbb{R}^{B\times T\times D}$ denotes layer l's output. We use "block" and "layer" interchangeably.

The mix block at sequence position t receives, in addition to x_t, a side input drawn from a stack of source representations $H^{(l)}$ shifted one position right along T, so position t reads source l at position $t-1$. The four-stage construction below produces a single $\mathrm{inj}_t \in \mathbb{R}^D$ that is added to the layer-normed residual stream just before attention.

Stage 1 — per-source-layer normalization. Each source is normalized with its own learned RMSNorm gain $\gamma^{(l)} \in \mathbb{R}^D$:

$$\tilde{H}^{(l)} = \gamma^{(l)} \odot \frac{H^{(l)}}{\sqrt{\frac{1}{D}\sum_{d=1}^{D}(H_d^{(l)})^2 + \varepsilon}}. \tag{3}$$

In pre-norm transformers the residual stream's norm grows with depth, so deep sources naturally dominate shallow ones in raw magnitude; the per-source gain absorbs this so that routing weights below express genuine importance.

Stage 2 — routing across source depths. Two flavors of routing.

Static. A learned scalar α_l per source layer combines the normalized sources into a single mix vector:

$$m = \sum_l \alpha_l \, \tilde{H}^{(l)} \in \mathbb{R}^{B \times T \times D}. \tag{4}$$

The mix vector then feeds a content gate g, computed from a projection $W_k m$ and the layer-normed residual stream, and a content path $W_q m$; the side input is the gated content $\text{inj} = g \odot W_q m$.

Dynamic. An attention-style per-source readout. For each source l,

$$k^{(l)} = \text{RMSNorm}^{(l)}(H^{(l)}), \tag{5}$$

$$q^{(l)} = \text{RMSNorm}_q\Big(W_q^{(l)} \, \text{LN}_1(x)\Big), \tag{6}$$

$$v^{(l)} = W_k^{(l)} \, k^{(l)}, \tag{7}$$

$$\text{gate}^{(l)} = \sigma\Big(q^{(l)} \odot k^{(l)}\Big), \tag{8}$$

$$\text{contrib}^{(l)} = \text{gate}^{(l)} \odot v^{(l)}, \tag{9}$$

and the side input is $\text{inj} = \sum_l \text{contrib}^{(l)}$. We use per-source projections $W_q^{(l)}, W_k^{(l)}$, giving the dynamic flavor more parameters than static.

Stage 3 — injection. We zero-initialize the final projection in each flavor, so $\text{inj} = 0$ at the start of training. The model begins as a vanilla GPT and learns to use the channel from there.

Modified block.

$$x \leftarrow x + \text{Attn}(\text{LN}_1(x) + \text{inj}), \tag{10}$$

$$x \leftarrow x + \text{MLP}(\text{LN}_2(x)). \tag{11}$$

2.3 Approximating the source at training time

At inference, the source representations are simply the cached layer outputs of the previous generation step. Under teacher forcing, no such "previous step" exists and the channel must be approximated. We compare two approximations.

Two-pass. A no-gradient vanilla forward is run, producing layer outputs $H^{(0..L-1)}$. These are stacked, shifted one position right along T, and exposed to every mix block in a second forward pass through which gradients flow. Every block sees all L source layers. The training-time source is vanilla; the inference-time source is mix-aware. This distribution gap is the cost of the approximation.

Causal-in-depth. A single forward is run. As block i produces its output, that output is detached, shifted one position right along T, and added to a stack visible to subsequent blocks. Block i sees source layers $0, \ldots, i-1$ at position $t-1$. Block 0 has no source. The training and inference paths are structurally identical, but each block can read only from depths below its own.

Both approximations respect the autoregressive causal mask. Both yield single-pass inference. The two-pass approximation has strictly greater nominal connectivity (every block sees every source layer); the causal-in-depth approximation has no train/inference distribution gap.

3 Experimental Setup

Task. Operands a, b are sampled with digit counts drawn uniformly from $\{1, 2, 3, 4, 5, 6\}$; the model is trained to produce the string `a+b=`$a+b$ with BOS/EOS tokens. We evaluate on seven fixed digit-count buckets: 1 through 6 digits in-distribution and 7 digits as a held-out bucket. Each bucket has 500 samples; an output is correct iff the post-`=` string exactly matches the ground truth.

Architecture. All models share $D = 64$, 4 attention heads, MLP expansion 4, dropout 0.1, block size 28, and a 15-token vocabulary (digits, `+`, `=`, BOS, EOS, PAD). Weight tying is disabled. We train at both $L = 4$ and $L = 8$ to probe the depth dependence. Vanilla parameter counts are 203,776 at $L = 4$ and 403,712 at $L = 8$. Mix variants add at most $\sim$58% on top at $L = 8$ (Tables 1 and 2).

Training. AdamW with learning rate 3×10^{-4}, weight decay 0.01, gradient clipping at 1.0, cosine schedule, batch size 256, 100,000 training samples, 40 epochs. The same training loader and validation sets are used across all runs of a given depth.

Models compared. For each depth $L \in \{4, 8\}$, we report four configurations:

- `vanilla`: standard GPT, no mixing.
- `twopass_static`: two-pass approximation, static routing, scalar gate, static α, shared norm.
- `causal_static`: causal-in-depth, static routing, scalar gate, static α, shared norm.
- `causal_dynamic`: causal-in-depth, dynamic routing, elementwise gate, per-source projections, shared norm.

For completeness we also ran `twopass_dynamic`; it underperforms `twopass_static` at both depths and we report it in the tables but do not analyze it separately. All runs use `cache_mode=shift` (literal $t - 1$ read; we did not run the EMA-cache variant in this set of experiments). All runs are single-seed.

4 Results

4.1 Four layers: static wins, dynamic loses

At $L = 4$, `causal_static` produces large gains over vanilla on 2-digit (+3.0), 3-digit (+31.2), and 5-digit (+7.6) buckets and matches vanilla on 4-digit. The 3-digit gain is the most striking: from 0.586 to 0.898, a 1.53× accuracy ratio. The mix variant adds only ˜25k parameters ($\sim$12.5% over vanilla), so the gain is not attributable to capacity alone.

The dynamic variant at $L = 4$ produces smaller and less consistent gains: +3.8 on 2-digit, +6.2 on 3-digit, +12.4 on 5-digit, but a regression on 4-digit (0.450 → 0.088 against the cell-2 vanilla; 0.468 → 0.088 against cell-1 vanilla). The simpler routing dominates the more expressive one in this regime.

The two-pass approximation underperforms vanilla on every difficulty bucket, in both flavors. The static variant's α parameters *decrease* during training (read from the W&B logs) — the

Model	Params	1d	2d	3d	4d	5d	6d	7d (OOD)
vanilla	203,776	1.000	0.944	0.586	0.468	0.424	0.018	0.000
twopass_static	238,352	1.000	0.742	0.192	0.040	0.120	0.002	0.000
twopass_dynamic	338,176	1.000	0.504	0.064	0.000	0.058	0.058	0.000
causal_static	229,318	1.000	**0.974**	**0.898**	0.462	**0.500**	0.014	0.000
causal_dynamic	254,272	1.000	**0.982**	0.648	0.088	**0.582**	0.024	0.000

Table 1: Per-bucket accuracy at $L = 4$, 500 samples per bucket, after 40 epochs. Best (or tied-best) per column shown in bold, comparing only against the vanilla baseline. Both vanilla baselines reported in the corresponding sweep cells differ by less than 3 points on each bucket; we use the cell-1 vanilla here since it shares a sweep with both static-flavor mix variants.

optimizer is suppressing the side channel because the source distribution it sees at training does not match what the model would receive at inference. We do not have a direct probe of this hypothesis but the diagnostic is consistent with it.

4.2 Eight layers: static loses, dynamic dominates

Model	Params	1d	2d	3d	4d	5d	6d	7d (OOD)
vanilla	403,712	1.000	0.988	**0.910**	0.792	0.752	0.226	0.000
twopass_static	474,944	0.986	0.742	0.134	0.016	0.000	0.134	0.000
twopass_dynamic	940,800	0.696	0.228	0.110	0.006	0.000	0.002	0.000
causal_static	464,220	1.000	0.972	0.740	0.268	0.090	0.006	0.000
causal_dynamic	638,912	1.000	**1.000**	**0.974**	**0.962**	**0.950**	**0.890**	0.000

Table 2: Per-bucket accuracy at $L = 8$, 500 samples per bucket, after 40 epochs. Best per column in bold. Two distinct vanilla runs are reported in the underlying sweep cells; for the static row we compare against cell-1 vanilla (shown), for the dynamic row against cell-2 vanilla (0.088 on 4-digit, 0.132 on 6-digit). Cell-2 vanilla is anomalously poor on 4-digit; we treat this as evidence of high seed variance on the harder buckets, discussed in §6.

The picture inverts. Vanilla at $L = 8$ is far stronger than at $L = 4$: 3-digit accuracy rises from 0.586 to 0.910, 4-digit from 0.468 to 0.792, 5-digit from 0.424 to 0.752. The vanilla baseline absorbs most of the difficulty on its own.

The static mix variant that won at $L = 4$ now *loses* to vanilla on 3-digit (0.740 vs 0.910), 4-digit (0.268 vs 0.792), and 5-digit (0.090 vs 0.752). The static α parameters in this run *decrease* monotonically during training (logs show `block1_alpha0` going from ~ 0.1 to -0.1), the same suppression pattern observed for two-pass at $L = 4$. The optimizer has decided the simple side channel is not worth using at this depth.

The dynamic mix variant produces near-perfect in-distribution accuracy: 0.974 at 3-digit, 0.962 at 4-digit, 0.950 at 5-digit, and 0.890 at 6-digit where the same-depth vanilla is at 0.226 (cell 1) or 0.132 (cell 2). The gap on 6-digit is between 6.6× and 4.0×; the gap on 4-digit is between 1.2× and 11× depending on which vanilla baseline is used. Train loss reaches 1.358, the lowest of any run in the sweep.

The dynamic variant adds more parameters than the static variant ($+235k$ vs $+60k$ at $L = 8$), so capacity is a confound; we discuss this in §6.

4.3 The flavor-by-depth interaction

The clearest single artifact of this study is the interaction between channel flavor and model depth, summarized in Table 3.

	$L = 4$		$L = 8$	
Bucket	vanilla	causal_static	vanilla	causal_dynamic
3d	0.586	**0.898**	0.910	**0.974**
4d	0.468	0.462	0.792	**0.962**
5d	0.424	**0.500**	0.752	**0.950**
6d	0.018	0.014	0.226	**0.890**

Table 3: The flavor-by-depth interaction. The static channel helps at $L = 4$ and stops helping at $L = 8$; the dynamic channel helps little at $L = 4$ and dominates at $L = 8$. Bold entries are accuracies meaningfully above the corresponding vanilla baseline (gap ≥ 0.05).

The static channel's relative advantage over vanilla on 3-digit collapses from +31 points at $L = 4$ to −17 points at $L = 8$. The dynamic channel's relative advantage on 4-digit rises from −38 points (against cell-1 vanilla) at $L = 4$ to +17 points (against cell-1 vanilla; +87 against cell-2 vanilla) at $L = 8$. Adding depth helps the dynamic channel substantially more than it helps the static channel or vanilla.

4.4 Length generalization

No model in either depth tier achieves non-zero accuracy on the 7-digit OOD bucket. The mix mechanism does not enable generalization beyond the training digit range. The 6-digit in-distribution bucket is also at floor or near floor for all models except the $L = 8$ dynamic variant, suggesting that 6-digit is at the ragged edge of what 100k samples and 40 epochs can teach this architecture, and that the dynamic channel at $L = 8$ is the only configuration that crosses that edge.

5 Discussion

The most parsimonious reading of the data is that the cross-step channel and model depth are partly substitutes and partly complements, and which one applies depends on how the channel routes information across source depth.

The static channel substitutes for depth. A simple α-weighted sum of previous-step layer outputs gives the model a way to inject a single time-shifted snapshot of its own recent computation. At $L = 4$, this is genuinely useful — the vanilla model lacks the depth to work through multi-digit carries within a single forward, and the channel provides a shortcut. At $L = 8$, the vanilla model has the depth to do this work itself, the channel's content becomes redundant, and the optimizer turns it off (the α values shrink during training). This is consistent with one of the explanations we considered prior to running the depth sweep: "depth substitutes for the cross-step channel."

The dynamic channel complements depth. An attention-style per-source-layer readout is qualitatively different. Rather than producing one snapshot, it lets the model issue a per-channel content-conditioned query against each source layer's representation. At $L = 4$ this routing is either too expressive to learn from 100k examples or simply not worth the parameters; the gains are small

and inconsistent. At $L = 8$ the same mechanism produces near-perfect in-distribution accuracy across all difficulty buckets. Whatever cross-step information the dynamic channel extracts, it pays off only when the model has enough depth to integrate it.

Capacity vs. mechanism. The dynamic variant adds substantially more parameters than the static variant, which is a confound for the interaction. We cannot, from these experiments alone, say what fraction of the dynamic-variant gain at $L = 8$ is cross-step communication versus generic capacity. A controlled experiment scaling the static variant's mix capacity to match the dynamic's — by widening its W_q, W_k projections, or by using per-source projections in the static formulation — would discriminate these. We have not yet run that experiment.

Two-pass underperforms in all conditions. The two-pass approximation, despite its strictly greater connectivity (every block sees every source layer), produces accuracy below vanilla at both depths and in both flavors. The internal diagnostics show α shrinking during training in the static variant, suggesting the optimizer suppresses the channel rather than learning to use it. The simplest explanation is the train/inference distribution gap intrinsic to the two-pass design: the training-time source is a no-gradient vanilla forward while the inference-time source is the mix-aware activation from the previous generation step, and the model cannot bridge the two distributions in this training budget. The causal-in-depth design avoids this gap by construction.

Length generalization is unaffected. No condition produces non-zero accuracy on 7-digit inputs. Whatever the channel teaches the model, it teaches it within the digit range it has seen, not as an algorithm that generalizes to longer operands. This is not specific to our mechanism; vanilla transformers fail the same test in the same way, and length generalization on arithmetic is an open problem requiring approaches beyond architectural side channels.

6 Limitations

We list the constraints on interpretation, in roughly decreasing order of severity.

Single seed everywhere. Each configuration was trained once. Within the eight-layer sweep, two independent vanilla runs (one per cell) produced 4-digit accuracy of 0.792 and 0.088 — a 70-point seed swing on a single bucket. This means specific gap magnitudes (e.g., "+87 points on 4-digit") are not stable; only the direction of large effects is. We are confident that `causal_dynamic` at $L = 8$ is meaningfully above any vanilla we ran, because both vanillas are far below it on all hard buckets, but the precise gap is unreliable. Seed reruns are the most important follow-up.

Capacity confound for the dynamic variant. The $L = 8$ `causal_dynamic` model has roughly $1.6\times$ the total parameters of vanilla, with most of the extra parameters in per-source W_q, W_k projections. We cannot separate "cross-step routing helps" from "more parameters help" from these experiments. A capacity-matched static variant is needed.

Toy task and tiny model. 1-to-6-digit addition with $D = 64$, $L \in \{4, 8\}$, 4 heads. Nothing about these results extrapolates to language modeling, longer arithmetic, or larger architectures.

Partial ablation grid. We did not run the EMA cache mode, the `pre_norm` injection point, or `separate`-norm variants in this depth sweep. We did not run a controlled comparison of static-with-extra-capacity versus dynamic. We did not run intermediate depths ($L = 6$, $L = 10$, $L = 12$) that would clarify whether the dynamic-favors-depth trend continues, saturates, or reverses.

No length generalization. The 7-digit OOD bucket is at zero for all conditions; we make no claim about generalization beyond the trained digit range.

Diagnostic interpretation. Our reading of the two-pass underperformance leans on the observation that mix parameters (specifically α) decrease during training. This is suggestive but not a controlled probe of the train/inference distribution gap hypothesis. A direct test — e.g., replacing the inference-time cached source with a vanilla-forward source and measuring accuracy — would discriminate this hypothesis from alternatives.

Author–reviewer overlap. The architectural design and ablation set were shaped through interaction between the author and the AI assistant that drafted this writeup. Bias in the framing of the results is therefore possible, even though the numbers themselves are not in dispute.

7 Conclusion

A thin cross-step side channel added to a small autoregressive transformer substantially affects multi-digit addition accuracy, in a way that depends on the joint choice of routing flavor and model depth. A simple static α-weighted channel is a useful shortcut at four layers and is absorbed by depth at eight. An expressive dynamic attention-style channel gives little at four layers and produces near-perfect in-distribution accuracy at eight, where same-depth vanilla still struggles on hard buckets. A two-pass training-time approximation of the channel underperforms vanilla in all conditions, consistent with — though not proven by — a train/inference distribution-gap reading.

The interaction between channel design and depth is, to our knowledge, the most interesting finding here. It suggests that "does adding a cross-step channel help" is the wrong question; the right question is which channel design matches which depth regime. The static-versus-dynamic divide we observe may or may not generalize to other tasks and other model scales, but in this small, controlled setting it is robust to the comparisons we have made and visible across multiple buckets.

We report this as a positive but preliminary finding, contingent on the limitations enumerated above; in particular, seed reruns and a capacity-matched static-variant control are the two most important follow-ups before stronger conclusions can be drawn.

Acknowledgments. Calculations were performed on a single consumer GPU. Logging via Weights & Biases. Drafting assistance by Claude (Anthropic). All experimental decisions, runs, and final review by the author.

References

[1] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. *ICLR*, 2019.

[2] Nayoung Lee, Kartik Sreenivasan, Jason Lee, Kangwook Lee, and Dimitris Papailiopoulos. Teaching arithmetic to small transformers. *arXiv:2307.03381*, 2023.

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

[4] Biao Zhang and Rico Sennrich. Root mean square layer normalization. *NeurIPS*, 2019.