TRANSFORMERS MISS THE MIDDLE DIGITS: REPAIRING LONG MULTIPLICATION BY RECURSION AND ROBUST RECOMBINATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Transformers trained to multiply long integers tend to get the first and last digits of the product right and the middle ones wrong. We show that this is structure, not noise. For a flat model that writes the product directly, the five central digits of 5×5 products are predicted at chance level (10–13%) while the first digit is still 74% accurate, and the failing band widens with the number of digits. The triangular profile of column interactions motivates this diagnosis. Under an explicit residual-channel model, we prove a digitwise error law and conditions for a central error peak and a widening band near chance. We quantify corrections for carry errors, dependence, and decoder mismatch. We also prove an exact local recombination rule and attention-leakage and margin conditions for its stable execution. We build the *Recursive Arithmetic Transformer* (RAT), a single Transformer that is never supervised on products beyond 2×2 digits. Exact oracles for SPLIT move its accuracy on 9- and 10-digit products by at most 4.2 points, and training COMBINE on wider inputs raises exact match from 73% to 86% on 10×10 and from 36% to 55% on 18×18 multiplication, with a gain at every length from 8 to 18 digits. Long-multiplication failures should be diagnosed, and repaired, one stage at a time.

1 INTRODUCTION

Multiplication is a stubborn test for Transformers: models that add numbers with tens or hundreds of digits (Jelassi et al., 2023; McLeish et al., 2024; Cho et al., 2024a) struggle to multiply numbers with a handful of them (Dziri et al., 2023; Lee et al., 2023; Shen et al., 2023). Much of the recent progress changes how a *single* sequence is written — reversing digits, adding scratchpads, coupling or randomizing positions (Nye et al., 2021; Lee et al., 2023; Ruoss et al., 2023; Cho et al., 2024b) — and each change buys some additional length; Table 1 shows where multiplication currently stands.

Table 1: Long multiplication with Transformers at a glance: the first row is measured in this paper (Section 2), rows two–four restate the cited abstracts, and the last row is our method, from the repair study of Section 6.

Approach	Multiplication result
Flat model, MSD-first output	exact match 27, 19, 4, 0% on 2×2, 3×3, 4×4, 5×5
Positional description (Shen et al., 2023)	up to 15 digits, essentially perfect up to 12, where usual training fails at 4
Train-set priming (Jelassi et al., 2023)	5×3 extends to 35×3 only after adding 10–50 long examples
Scratchpad and position coupling (Cho et al., 2024b)	about 2–3× longer operands than seen in training
RAT (ours), base case 2×2	90% at 8×8 and 36% at 18×18; 95% and 55% with robust COMBINE

A closer look at *where* the errors fall changes the picture. We take a flat Transformer that writes the product directly and score every output digit separately (Figure 1). The first digit of a 5×5 product is right 74% of the time. The five digits in the middle are right 10–13% of the time, which is the rate of guessing. Exact match counts these outputs as equally wrong and hides the pattern: what fails is a wide band that starts at the center of the product and grows with the number of digits.

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

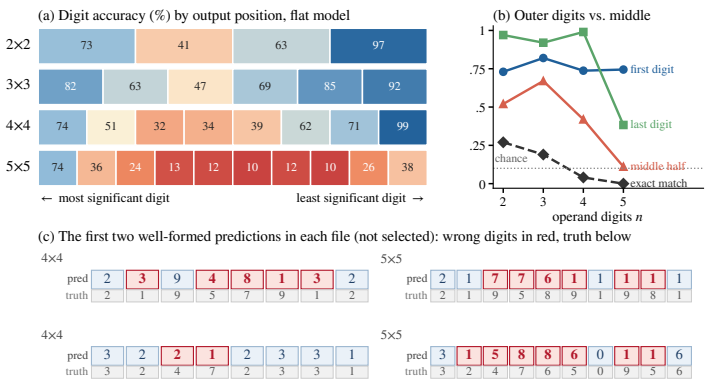

Figure 1: **The middle digits break first.** (a) Accuracy of each output digit of a flat Transformer writing the $2n$-digit product directly, most significant digit first (100 products per n; malformed outputs excluded, Section 2); uniform ten-digit guessing has accuracy 10%. (b) First/last digits degrade slowly with n; the middle half collapses to chance. (c) First well-formed predictions in each file, not selected: outer digits right, middle wrong.

The arithmetic suggests a mechanism. Among raw convolution columns $0 \le r \le 2n-2$, the number of digit products is $w_n(r) = \min(r+1, 2n-1-r)$ (Lemma 1), largest in the middle. Carries add dependencies across columns, while the most significant output column $2n-1$ is carry-only. This count describes a particular arithmetic representation; by itself it is not an error lower bound. Theorem 1 connects the count to a digitwise probability under explicit product-residual and carry assumptions: more imperfect contributions drive the reference digit toward chance. Theorem 2 quantifies when that positional ordering transfers to a greedy Transformer despite channel dependence, decoder mismatch, and earlier generation errors. We then analyze the bounded arithmetic interactions introduced by recursion and their realization in a growing prompt.

Multiplication offers exactly such a computation: the block identity $AB = P_{\ell\ell} + (P_{\ell h} + P_{h\ell})10^k + P_{hh}10^{2k}$ cuts the interaction profile of an $n \times n$ product into four profiles of half the width plus an addition (Figure 4, Appendix L), and applied recursively until every leaf is a 2×2 product it leaves only two kinds of steps: multiplying two-digit numbers, and adding four aligned numbers with carries. We call the resulting system the *Recursive Arithmetic Transformer* (RAT): one Transformer used in three roles, BASE, SPLIT, COMBINE, by a small external executor that owns the recursion stack, never supervised on a product larger than 2×2, its full recursive trace never a training example (Section 4).

The design makes the bottleneck measurable. Recombination needs only a fixed-size arithmetic state, but its prompt still grows with width. Experiments locate the responsible frame: an exact SPLIT oracle barely moves accuracy on 9- and 10-digit products, while training COMBINE at four times the baseline scale raises exact match at every length from 8 to 18 digits, supporting a stage-specific repair without identifying a unique learned mechanism.

Our contributions: (i) we document the *middle-digit bottleneck* and formulate an interaction-width hypothesis (Sections 2–3); (ii) we introduce RAT, a shared Transformer used by an external recursive executor in three frames (Section 4); (iii) we prove a conditional digitwise error law with a central peak and widening band, an exact local carry-recovery rule, and stability conditions for its neural execution, with a supporting loss-to-error proposition (Section 5, Appendix B); and (iv) three training seeds agree on the main failure lengths, oracle substitution isolates the dominant residual stage, and COMBINE adaptation gives a consistent dose–response over $n \in \{8, 10, \ldots, 18\}$: $+12.5$ points on average at $4\times$ scale and $+19$ at 18×18 (Section 6).

2 A MOTIVATING EXPERIMENT: THE MIDDLE DIGITS BREAK FIRST

We begin with the empirical phenomenon that motivates this paper. Exact-match accuracy answers whether a product is right; it does not say *where* a wrong product goes wrong. Following the digit-

level analyses of Lee et al. (2023), we align each predicted product with the target and score every output position separately.

Setup. We evaluate a flat decoder-only Transformer that reads $A * B =$ and writes the $2n$-digit product directly, most significant digit first, the output format studied by Lee et al. (2023). For each $n \in \{2, 3, 4, 5\}$ we score 100 test products, excluding predictions that are not a string of exactly $2n$ digits: none for $n \leq 3$, one for $n = 4$, six for $n = 5$ (stray tokens such as `A=` in place of digits). Every number in this section is computed from the released prediction files.

The failure is positional. Figure 1(a) shows the accuracy of each output digit. At $n = 5$ the first digit is right 74% of the time; the five central digits are right 13, 12, 10, 12, and 10%, near the 10% uniform-guessing benchmark. The last digit is also degraded (38%), but still three times more accurate than any central digit. Figure 1(c) makes this concrete: the model writes `2 1 7 7 6 1 1 1 1 1` for the target `2 1 9 5 8 9 1 9 8 1`, outer digits right, interior wrong.

The failing band grows with n. Appendix Table 8 and Figure 1(b) summarize the trend. Exact match falls from 27% at $n = 2$ to 19, 4, and 0% at $n = 5$ (none of 94 well-formed outputs). The first digit stays between 73 and 82%, while the middle half falls from 67% at $n = 3$ to 42% at $n = 4$ and 11% at $n = 5$: the band does not just deepen, it widens, from two digits below 35% at $n = 4$ to five at or near chance at $n = 5$.

Difficulty follows interaction width, with carries on top. Pooling columns across n by the number w of digit products they aggregate (Appendix Table 5), accuracy falls monotonically with w: in the lower half, $w = 1, \ldots, 5$ are right 82, 61, 47, 26, 10% of the time, and at equal w the upper half, with its longer carry chain, is markedly worse (48, 35, 24, 12% for $w = 1, \ldots, 4$). Each entry averages one to four columns, so this is illustration, not a fit, but it raises the question the rest of the paper answers: *why is the middle special, and what does that tell us about how multiplication should be computed?*

The 10% reference is uniform guessing over ten digits, not a position-specific majority baseline, and MSD-first flat outputs differ in generation order from RAT's LSD-first outputs; the comparison is descriptive.

3 From Diagnosis to Design: The Interaction-Width Hypothesis

Write two n-digit operands as $A = \sum_{i<n} a_i 10^i$ and $B = \sum_{j<n} b_j 10^j$. Column r of the product AB receives

$$u_r = \sum_{i+j=r} a_i b_j + \kappa_r, \qquad y_r = u_r \bmod 10, \qquad \kappa_{r+1} = \lfloor u_r/10 \rfloor, \tag{1}$$

where y_r is the output digit and κ_r the carry from the column below.

Lemma 1 (Interaction profile)**.** *For $0 \leq r \leq 2n-2$ the number of digit products entering column r is $w_n(r) = \min(r+1,\ 2n-1-r)$, and the carry κ_r is a function of all $\sum_{r'<r} w_n(r')$ products in the columns below.*

The proof counts pairs (i, j) (Appendix C). The least significant column has one product and no incoming carry. The central raw column has n products, and the top output column is carry-only. A carry is a function of earlier products, but this functional dependence does not imply that every such product must be retrieved directly for the next prediction. Proposition 4 makes that distinction explicit for LSD-first arithmetic. The count alone does not prove the location or probability of errors in the MSD-first flat baseline. Section 5.1 supplies a conditional probability law, and Section B.1 states the additional assumptions needed to connect it to that decoder.

Hypothesis. We hypothesize that reducing arithmetic interaction width can mitigate the positional bottleneck, provided the model also learns reliable alignment and retrieval (H1). Recombination is a plausible source of the remaining error because it aligns several numbers across growing prompts (H2). If it dominates for a checkpoint, replacing COMBINE by an oracle or training it on a more suitable distribution should gain more than repairing an already reliable frame (H3). These are conditional empirical predictions; copying and routing can also fail as inputs grow. The digitwise law makes (H1) more specific: with comparable residual-channel quality and carry reliability, more

contributions lower reference accuracy, and the ordering persists for the decoder if its quantified discrepancies are smaller than the predicted separation.

The four blocks in Figure 4 (Appendix) are products of half the width. Recursing to $M = 2$ bounds leaf multiplication width. Each internal node then adds four shifted partial products, of which at most three are active in any one column. Section 5 proves that the correct next sum digit depends on two adjacent column sums and the preceding correct output digit. It then quantifies the additional retrieval and margin conditions required by the neural decoder. Many calls remain, so local errors can still accumulate through the recursion tree.

4 RAT: Recursion That Removes Wide Interactions

The hypothesis of Section 3 asks for a computation whose every prediction has bounded interaction width. This section turns the block identity into an algorithm, explains what each of its three blocks does, identifies its arithmetic inputs, and states how it is trained.

Shared decoder and loss. Each frame is $Z = (x, \pi, m)$: tokens, decimal place IDs, and a loss mask. A standard causal decoder (Vaswani et al., 2017), shared by BASE, SPLIT, and COMBINE, uses

$$h_t^{(0)} = e(x_t) + p(\pi_t), \qquad f_\theta(\cdot \mid x_{<t}; \pi) = \mathrm{softmax}(W_o h_{t-1}^{(H)}), \tag{2}$$

and is trained by

$$\mathcal{L}(\theta) = \frac{\sum_{b,t} m_t^{(b)} [-\log f_\theta(x_t^{(b)} \mid x_{<t}^{(b)}; \pi^{(b)})]}{\sum_{b,t} m_t^{(b)}}. \tag{3}$$

Prompt tokens have mask zero.

4.1 Algorithm

RAT is a single decoder-only Transformer f_θ used in three roles, with a small external executor that owns the recursion stack. Integers are written least significant digit first, so carries travel in the direction of generation; every digit carries an explicit decimal *place ID*, digit j at local origin o receiving the learned embedding $p(o + j)$. The origin resets on every call, so place IDs never grow with recursion depth and each call's token count depends only on its own width. The vocabulary is the ten digits plus a few control tokens (`SPLIT`, `CALL`, `COMBINE`, separators).

Algorithm 1 is the whole method: a malformed generation at any call returns a failure that propagates to the root, so an exact-match error is an error somewhere in the tree. An 8×8 product's tree has 16 BASE leaves, 5 SPLIT calls, and 5 COMBINE calls; the 26-call trace never appears as a training example (Appendix K works through 1234×5678 call by call).

4.2 The three blocks

Table 9 (Appendix L) summarizes the three frames.

BASE **multiplies, and only for at most** $M = 2$ **digits.** Given operands of at most M digits, BASE writes a grade-school scratchpad, the shifted partial products $A\, b_j$ followed by their sum, then the

Algorithm 1 RAT inference. The executor calls the same network f_θ in three roles.

function MULTIPLY(A, B)
  prompt f_θ with $A * B =$ at local origin o=1 and decode greedily ▷ first token routes: `SPLIT` or BASE
  if first token is `SPLIT` **then** ▷ SPLIT: $L > M$
    parse k and the four pairs $(U_c, V_c)_{c=1}^4$ (low/high halves of A, B)
    for $c = 1, \ldots, 4$ **do** $P_c \leftarrow$ MULTIPLY(U_c, V_c)
    end for
    return f_θ-decoded COMBINE$(P_1, P_2, P_3, P_4; k)$ ▷ $P_1 + (P_2 + P_3)10^k + P_4 10^{2k}$
  else ▷ BASE: $L \le M$
    return the product parsed from the scratchpad
  end if
end function

product — the only frame that multiplies, and by construction a column of a BASE call aggregates at most M digit products. Its multiplication training support contains products up to 2×2; no larger direct multiplication target is used.

SPLIT **copies.** Given operands of $L > M$ digits, SPLIT writes `SPLIT`, the width $k = \lceil L/2 \rceil$, and the four child pairs the block identity requires. Every emitted operand digit is a copy of one prompt digit (digit j of A_{hi} is digit $k + j$ of A), with alignment represented by place IDs. Predicting the split width, routing tokens, and boundaries are additional structural decisions; the copying observation alone does not guarantee their correctness.

COMBINE **adds, with carries.** Given the four child products and k, COMBINE writes the aligned sum; the powers $10^k, 10^{2k}$ are never tokens but place-ID shifts $0, k, k, 2k$, so digits that must be added share a place ID. Column by column,

$$s_r = \sum_{c=1}^{4} d_{c,r} + \kappa_r, \qquad y_r = s_r \bmod 10, \qquad \kappa_{r+1} = \lfloor s_r/10 \rfloor, \qquad \kappa_r \in \{0, 1, 2, 3\}, \quad (4)$$

where $d_{c,r}$ is the digit of the cth aligned addend. The displayed carry range is valid for generic four-addend inputs. For legal child products, the low and high addends never overlap, so at most three addends are active and the carry is at most two. Carries are not emitted as separate tokens. Theorem 3 shows how a correct preceding output digit and the preceding column sum recover the next carry exactly. Both SPLIT and COMBINE frames grow with operand width; bounded arithmetic locality does not by itself ensure reliable retrieval from either growing prompt.

4.3 TRAINING

Training examples are generated online for the shared parameters: each batch slot draws a frame from a categorical mixture and generates a legal instance for equation 3, and the complete recursive trace never appears as a training example. In the diagnostic configuration (Appendix J) the mixture is $[0.15, 0.10, 0.75]$ for BASE, SPLIT, COMBINE.

Robust recombination. We test (H3) by continuing to train COMBINE on addends $\rho \in \{3, 4\}$ times wider than in normal training ("3×", "4×" compositional scale), with program, executor, frame formats, and evaluation unchanged. Parameters are shared, so adaptation can change other frames; the protocol is in Appendix G.

5 DIGITWISE ERROR, LOCAL ARITHMETIC, AND DECODER STABILITY

We connect column interaction width to digitwise error under an explicit residual channel (Figure 2), then identify the exact arithmetic state of recombination. Decoder-transfer bounds, stability certificates and a loss-to-error proposition are stated in Appendix B. Notation is collected in Table 3. Malformed or nonterminating executions count as failures. None of the results is an unconditional Transformer limitation, and none says that training finds the proposed mechanism.

5.1 A POSITION-SPECIFIC ERROR LAW FOR FLAT MULTIPLICATION

The number of terms in a column is not itself an error probability. We now specify a residual channel under which that count determines a digitwise law. The statement is conditional: a greedy Transformer need not implement this channel. The next subsection quantifies the discrepancy required to transfer the law to its actual predictions.

Work in radix $b \geq 2$ and index arithmetic columns from the least significant end, irrespective of generation order. At column r, list the $w_r = w_n(r)$ products as $t_{r,1}, \ldots, t_{r,w_r}$, so that $y_r = (\kappa_r + \sum_{j=1}^{w_r} t_{r,j}) \bmod b$. All residuals below take values in $\mathbb{Z}_b$. Let C_r be an incoming carry error modulo b and $E_{r,j}$ a residual in the jth product contribution. Define the reference prediction by

$$\widetilde{y}_r = \left(y_r + C_r + \sum_{j=1}^{w_r} E_{r,j} \right) \bmod b. \quad (5)$$

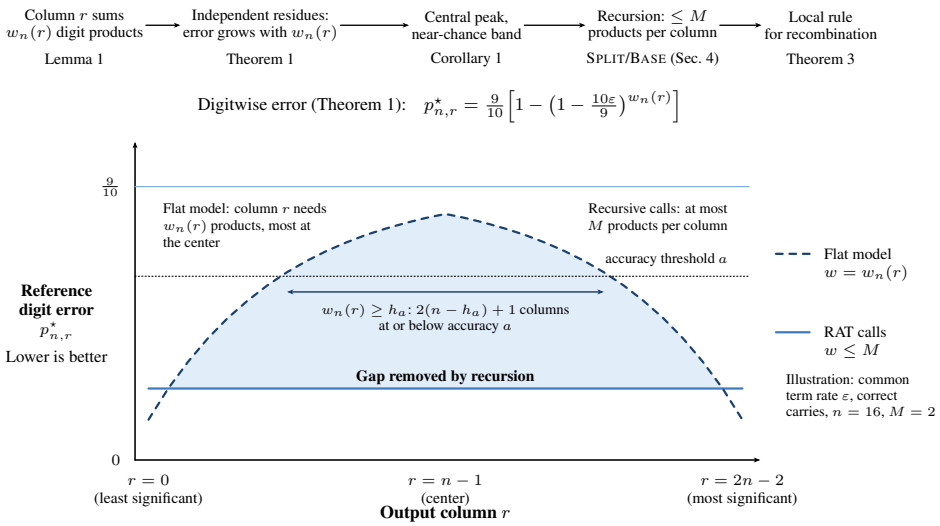

Figure 2: **From the interaction profile to a digitwise error law and to recursion.** A column of the product sums $w_n(r)$ digit products (Lemma 1); under the residual channel of Theorem 1 the reference digit error grows with $w_n(r)$, peaks at the center, and a widening band falls below any fixed accuracy (Corollary 1). Recursion keeps every call at $\leq M$ products per column. The curves are illustrative, not data: they evaluate the formula at $n = 16$, $\varepsilon = 0.12$ with correct carries, and the horizontal line applies the same channel to a call of width $M = 2$; the theorem does not assert that a trained model follows this channel.

This isolates errors at a specified arithmetic column; it does not assume that an MSD-first decoder explicitly emits a carry. Probabilities refer to a specified test-input population and any randomness in the reference channel. For the actual model below, the checkpoint and greedy tie-breaking rule are fixed; its error probability is over that input population.

For $0 \leq \varepsilon \leq 1$, define the symmetric residue law

$$\mu_\varepsilon(0) = 1 - \varepsilon, \qquad \mu_\varepsilon(e) = \frac{\varepsilon}{b-1} \quad (e \neq 0), \qquad \alpha(\varepsilon) = 1 - \frac{b\varepsilon}{b-1}. \tag{6}$$

Symmetry here concerns errors modulo b, not the distribution of true product digits. Decimal digit products themselves are not uniform residues.

Theorem 1 (Digitwise error from aggregation and carry)**.** *Suppose $E_{r,1}, \ldots, E_{r,w_r}$ are mutually independent, have laws $\mu_{\varepsilon_{r,1}}, \ldots, \mu_{\varepsilon_{r,w_r}}$, and are independent of C_r. Put $q_r = \Pr(C_r = 0)$ and $A_r = \prod_{j=1}^{w_r} \alpha(\varepsilon_{r,j})$, with empty product one. The law of the nonzero carry errors may be arbitrary. Then*

$$p^\star_r := \Pr(\widetilde{y}_r \neq y_r) = \frac{b-1}{b} - \left(q_r - \frac{1}{b}\right) A_r. \tag{7}$$

In decimal multiplication, a correct incoming carry and a common product error rate ε give

$$p^\star_{n,r} = \frac{9}{10}\left[1 - \left(1 - \frac{10\varepsilon}{9}\right)^{w_n(r)}\right], \qquad 0 \leq r \leq 2n - 2. \tag{8}$$

The formula accounts for distinct residuals that cancel modulo ten.

Why the law has this form. If a_w is the probability that w independent residuals with a common rate sum to zero, then $a_{w+1} = (1 - \varepsilon)a_w + \varepsilon(1 - a_w)/(b - 1)$ and $a_0 = 1$. Consequently $a_w = 1/b + (1 - 1/b)\alpha(\varepsilon)^w$. The same convolution calculation with unequal rates and an arbitrary independent carry gives equation 7; a full proof is in Appendix D.

Corollary 1 (Central peak and widening low-accuracy band)**.** *Let $0 < \varepsilon < (b-1)/b$ be common to all terms and columns and write $\alpha = \alpha(\varepsilon) \in (0, 1)$. If $q_r = q > 1/b$ is also common, then for*

$w_r > w_s$,

$$p_r^\star - p_s^\star = \left(q - \frac{1}{b}\right)(\alpha^{w_s} - \alpha^{w_r}) > 0. \tag{9}$$

Thus the maximum reference error is at the central convolution column $r = n - 1$ and increases to $(b-1)/b$ as $n \to \infty$ with these channel parameters fixed. For correct carries, fix an accuracy threshold $a \in (1/b, 1)$ and put

$$h_a = \left\lceil \frac{\log((a - 1/b)/(1 - 1/b))}{\log \alpha} \right\rceil. \tag{10}$$

Among the $2n-1$ convolution columns, the positions with accuracy at most a are exactly those with $w_n(r) \geq h_a$. Their number is $\max\{0,\, 2(n - h_a) + 1\}$.

A larger central band therefore falls below any fixed accuracy threshold as n grows, provided the channel description applies. The carry-only top column $r = 2n - 1$ has accuracy q_r and is excluded from the triangular count; different q_r or term rates can shift the hardest column away from the center, and when $q_r = 1/b$ the carry already makes the reference digit uniform, so the stated conditions, including generation direction, cannot be omitted when reading the empirical profile.

Transfer to the decoder. The symmetric law is a reference model, not a description of every trained Transformer. Appendix B.1 (Theorem 2) gives lower and upper bounds on each digit's actual teacher-forced and free-running error in terms of $p_r^\star$ and explicit budgets for dependence, carry errors, realization and prefix errors, and an explicit condition under which the actual middle digit is provably less reliable than an outer digit. The bounds can be vacuous when the budgets are large; we do not assume them small for the reported checkpoints. Recursion removes the factor $\alpha^{w_n(r)}$ from a multiplication step by replacing it with at most M factors; the flat-column law is not an end-to-end RAT error formula, since recursion changes the dependence structure of errors.

5.2 An exact local rule for recombination

Locality here means locality in decimal *columns*, not in the token stream: aligned digits can be far apart in the prompt. For a sum of aligned addends with at most $q \leq b$ active at any column, column sums S_r, and true carry κ_r, Theorem 3 (Appendix B.2) shows $0 \leq \kappa_r \leq q - 1$ and that the next output digit is a function of only two adjacent column sums and the preceding correct output digit, $y_r = G(S_{r-1}, y_{r-1}, S_r) = \big(S_r + \lceil (S_{r-1} - y_{r-1})/b \rceil\big) \bmod b$, independently of frame width. For legal RAT partial products (shifts $0, k, k, 2k$) at most three addends are active, so in decimal the carry lies in $\{0, 1, 2\}$ and the rule needs at most seven numerical inputs; for unrestricted four-addend inputs the bound is nine. Carries therefore need not be emitted as tokens. The theorem does not assert that the trained model learned this rule, nor does it determine end-of-sequence decisions, split widths, or which prompt token fills each arithmetic slot.

Stable execution and composition. Appendix B.3 shows that a finite-state readout with positive margin executes the local rule when attention leakage satisfies $q(b-1)\epsilon < \frac{1}{2}$ (Theorem 4), that distractor growth cannot be ignored (Proposition 1), and gives a multilayer perturbation certificate for the shared decoder (Theorem 5). Corollary 2 composes correct frames over the recursion tree, and Proposition 2 bounds the failure probability by the summed per-frame losses, $\Pr\{R_\theta \neq AB\} \leq \frac{1}{\log 2}\big[4^D \lambda_{\text{BASE}} + \frac{4^D - 1}{3}(\lambda_{\text{SPLIT}} + \lambda_{\text{COMBINE}})\big]$ for width at most $M2^D$, a quadratic worst-case certificate in n; Corollary 3 shows how wider COMBINE training shrinks its excess term. These are sufficient conditions and premises; they are not verified on our checkpoints.

6 Experiments

We ask where RAT fails (Section 6.1), which frame is responsible (Section 6.2), and whether repairing it repairs the product (Section 6.3). The metric is exact integer match under greedy decoding; a malformed generation counts as wrong. The two studies use separate checkpoints and protocols and are not comparable: the *diagnostic* study (Appendix J) uses three seeds and 1000 products per length, and the *repair* study compares a normally trained system with two COMBINE-adapted variants at 100 products per length, $n \in \{8, 10, \ldots, 18\}$. No product beyond 2×2 is a training example.

6.1 DIAGNOSIS: RAT FAILS AT PARTICULAR LENGTHS, AND SEEDS AGREE ON WHERE

Table 11 (Appendix) reports exact match for n-digit operands. RAT is accurate for most lengths, at least 93.7% for $n \in \{3, 4, 6, 7, 8\}$ in every seed, dips at $n = 5$ (82.8–84.3%), and collapses at $n = 9, 10$ (48.6–54.3%); the three seeds agree within 4 points at every length (Appendix Table 6), so the pattern belongs to the recipe, not a single run. The dips are not where a length-only account would put them: $n = 6, 7, 8$ are fine while $n = 5$ is not, and the collapse begins exactly where the recursion depth grows from 2 to 3 and the root SPLIT leaves the densely sampled range up to 8 digits. A documented long-copy branch extends nominal SPLIT support to 32, so this is not a clean out-of-support transition. The next experiment measures how much each frame contributes.

6.2 ORACLE SUBSTITUTION: THE RESPONSIBLE FRAME

To find the responsible frame we run the recursion with one frame replaced by its exact symbolic oracle and everything else neural, at $n = 9, 10$ (Table 2). An oracle SPLIT at the root, making the root decomposition exact, leaves accuracy unchanged ($+0.3, +0.1$ points); oracles at every node move it by at most $+4.2$. Replacing the third frame by its oracle lifts accuracy to 92.4% and 96.4%, the pattern (H3) predicted: exact solutions of the other steps change little, and the residual error lives in recombination. The recursion still has depth 3 under the third oracle, so depth alone is not the cause either.

Table 2: Oracle substitution at $n = 9$ and $n = 10$ (exact match %, 1000 products, one training seed). Each row replaces one frame by an exact symbolic oracle and leaves the rest neural.

Substitution	$n = 9$	$n = 10$
None (all neural)	54.3	50.6
Oracle SPLIT, root only	54.6	50.7
Oracle SPLIT, every node	58.5	51.6
Oracle COMBINE, every node	92.4	96.4

6.3 REPAIR: ROBUST RECOMBINATION

The diagnosis says to train COMBINE. We compare a normally trained system with COMBINE-adapted variants at $3\times$ and $4\times$ compositional scale (Section 4), with the recursive program, executor, and evaluation held fixed. Appendix Table 10 lists exact match for every length in the sweep, and Figure 3 plots it with finite-sample uncertainty.

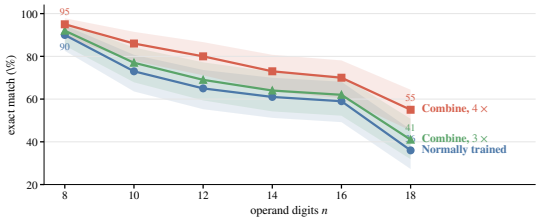

Figure 3: **Targeting recombination improves the full product.** Exact match under normal training and $3\times/4\times$ COMBINE adaptation (100 products per point; bands: 95% Wilson intervals). Both improve every length; $4\times$ is uniformly best.

The effect is uniform and grows with scale. Both adapted variants beat normal training at all six lengths, so the result does not rest on one favorable length or one recursion depth, and the ordering normal $< 3\times < 4\times$ holds at every length as well: averaged over the sweep, $3\times$ adds 3.5 points and $4\times$ adds 12.5. The gain is also largest where the task is hardest. At $n = 8$ normal training is already at 90% and $4\times$ reaches 95%; at $n = 18$ normal training drops to 36% and $4\times$ reaches 55%, a gain of 19 points, and on 10×10 it recovers 13 points, from 73% to 86%. This is consistent with repairing a dominant recombination bottleneck, but the runs do not measure the retrieval leakage or margins of Theorems 4 and 5, so they do not establish that adaptation satisfies those certificates.

What the evidence supports, and does not. With 100 products per cell the 95% Wilson half-width is 6–10 points, so the evidence is the same ordering repeating at six lengths, not any single cell. Adaptation does not remove the decay with n: at 18 digits the best system is right 55% of the time, consistent with error accumulating over up to 286 calls (Appendix F). More seeds and wider adaptation ($\rho > 4$) are the next controls.

7 Related Work

Arithmetic and length generalization. Reversed outputs, scratchpads, and explicit positions improve arithmetic reach (Nye et al., 2021; Lee et al., 2023; Shen et al., 2023); relative, randomized, and digit-aligned positions extend addition to hundreds of digits (Jelassi et al., 2023; Ruoss et al., 2023; McLeish et al., 2024; Cho et al., 2024a); a coupled scratchpad length-generalizes multiplication (Cho et al., 2024b). These alter one sequence; RAT changes the computation, bounding leaf multiplication width and the arithmetic stencil of recombination.

Where multiplication fails. Dziri et al. (2023) show multiplication failing as compositional depth grows; Lee et al. (2023) analyze arithmetic by output position; Wei (2026) argues the long-range dependency is an artifact of representation and removes it with a two-dimensional local encoding. We agree the interaction width belongs to the flat serialization; our digitwise analysis separates carry reliability from local aggregation, and its relevance to a learned checkpoint requires validation.

Formal perspectives. RASP and RASP-L characterize length-uniform algorithms (Weiss et al., 2021; Zhou et al., 2023); Huang et al. (2024) show Limit Transformers cannot express copying or n-digit addition. Cho et al. (2024b) and Hou et al. (2024) construct arithmetic and general algorithms; our contribution is the specialization to RAT's sum-only recombination interface and its stability under imperfect retrieval, not a general expressivity theory.

Recursion and decomposition. Neural Programmer-Interpreters and Divide-and-Conquer Networks learn split/merge programs from traces (Reed & de Freitas, 2015; Cai et al., 2017; Nowak et al., 2016); Recursion of Thought, least-to-most, and decomposed prompting apply the same idea to language models (Lee & Kim, 2023; Zhou et al., 2022; Khot et al., 2022). Named interfaces let us blame one stage and repair it alone; we use the four-product identity rather than Karatsuba's three-product trick (Karatsuba & Ofman, 1963), keeping COMBINE an aligned addition.

8 Conclusion and Future Work

Flat models show a pronounced middle-digit bottleneck, which the triangular arithmetic profile motivates but does not alone explain. Under explicit residual-channel assumptions our digitwise law predicts a central error peak and a widening band near chance. RAT bounds multiplication at its leaves and gives recombination a fixed-size arithmetic state with an exact local carry rule. An exact COMBINE oracle gains far more than an exact SPLIT oracle, and wider COMBINE training improves all six lengths from 8 to 18 digits, by 12.5 points on average and 19 at 18×18.

Limitations. The repair uses 100 products per length and one run per variant, on a different checkpoint family from the three-seed diagnosis; the flat baseline stops at $n \leq 5$. The attention certificates are unverified sufficient conditions, and the digitwise channel has not been measured. No theorem shows that SGD learns the proposed mechanism, that every Transformer has a middle-digit bottleneck, or that RAT is correct at arbitrary lengths.

AI use statement

AI tools assisted with manuscript drafting, language editing. The authors remain responsible for the mathematical claims and reported experimental results.

References

Jon Z. Cai, Richard Shin, and Dawn Xiaodong Song. Making neural programming architectures generalize via recursion. *ArXiv*, abs/1704.06611, 2017. URL https://api.semanticscholar.org/CorpusID:1844940.

Hanseul Cho, Jaeyoung Cha, Pranjal Awasthi, Srinadh Bhojanapalli, Anupam Gupta, and Chulhee Yun. Position coupling: Improving length generalization of arithmetic transformers using task structure. *Advances in Neural Information Processing Systems 37*, 2024a. URL https://api.semanticscholar.org/CorpusID:273695226.

Hanseul Cho, Jaeyoung Cha, Srinadh Bhojanapalli, and Chulhee Yun. Arithmetic transformers can length-generalize in both operand length and count. *ArXiv*, abs/2410.15787, 2024b. URL https://api.semanticscholar.org/CorpusID:273502614.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Li-Wei Jiang, Bill Yuchen Lin, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Sean Welleck, Xiang Ren, Allyson Ettinger, Zaïd Harchaoui, and Yejin Choi. Faith and fate: Limits of transformers on compositionality. *ArXiv*, abs/2305.18654, 2023. URL https://api.semanticscholar.org/CorpusID:258967391.

Kaiying Hou, David Brandfonbrener, Sham M. Kakade, Samy Jelassi, and Eran Malach. Universal length generalization with turing programs. *ArXiv*, abs/2407.03310, 2024. URL https://api.semanticscholar.org/CorpusID:270924396.

Xin-Ting Huang, Andy Yang, S. Bhattamishra, Yash Sarrof, Andreas Krebs, Hattie Zhou, Preetum Nakkiran, and Michael Hahn. A formal framework for understanding length generalization in transformers. *ArXiv*, abs/2410.02140, 2024. URL https://api.semanticscholar.org/CorpusID:273098125.

Samy Jelassi, Stéphane d'Ascoli, Carles Domingo-Enrich, Yu-Huai Wu, Yuan-Fang Li, and François Charton. Length generalization in arithmetic transformers. *ArXiv*, abs/2306.15400, 2023. URL https://api.semanticscholar.org/CorpusID:259262196.

Anatolij A. Karatsuba and Yu. Ofman. Multiplication of multidigit numbers on automata. *Soviet physics. Doklady*, 7:595–596, 1963. URL https://api.semanticscholar.org/CorpusID:117858583.

Tushar Khot, H. Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. Decomposed prompting: A modular approach for solving complex tasks. *ArXiv*, abs/2210.02406, 2022. URL https://api.semanticscholar.org/CorpusID:252715485.

Nayoung Lee, Kartik K. Sreenivasan, Jason D. Lee, Kangwook Lee, and Dimitris Papailiopoulos. Teaching arithmetic to small transformers. *ArXiv*, abs/2307.03381, 2023. URL https://api.semanticscholar.org/CorpusID:259375870.

Soochan Lee and Gunhee Kim. Recursion of thought: A divide-and-conquer approach to multi-context reasoning with language models. *ArXiv*, abs/2306.06891, 2023. URL https://api.semanticscholar.org/CorpusID:259137462.

Sean McLeish, Arpit Bansal, Alex Stein, Neel Jain, John Kirchenbauer, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, Jonas Geiping, Avi Schwarzschild, and Tom Goldstein. Transformers can do arithmetic with the right embeddings. *ArXiv*, abs/2405.17399, 2024. URL https://api.semanticscholar.org/CorpusID:270062339.

Alex W. Nowak, David Folqué, and Joan Bruna. Divide and conquer networks. *arXiv: Learning*, 2016. URL https://api.semanticscholar.org/CorpusID:21473141.

Maxwell Nye, Anders Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show your work: Scratchpads for intermediate computation with language models. *ArXiv*, abs/2112.00114, 2021. URL https://api.semanticscholar.org/CorpusID:244773644.

Scott E. Reed and Nando de Freitas. Neural programmer-interpreters. *CoRR*, abs/1511.06279, 2015. URL https://api.semanticscholar.org/CorpusID:7034786.

Anian Ruoss, Gr'egoire Del'etang, Tim Genewein, Jordi Grau-Moya, R. Csordás, Mehdi Abbana Bennani, Shane Legg, and Joel Veness. Randomized positional encodings boost length generalization of transformers. *ArXiv*, abs/2305.16843, 2023. URL https://api.semanticscholar.org/CorpusID:258947457.

Ruoqi Shen, Sébastien Bubeck, Ronen Eldan, Yin Tat Lee, Yuanzhi Li, and Yi Zhang. Positional description matters for transformers arithmetic. *ArXiv*, abs/2311.14737, 2023. URL https://api.semanticscholar.org/CorpusID:265456551.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Neural Information Processing Systems*, 2017. URL https://api.semanticscholar.org/CorpusID:13756489.

Zichao Wei. On the mirage of long-range dependency, with an application to integer multiplication. *ArXiv*, abs/2603.29069, 2026. URL https://api.semanticscholar.org/CorpusID:286977326.

Gail Weiss, Yoav Goldberg, and Eran Yahav. Thinking like transformers. *ArXiv*, abs/2106.06981, 2021. URL https://api.semanticscholar.org/CorpusID:235421630.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed H. Chi. Least-to-most prompting enables complex reasoning in large language models. *ArXiv*, abs/2205.10625, 2022. URL https://api.semanticscholar.org/CorpusID:248986239.

Hattie Zhou, Arwen Bradley, Etai Littwin, Noam Razin, Omid Saremi, Josh M. Susskind, Samy Bengio, and Preetum Nakkiran. What algorithms can transformers learn? a study in length generalization. *ArXiv*, abs/2310.16028, 2023. URL https://api.semanticscholar.org/CorpusID:264439160.

Ethics statement

This work studies synthetic integer-arithmetic tasks. It involves no human subjects or personal data and no deployment in a safety-critical setting. We distinguish carefully between demonstrated generalization and claims of arbitrary-length correctness, which we do not make.

Reproducibility statement

Section 4 and Appendix I specify the recursive computation, frame encoders, and training objective. Proofs are in Appendices C, D, and E. Appendix J lists the training configuration, and the figures and tables of Section 2 are generated from the released prediction files.

A NOTATION

Table 3: Main symbols. Several base letters are reused in different sections; the table gives each meaning, and the section where it applies.

Symbol	Meaning
n, M, D	operand width in digits; base cutoff ($M = 2$); recursion depth, $n = M2^D$
a_i, b_j, A, B	digits and operands (Sections 3–4); b without subscript is the radix (Section 5, $b = 10$)
$w_n(r)$	number of digit products in column r, $\min(r + 1, 2n - 1 - r)$
u_r, y_r, κ_r	column total including carry, output digit, incoming carry
C_r, $E_{r,j}$	carry error and residual of the jth product in column r (mod b)
ε, $\alpha(\varepsilon)$	term error rate; $1 - b\varepsilon/(b - 1)$
q_r, A_r	$\Pr(C_r = 0)$; $\prod_j \alpha(\varepsilon_{r,j})$ (not the operand A)
$p^\star_r$, h_a	reference digit error of Theorem 1; smallest width with accuracy at most a
$\eta_r, \delta_{r,j}, \beta_r$	realization, conditional-TV, and prefix discrepancies of Theorem 2; B_r, D_r are the resulting budgets (not the operand B or depth D)
S_r, q, G	aligned column sum; maximum number of active addends; local recovery map of Theorem 3
ϵ, Δ, m	attention leakage; score gap; number of outside keys (also the loss mask m_t in equation 3 and a margin m_r in Appendix B.1)
k, $P_{\ell\ell}, P_{\ell h}, P_{h\ell}, P_{hh}$	split width; the four half-products of the block identity
$R_\theta(A, B)$	RAT's output for (A, B)
Λ_Q, λ_Q	per-frame summed loss and its expected bound, $Q \in \{\text{BASE}, \text{SPLIT}, \text{COMBINE}\}$
ρ, K	COMBINE adaptation scale ($3\times, 4\times$); width range with low COMBINE loss

B ADDITIONAL THEORY STATEMENTS

This appendix collects the statements moved out of the main text; proofs are in Appendices C, D and E.

B.1 TRANSFERRING THE LAW TO AN AUTOREGRESSIVE DECODER

The exact symmetric law is a transparent reference model, not an assumed description of every trained Transformer. We allow biased, dependent residuals and quantify agreement with the decoder. For probability measures on a finite set, write $\mathrm{TV}(P, Q) = \frac{1}{2}\sum_x |P(x) - Q(x)|$. Let $\widehat{y}^{\mathrm{TF}}_r$ be the actual greedy next token given the correct prompt and correct preceding output tokens, at the token position corresponding to arithmetic column r. A nondigit token is incorrect. Take proposed residuals $C_r, E_{r,1}, \ldots, E_{r,w_r}$ on the same input probability space, and define $\widetilde{y}_r$ by equation 5. These residuals need not be independent. For every history of positive probability, assume

$$\mathrm{TV}\big(\mathcal{L}(E_{r,j} \mid C_r, E_{r,1}, \ldots, E_{r,j-1}), \mu_{\varepsilon_{r,j}}\big) \le \delta_{r,j}. \tag{11}$$

This is stronger than matching marginal error rates and explicitly controls dependence on the incoming carry. Suppose also that $\Pr(\widehat{y}^{\mathrm{TF}}_r \neq \widetilde{y}_r) \le \eta_r$. The latter condition measures realization of the proposed arithmetic computation, including routing, formatting, and readout errors at this token.

Theorem 2 (Digitwise decoder bounds with misspecification)**.** *Under the preceding conditions, compute $p^\star_r$ from equation 7 using $q_r = \Pr(C_r = 0)$ and the reference rates $\varepsilon_{r,j}$, and put*

$$B_r = \min\left\{1, \eta_r + \sum_{j=1}^{w_r} \delta_{r,j}\right\}. \tag{12}$$

Then the actual teacher-forced error probability satisfies

$$\max\{0, p^\star_r - B_r\} \le p^{\mathrm{TF}}_{\theta,r} \le \min\{1, p^\star_r + B_r\}. \tag{13}$$

Let $\widehat{y}^{\mathrm{FR}}_r$ be the corresponding free-running greedy token, counting early termination or missing tokens as incorrect, and let β_r be the probability that its generated prefix before this position differs from the correct prefix. Set $D_r = \min\{1, B_r + \beta_r\}$. Then

$$\max\{0, p^\star_r - D_r\} \le p^{\mathrm{FR}}_{\theta,r} \le \min\{1, p^\star_r + D_r\}. \tag{14}$$

No independence between prefix errors and local errors is required. In the common-rate, common-carry setting of Corollary 1, the actual positional comparison is

$$p^{\mathrm{FR}}_{\theta,r} - p^{\mathrm{FR}}_{\theta,s} \geq \left(q - \frac{1}{b}\right)(\alpha^{w_s} - \alpha^{w_r}) - D_r - D_s. \tag{15}$$

Consequently, the middle digit is provably less reliable than an outer digit whenever the reference separation exceeds the two discrepancy budgets. The teacher-forced comparison uses $B_r + B_s$ instead.

For unequal carry reliabilities, the reference separation in equation 15 becomes $(q_s - 1/b)A_s - (q_r - 1/b)A_r$. This makes a distinction that an increasing upper bound cannot make: we obtain lower and upper bounds for each digit, and an explicit sufficient condition for ordering the *actual* error probabilities. The bounds may be uninformative when the discrepancy budgets are large; we do not assume them small for the reported checkpoints. In MSD-first output, β_r concerns more significant output tokens, whereas C_r concerns the arithmetic carry from less significant columns. They are different mechanisms.

A connection to the decoder's logits. The multilayer comparison in Theorem 5 provides one way to bound η_r. Choose a reference attention computation with the same parameters, and let $z_r, \bar{z}_r$ be actual and reference logits on the correct prefix. Suppose it realizes the proposed token $\widetilde{y}_r$ outside an event J_r and that $\|z_r - \bar{z}_r\|_\infty \leq u_r$ outside an event U_r. On J_r^c, let $m_r = \bar{z}_{r,\widetilde{y}_r} - \max_{a \neq \widetilde{y}_r} \bar{z}_{r,a}$, with the maximum over the entire vocabulary. One admissible choice is

$$\eta_r := \Pr(J_r \cup U_r \cup \{m_r \leq 2u_r\}), \tag{16}$$

where m_r, u_r may be assigned arbitrarily on the excluded events. Theorem 5 supplies $u_r = K_{\mathrm{out},r} E_{L,r}$ on a certified domain. Here the margin is for the reference's proposed token, which may be wrong. The argument therefore transfers both correct and incorrect reference predictions. An arbitrary masked decoder is not automatically a realization of equation 5; that identification and the residual-channel bounds remain explicit premises.

How recursion addresses the mechanism. Under the same channel quality and carry reliability, replacing a flat column with many product contributions by one with at most M such contributions removes the factor $\alpha^{w_n(r)}$ from that local multiplication step and replaces it by a product of at most M factors. RAT does this at its leaves. Its recombination adds shifted partial products, and its additional retrieval, carry, structural, and repeated-call errors require the separate analysis below. The flat-column law is not an end-to-end RAT error formula: recursion changes both the computation and the dependence structure of its errors.

B.2 An exact local rule for recombination

Locality here means locality in *decimal columns*, not proximity in the serialized token stream. Aligned digits can be far apart in the prompt. The theorem specifies the information required for arithmetic; the retrieval results specify when attention supplies that information reliably.

Work first in radix $b \geq 2$. Let $d_{c,r} \in \{0, \ldots, b-1\}$ be aligned input digits, taking missing digits to be zero, and let $S_r = \sum_c d_{c,r}$. Assume at most q addends can be active at any column, with $1 \leq q \leq b$. The exact addition recurrence is

$$y_r = (S_r + \kappa_r) \bmod b, \qquad \kappa_{r+1} = \left\lfloor \frac{S_r + \kappa_r}{b} \right\rfloor, \qquad \kappa_0 = 0. \tag{17}$$

Theorem 3 (Arithmetic locality)**.** *Under equation 17, $0 \leq \kappa_r \leq q - 1$ for every r. For $r \geq 1$,*

$$\kappa_r = \left\lceil \frac{S_{r-1} - y_{r-1}}{b} \right\rceil, \qquad y_r = G(S_{r-1}, y_{r-1}, S_r), \tag{18}$$

where

$$G(u, v, w) = \left(w + \left\lceil \frac{u - v}{b} \right\rceil\right) \bmod b. \tag{19}$$

The formula also applies to $r = 0$ with $S_{-1} = y_{-1} = 0$. It uses at most $2q$ input digits and one preceding output digit, independently of frame width. Its application to autoregressive decoding assumes that the preceding emitted digit is correct.

For legal RAT partial products with shifts $0, k, k, 2k$, at most three addends are active at any column. In decimal, the carry lies in $\{0, 1, 2\}$ and the rule needs at most seven numerical inputs. For unrestricted four-addend decimal inputs, the safe bound is nine.

The specialization uses $P_{\ell\ell} < b^{2k}$: the unshifted low product ends before the high product begins at column $2k$. The two middle products may overlap either one. This refinement holds for correct arithmetic children; arbitrary erroneous child integers need not satisfy the same support bounds.

Theorem 3 supplies a local rule for our existing sum-only continuation. Carries need not be additional output tokens. It does not assert that the trained model has learned this rule. It also does not determine the end-of-sequence decision, the split width, or which prompt token occupies each arithmetic slot.

B.3 Stable arithmetic from approximate retrieval

The ceiling in equation 19 should not be applied directly to noisy real-valued sums: a valid state may lie exactly at one of its discontinuities. A finite-state readout avoids that problem. Define the admissible aggregate states

$$\mathcal{S}_q = \left\{ (u, v, w) : \begin{array}{l} 0 \le u, w \le q(b-1), \quad u, v, w \text{ integers}, \\ v = (u + c) \bmod b \text{ for some } c \in \{0, \dots, q-1\} \end{array} \right\}. \tag{20}$$

The initial state is included. For digit a, set

$$g_a(x) = - \min_{s \in \mathcal{S}_q : G(s) = a} \|x - s\|_\infty . \tag{21}$$

Each class is nonempty. This explicit readout is an existence construction, not an identification of our learned feed-forward weights.

Lemma 2 (A positive-margin local readout)**.** *At every $s \in \mathcal{S}_q$, the correct logit in equation 21 is zero and all incorrect logits are at most -1. Every g_a is 1-Lipschitz in the maximum norm. Consequently, if $\|\widehat{s} - s\|_\infty < 1/2$, greedy decoding of $g(\widehat{s})$ returns $G(s)$. The functions g_a have a finite ReLU realization whose size depends on b, q, but not on operand length.*

Consider a scalar retrieval head for one required input digit. Its allowed set is nonempty and all allowed values equal that digit; all other scalar values lie in $[0, b-1]$. Missing slots must have a verified zero default. Let ϵ be the total attention weight outside the allowed set. Then the scalar retrieval error is at most $(b-1)\epsilon$.

Theorem 4 (Stable local recombination)**.** *Suppose the correct previous output digit is available exactly. At each of the two columns needed by Theorem 3, retrieve the at most q active input digits with heads of the preceding form, each with leakage at most ϵ. Sum the retrieved digits and use the readout equation 21. The next digit is correct whenever*

$$q(b-1)\epsilon < \frac{1}{2}. \tag{22}$$

If the preceding output digit is also retrieved approximately with error η, replace the left side by $\max\{q(b-1)\epsilon, \eta\}$. If the bound holds at every digit and structural and termination decisions are correct, the entire recombination is correct by induction.

For one such head, let m be the number of eligible causal keys outside its allowed set and let Δ be the largest allowed score minus the largest outside score. When $m > 0$,

$$\epsilon \le \frac{m e^{-\Delta}}{1 + m e^{-\Delta}}. \tag{23}$$

Thus a sufficient strict score-gap condition, when the previous digit is exact, is

$$\Delta > \log\big(m[2q(b-1) - 1]\big). \tag{24}$$

For $m = 0$ the leakage is zero and no gap condition is needed. For decimal inputs, equation 22 becomes $\epsilon < 1/72$ for generic four-addend inputs and $\epsilon < 1/54$ for legal RAT inputs with verified inactive slots.

The theorem is a constructive sufficient condition. It does not assume that the checkpoint has scalar digit-valued heads or the readout equation 21; these are explicit premises for this specialization. The next result addresses the shared multilayer decoder without those architectural identifications.

Proposition 1 (Distractor growth cannot be ignored)**.** *There is a one-head binary copying task with a fixed relevant set of size one and fixed score gap Δ for which greedy copying is correct when $m < e^{\Delta}$ and incorrect when $m > e^{\Delta}$. Thus constant arithmetic arity and a constant positive score gap alone do not guarantee correctness for unbounded frame lengths under unrestricted softmax attention.*

This proposition concerns a specified readout and distractor family, not all Transformers. It identifies a reason for the logarithmic distractor term in equation 24. Masking can keep m bounded; learned score separation can increase with length. The existing dense-attention implementation requires evidence for its own behavior.

B.4 A CERTIFICATE FOR THE ACTUAL MULTILAYER DECODER

Fix a correct prompt and a correct output prefix. All quantities in this section refer to this ideal next-token input. Write an inference-time decoder block as

$$H^{\ell} = F_{\ell}\big(H^{\ell-1} + A_{\ell}(H^{\ell-1})\big), \qquad \ell = 1, \dots, L. \tag{25}$$

Here A_{ℓ} is the multihead attention contribution; pre-normalization, when present, is included in its query, key, and value maps. The positionwise map F_{ℓ} includes the remaining normalization and feed-forward residual operations. Inference dropout is disabled. This covers ordinary pre- and post-normalized decoder blocks with suitable definitions of the maps.

Choose a nonempty subset $J_{\ell ht}$ of the causal keys for each head h and query position t. A reference decoder uses the same parameters and input embeddings, but restricts each attention softmax to these keys. Write $\bar{A}_{\ell}$ for this restricted attention map, $\bar{H}^{\ell}$ for its hidden states, and $\bar{z}$ for its output logits. These sets are fixed from the ideal prompt and its arithmetic roles. The reference is a mathematical intervention; it is not an extra inference-time oracle in the deployed method.

Use $\|H\|_{\max,2} = \max_t \|H_t\|_2$. On a domain containing the relevant trajectories and intermediate arguments, suppose F_{ℓ} is M_{ℓ}-Lipschitz and $\bar{A}_{\ell}$ is C_{ℓ}-Lipschitz in this norm. Let $O_{\ell h}$ be the output projection of head h, and bound each actual value vector by $V_{\ell h}$. Define the leakage from the *actual* attention pass by

$$\epsilon_{\ell h} = \max_t \sum_{j \notin J_{\ell ht}} a_{\ell h, tj}(H^{\ell-1}), \qquad \delta_{\ell} = 2 \sum_h \|O_{\ell h}\|_{2\to 2}\, V_{\ell h} \epsilon_{\ell h}. \tag{26}$$

The maximum may be restricted to a backward-closed dependency set if every required residual, query, key, and value predecessor is included at the preceding layer. Final-layer attention alone is insufficient.

Theorem 5 (Multilayer locality and stability certificate)**.** *Under the preceding conditions, put $E_0 = 0$ and*

$$E_{\ell} = M_{\ell}\big[(1 + C_{\ell})E_{\ell-1} + \delta_{\ell}\big]. \tag{27}$$

Then $\big\|H^{\ell} - \bar{H}^{\ell}\big\|_{\max,2} \le E_{\ell}$, and

$$E_L = \sum_{\ell=1}^{L} M_{\ell}\delta_{\ell} \prod_{j=\ell+1}^{L} M_j(1 + C_j). \tag{28}$$

If the final logit map is K_{out}-Lipschitz from the last hidden state in Euclidean norm to logits in maximum norm, then

$$\|z - \bar{z}\|_{\infty} \le K_{\mathrm{out}} E_L. \tag{29}$$

For the correct next token y, define its reference margin against the entire *vocabulary by*

$$\gamma = \bar{z}_y - \max_{a \ne y} \bar{z}_a. \tag{30}$$

The actual decoder emits y whenever

$$\boxed{\gamma > 2K_{\mathrm{out}} E_L.} \tag{31}$$

Each $\epsilon_{\ell h}$ can be bounded by equation 23, using a uniform lower bound on that head's relevant score gaps and an upper bound on its distractor counts at the actual hidden states.

For our linear output map, $K_{\text{out}} = \|W_o\|_{2\to\infty}$; any additional final normalization must be included. Appendix E gives explicit sufficient constants C_ℓ for scaled dot-product attention. Empirical sensitivity estimates are diagnostic estimates, not certified global Lipschitz upper bounds.

To turn the reference into a truly local arithmetic decoder, its backward dependency graph must close on the numerical stencil of Theorem 3 and verified structural information. Its readout must realize the local rule with a positive margin. A sparse final attention map does not establish either condition: earlier hidden states may already contain nonlocal information, and learned positional embeddings may introduce position-dependent behavior. Formula equation 31 remains a valid promptwise perturbation certificate even when a length-uniform local realization has not been established.

Corollary 2 (Composition over the ideal RAT tree)**.** *For an operand pair, let $\mathcal{I}$ be its exact internal nodes and $\mathcal{D}_v$ the true sum-digit positions of ideal* COMBINE *frame v. Let F_{other} be the event that some ideal* BASE *or* SPLIT *frame is not reproduced correctly, or an ideal* COMBINE *frame has an incorrect end marker or other nondigit structural decision under its correct prefix. Define*

$$B_{vr} = \{\gamma_{vr} \le 2K_{\text{out},vr}E_{L,vr}\}. \tag{32}$$

Treat any position for which the certificate's required domain bounds are unavailable as bad. Then

$$\mathbf{1}\{R_\theta(A,B) \neq AB\} \le \mathbf{1}\{F_{\text{other}}\} + \sum_{v\in\mathcal{I}}\sum_{r\in\mathcal{D}_v}\mathbf{1}\{B_{vr}\}. \tag{33}$$

Taking expectations yields a failure-probability bound without independence assumptions. If all certificates and structural conditions hold uniformly on a supported operand range, RAT is exact on that range.

The corollary uses correct-prefix inputs only. At a first actual error, preceding decisions are correct; if no such error is possible in an ideal frame, bottom-up induction supplies correct child products and the correct root result. This does not interpret a teacher-forced loss or margin at a later wrong token as a guarantee under an already corrupted generated prefix.

B.5 A SUPPORTING LOSS-TO-ERROR CERTIFICATE

Per-frame loss. For an ideal frame v of type $Q \in \{\text{BASE}, \text{SPLIT}, \text{COMBINE}\}$, let p_v and z_v be its correct prompt and continuation, and define

$$\Lambda_Q(v;\theta) = \sum_t m_{v,t}\big[-\log f_\theta(z_{v,t} \mid p_v, z_{v,<t}; \pi_v)\big]. \tag{34}$$

Thus Λ_Q is exactly that frame's contribution to the numerator of the shared masked loss in equation 3. For COMBINE, the ideal prompt contains the four correct child products. Let $\mathcal{L}(A,B)$ and $\mathcal{I}(A,B)$ be the leaves and internal nodes of the exact recursion tree.

Proposition 2 (Loss-to-error bound for RAT)**.** *For every operand pair (A,B) under greedy decoding,*

$$\begin{aligned}\mathbf{1}\{R_\theta(A,B) \neq AB\} \le \frac{1}{\log 2}\Bigg[&\sum_{v\in\mathcal{L}(A,B)}\Lambda_{\text{BASE}}(v;\theta) \\ &+ \sum_{v\in\mathcal{I}(A,B)}\big(\Lambda_{\text{SPLIT}}(v;\theta) + \Lambda_{\text{COMBINE}}(v;\theta)\big)\Bigg].\end{aligned} \tag{35}$$

Hence the failure probability is bounded by the expectation of the right-hand side, without any independence assumption. If inputs have width at most $n = M2^D$ and every ideal frame has expected summed loss at most λ_{BASE}, λ_{SPLIT}, or λ_{COMBINE} according to its type, then

$$\Pr\{R_\theta(A,B) \neq AB\} \le \min\left\{1, \frac{4^D\lambda_{\text{BASE}} + \frac{4^D-1}{3}(\lambda_{\text{SPLIT}} + \lambda_{\text{COMBINE}})}{\log 2}\right\}. \tag{36}$$

Since $4^D = (n/M)^2$, fixed per-frame loss yields a quadratic worst-case certificate in operand width.

Proof idea. At the first wrong greedy token of a frame, the correct prefix is still present and the correct token has probability at most $1/2$; it therefore contributes at least $\log 2$ to equation 34. If every ideal BASE, SPLIT, and COMBINE frame is correct, induction over Algorithm 1 returns AB. Summing the first-error inequalities and using the tree counts proves the result. Appendix C gives the full proof.

Corollary 3 (Effect of wider COMBINE data)**.** *In the balanced tree, level d contains 4^d* COMBINE *calls of width $k_d = n/2^{d+1}$. Suppose their expected summed loss is at most λ_0 for $k \le K$ and $\lambda_1 \ge \lambda_0$ for $k > K$. Put*

$$j_K = \min\Big\{D, \max\Big\{0, \Big\lceil \log_2 \frac{n}{2K} \Big\rceil\Big\}\Big\}, \qquad N_{\mathrm{out}}(K) = \frac{4^{j_K} - 1}{3}.$$

Then

$$\Pr\{R_\theta(A,B) \neq AB\} \le \min\left\{1, \frac{4^D \lambda_{\text{BASE}} + \frac{4^D - 1}{3}(\lambda_{\text{SPLIT}} + \lambda_0) + N_{\mathrm{out}}(K)(\lambda_1 - \lambda_0)}{\log 2}\right\}. \quad (37)$$

If wider training extends the low-loss range from K to ρK without increasing the other losses, the excess term replaces $N_{\mathrm{out}}(K)$ by $N_{\mathrm{out}}(\rho K)$. Since $N_{\mathrm{out}}(K) \le \frac{1}{3}(n/K)^2$, its envelope shrinks by ρ^{-2}; doubling K leaves less than one quarter of the previously uncovered calls when $K \ge M$.

The certificate improves when adaptation lowers the relevant equation 34 terms of wide recombination calls, though the corollary does not prove optimization achieves that reduction or that a smaller upper bound entails smaller actual error — the adaptation experiment of Section 6 tests it directly.

Token weighting and shared-parameter changes. The ρ^{-2} envelope counts uncovered frames and assumes constant excess *summed loss per frame*. If a frame has $T_C(k) = \Theta(k)$ supervised tokens and a constant excess loss Δ per token, its excess contribution is instead proportional to Δk. With j_K as above,

$$\sum_{d=0}^{j_K - 1} 4^d k_d = \frac{n}{2}(2^{j_K} - 1) \le \frac{n^2}{2K},$$

so the corresponding envelope scales as K^{-1}. Neither rate is a universal law for observed accuracy. Because the same parameters serve all frames, changes to BASE and SPLIT losses must also be included when comparing pre- and post-adaptation certificates.

C PROOFS

Proof of Lemma 1. The products in column r are the pairs (i, j) with $i + j = r$ and $0 \le i, j \le n - 1$, that is, i ranges over $\max(0, r - n + 1) \le i \le \min(r, n - 1)$. For $r \le n - 1$ this is $0 \le i \le r$, which has $r + 1$ elements; for $r \ge n$ it is $r - n + 1 \le i \le n - 1$, which has $2n - 1 - r$ elements. The carry $\kappa_r = \lfloor u_{r-1}/10 \rfloor$ with u_{r-1} a function of the products in column $r - 1$ and of κ_{r-1}; unrolling the recursion from $\kappa_0 = 0$ shows that κ_r is a function of all products in columns $0, \ldots, r - 1$. □

Proof of Proposition 2. Fix a correct frame prompt and continuation $z = (z_1, \ldots, z_T)$. Let F be the event that greedy decoding does not reproduce this continuation exactly. On F, let t be the first position at which the decoded token $\widehat{z}_t$ differs from z_t. The generated prefix before t is $z_{<t}$, so the distribution used by greedy decoding is the same teacher-forced distribution that appears in equation 34. Since greedy decoding selected $\widehat{z}_t \neq z_t$,

$$f_\theta(z_t \mid p, z_{<t}) \le f_\theta(\widehat{z}_t \mid p, z_{<t}).$$

The two probabilities sum to at most one, hence the correct-token probability is at most $1/2$ and its negative log-likelihood is at least $\log 2$. Therefore, for every correct frame,

$$\mathbf{1}\{F\} \le \frac{\Lambda_Q(v; \theta)}{\log 2}. \quad (38)$$

The argument includes early termination and malformed generations: each has a first position at which a generated token or end marker differs from the target.

Now evaluate every frame on the ideal recursion tree $\mathcal{T}(A, B)$. If all ideal SPLIT frames are correct, induction from the root shows that the executor issues the correct child prompts. If every ideal BASE frame is also correct, every leaf returns its exact product. Finally, if all ideal COMBINE frames are correct, induction from the leaves shows that every internal node returns the block-identity sum and the root returns AB. Thus

$$\mathbf{1}\{R_\theta(A,B) \neq AB\} \le \sum_{v \in \mathcal{L}} \mathbf{1}\{F_{\text{BASE},v}\} + \sum_{v \in \mathcal{I}} \big(\mathbf{1}\{F_{\text{SPLIT},v}\} + \mathbf{1}\{F_{\text{COMBINE},v}\}\big).$$

Substituting equation 38 proves the pointwise bound. Taking expectation gives the probability claim; no independence is used. For width at most $n = M2^D$, the worst-case tree has at most 4^D leaves and $(4^D - 1)/3$ internal nodes. Applying the uniform expected-loss bounds and clipping at one gives equation 36. □

Proof of Corollary 3. At depth d the balanced tree has 4^d COMBINE calls of width $k_d = n/2^{d+1}$. Such a call is outside the assumed low-loss width range exactly when

$$d < \log_2 \frac{n}{2K}.$$

After truncation to the D internal levels, these are precisely $d = 0, \ldots, j_K - 1$. Their number is

$$N_{\mathrm{out}}(K) = \sum_{d=0}^{j_K-1} 4^d = \frac{4^{j_K} - 1}{3}.$$

Use λ_0 for all $(4^D - 1)/3$ COMBINE calls in the uniform part of Proposition 2, then add $\lambda_1 - \lambda_0$ for each of the $N_{\mathrm{out}}(K)$ uncovered calls. This gives equation 37.

If $j_K = 0$, the quadratic-envelope claim is trivial. Otherwise $j_K \le \lceil \log_2(n/2K) \rceil$, so $4^{j_K} \le 4(n/2K)^2 = (n/K)^2$ and hence $N_{\mathrm{out}}(K) \le \frac{1}{3}(n/K)^2$. Replacing K by ρK divides this envelope by ρ^2. If $K \ge M$ and $j_K > 0$, doubling K decreases j_K by one, and the fraction of uncovered calls remaining is

$$\frac{4^{j_K - 1} - 1}{4^{j_K} - 1} < \frac{1}{4}.$$

The assumptions on adaptation ensure that no other term in the certificate increases. □

D PROOFS AND EXTENSIONS OF THE DIGITWISE ERROR LAW

D.1 EXACT RESIDUE CONVOLUTION AND UNEQUAL TERM ERRORS

Let U_b be uniform probability on $\mathbb{Z}_b$, let δ_0 be unit mass at zero, and use $*$ for convolution modulo b. The symmetric residue law has the algebraic representation

$$\mu_\varepsilon = \alpha(\varepsilon)\delta_0 + (1 - \alpha(\varepsilon))U_b. \tag{39}$$

For $\varepsilon > (b-1)/b$, the coefficient of δ_0 is negative; the identity still holds for the probability measure on the left, but is not a convex-mixture interpretation. All monotonicity statements in the main text restrict α to $(0, 1)$.

Proof of Theorem 1. The identities $U_b * P = U_b$ for every probability measure P and $\delta_0 * P = P$ give, by induction,

$$\mu_{\varepsilon_{r,1}} * \cdots * \mu_{\varepsilon_{r,w_r}} = A_r\delta_0 + (1 - A_r)U_b.$$

This also holds for $w_r = 0$. Independence justifies computing the sum distribution by convolution. If $\nu_r = \mathcal{L}(C_r)$, then

$$\mathcal{L}\left(C_r + \sum_j E_{r,j} \bmod b\right) = A_r\nu_r + (1 - A_r)U_b.$$

The digit is correct exactly when this residual is zero. Its probability is $A_r q_r + (1 - A_r)/b$, yielding equation 7. The decimal specialization sets $b = 10$, $q_r = 1$, and all term rates equal to ε. □

The proof needs independence of the residuals and carry within a specified column. It does not need independence of errors across columns, independence between different input examples, or uniform target digits. In particular, true partial products can share operand digits across columns. These facts do not imply that the within-column residual independence premise holds for a learned network.

D.2 THE CENTRAL PEAK, BAND WIDTH, AND CARRY ASYMMETRY

Proof of Corollary 1. For fixed $q > 1/b$ and $0 < \alpha < 1$, equation 7 gives $p_r^\star = (b-1)/b - (q - 1/b)\alpha^{w_r}$. Subtract the corresponding expression at s to obtain equation 9. The triangular profile has its unique maximum $w_n(n-1) = n$, so this column has the maximum reference error. Its accuracy $1/b + (q - 1/b)\alpha^n$ decreases to $1/b$ with n when the channel parameters are held fixed.

With correct carries, accuracy is at most a exactly when

$$\alpha^{w_n(r)} \le \frac{a - 1/b}{1 - 1/b}.$$

Both sides are in $(0, 1)$; taking logarithms and dividing by $\log \alpha < 0$ gives $w_n(r) \ge h_a$. If $h_a \le n$, the qualifying columns are the integers $h_a - 1 \le r \le 2n - 1 - h_a$, a set of size $2(n - h_a) + 1$. Otherwise the set is empty. □

If only the carry reliability changes between two equal-width columns, their accuracy difference is $(q_r - q_s)A$ when their common factor $A \ge 0$. Thus smaller carry reliability can make a column on the upper side of the profile harder. This is a conditional mechanism for asymmetry, not a proof that q_r decreases with significance in the trained model. For $w_r = 0$, the formula reduces to accuracy q_r, as required for the carry-only top output column. At the least significant column, the true carry is zero; a reference that handles initialization exactly has $q_0 = 1$.

For unequal term rates with $0 \le \varepsilon_{r,j} < (b-1)/b$, define the effective residual load

$$L_r = -\sum_{j=1}^{w_r} \log \alpha(\varepsilon_{r,j}).$$

Then accuracy is $1/b + (q_r - 1/b)e^{-L_r}$. Counting terms alone is sufficient only when their quality is comparable. At fixed carry reliability $q > 1/b$, adding a term with $0 < \varepsilon < (b-1)/b$ increases reference error by $(q - 1/b)A[1 - \alpha(\varepsilon)]$, where A is the preceding product of factors. A zero factor is handled directly in the original product formula.

D.3 APPROXIMATE CHANNELS AND ACTUAL GREEDY PREDICTIONS

Proof of Theorem 2. Fix a column and suppress its subscript. Couple C with a reference C° by setting them equal. Recursively construct independent reference residuals $E_j^\circ \sim \mu_{\varepsilon_j}$, independent of C°. As long as preceding residuals agree, use a maximal coupling between the actual conditional law of E_j given its history and μ_{ε_j}. Equation equation 11 bounds the conditional disagreement probability by δ_j. After any disagreement, complete the coupling with the correct marginals. The reference draw at each step has its prescribed law conditional on the joint past, so the reference residuals have the required independence. The union bound over the first disagreement gives

$$\Pr\big((C, E_1, \dots, E_w) \ne (C^\circ, E_1^\circ, \dots, E_w^\circ)\big) \le \min\{1, \textstyle\sum_j \delta_j\}.$$

Consequently the probability that the actual proposed arithmetic residual is zero differs from the independent reference probability by at most $\min\{1, \sum_j \delta_j\}$. This is a statement about the residual sum, so possible dependence on the true digit creates no additional step. Theorem 1 computes the independent reference error probability $p^\star$.

For any two predictions u, v and target y,

$$\big|\mathbf{1}\{u \ne y\} - \mathbf{1}\{v \ne y\}\big| \le \mathbf{1}\{u \ne v\}.$$

Apply this to $\widehat{y}^{\mathrm{TF}}$ and $\widetilde{y}$ and take expectation. Their error probabilities differ by at most η. Combining both inequalities proves equation 13, including nondigit predictions and clipping to the probability interval.

On an input where the generated prefix before the specified token is correct, teacher-forced and free-running greedy decoding evaluate the same decoder on the same prompt and prefix with the same tie-breaking. Their predictions therefore agree. The same indicator inequality gives $|p_\theta^{\mathrm{FR}} - p_\theta^{\mathrm{TF}}| \le \beta$, proving equation 14. This argument does not condition the residual law on survival of a correct prefix. Such conditioning could change that law and would require a separate analysis. Subtract the upper bound at s from the lower bound at r and use equation 9 to obtain equation 15. Its unclipped form is valid even if negative, in which case it gives no positive ordering guarantee. □

A related first-error bound is

$$\beta_r \le \min\left\{1, \sum_{t<\tau(r)} \Pr(\widehat{z}_t^{\mathrm{TF}} \neq z_t)\right\}, \tag{40}$$

where $\tau(r)$ is the target token index in the chosen serialization, and the sum includes all preceding control and digit tokens. Indeed, a first free-running error occurs on a correct prefix and is therefore a teacher-forced error at that token. This controls whether any earlier error occurred; it does not assert that all later digits remain wrong. The paper's original flat profiles exclude malformed strings. Applying the theorem to that selected population would require its assumptions to hold on the selected population. For validation we instead require an unconditional error profile, reporting formatting failures separately.

D.4 Transferring a reference error through the full decoder

To justify equation 16, fix an input outside $J_r \cup U_r$ with $m_r > 2u_r$. For every competing vocabulary token a,

$$z_{r,\widetilde{y}_r} - z_{r,a} \ge \bar{z}_{r,\widetilde{y}_r} - \bar{z}_{r,a} - 2u_r \ge m_r - 2u_r > 0.$$

Thus the actual greedy token equals $\widetilde{y}_r$. Taking the contrapositive proves that disagreement is contained in the event on the right side of equation 16, and taking probabilities proves the claim. The argument does not require $\widetilde{y}_r = y_r$. It can therefore support a lower error bound when an erroneous arithmetic reference is both identified and stable under the perturbation. The inequality alone neither constructs that reference nor establishes the channel assumptions.

For a fixed finite checkpoint, actual-reference logit differences and token disagreements can be measured directly on held-out examples. Uniform norm certificates are stronger but can be conservative. Averaging small disagreements does not validate the uniform conditional total-variation bounds in equation 11; separate channel evidence or a proved structural model is needed.

D.5 Independent nonsymmetric channels

Symmetry gives the exact monotone law, but independent nonsymmetric channels can still be analyzed without treating all nonzero errors as equivalent. For a law μ on $\mathbb{Z}_b$, let

$$\widehat{\mu}(t) = \sum_{e=0}^{b-1} \mu(e) \exp(2\pi \mathrm{i} te/b), \qquad 0 \le t < b.$$

Proposition 3 (Residue mixing for general independent errors)**.** *Suppose the carry has law ν and is independent of mutually independent term residuals with laws $\mu_1, \dots, \mu_w$. Then*

$$\Pr(\widetilde{y} = y) = \frac{1}{b} \sum_{t=0}^{b-1} \widehat{\nu}(t) \prod_{j=1}^{w} \widehat{\mu}_j(t). \tag{41}$$

If $|\widehat{\mu}_j(t)| \le \rho < 1$ for every j and every $t \neq 0$, then

$$\left|\Pr(\widetilde{y} = y) - \frac{1}{b}\right| \le \frac{b-1}{b}\rho^w. \tag{42}$$

In particular, error is at least $(b-1)(1-\rho^w)/b$. This increasing lower bound and convergence to $1/b$ accuracy do not themselves imply that actual error is monotone at every finite width.

Proof. For any residue z, character orthogonality gives $\mathbf{1}\{z = 0\} = b^{-1} \sum_{t=0}^{b-1} \exp(2\pi \mathrm{i} tz/b)$. Apply it to $z = C + \sum_j E_j$ and take expectation. Independence factors the expectation and proves equation 41. The $t = 0$ term equals $1/b$. Since $|\widehat{\nu}(t)| \le 1$, the triangle inequality bounds the remaining $b-1$ terms by $(b-1)\rho^w/b$. Conjugate terms pair, so the expression is real. The error lower bound follows by subtracting the accuracy upper bound from one. □

Why conditions are necessary. Three elementary examples prevent stronger unconditional claims. First, if every decimal contribution has deterministic residual 1 and the carry is correct, the digit is wrong for $w = 1, \ldots, 9$ but correct again at $w = 10$. Second, independent errors uniform on $\{0, 5\}$ remain within that subgroup; the sum has zero residue with probability $1/2$ for every $w \geq 1$, rather than approaching $1/10$. Their nontrivial Fourier coefficients are not all smaller than one. Third, dependent residuals $E_2 = -E_1 \bmod b$ cancel exactly even if both contributions are often wrong. Bounded width is therefore a plausible computational advantage, but width alone cannot force a universal error law.

E Proofs of locality and decoder stability

E.1 Arithmetic locality and the legal RAT support

Proof of Theorem 3. Since at most q digits contribute to a column, $0 \leq S_r \leq q(b-1)$. If $0 \leq \kappa_r \leq q-1$, then

$$0 \leq \kappa_{r+1} \leq \left\lfloor \frac{q(b-1)+q-1}{b} \right\rfloor = q-1.$$

Induction starts at $\kappa_0 = 0$. The preceding column satisfies

$$\frac{S_{r-1} - y_{r-1}}{b} = \kappa_r - \frac{\kappa_{r-1}}{b}.$$

Because $q \leq b$, the final fraction belongs to $[0, 1)$. Taking ceilings proves the first identity in equation 18; substituting it into equation 17 proves the second. For $r = 0$, the synthetic preceding state gives carry zero. Each of the two sums uses at most q numerical inputs, and only one preceding output digit is used.

For the RAT specialization, take a split of width k with high-part width $h \leq k$. Then

$$A_\ell, B_\ell < b^k, \qquad A_h, B_h < b^h,$$

and the four aligned addends are

$$P_{\ell\ell}, \qquad b^k P_{\ell h}, \qquad b^k P_{h\ell}, \qquad b^{2k} P_{hh}.$$

The low product is less than b^{2k} and hence has no digits at or above column $2k$. The shifted high product has no digits below column $2k$. They cannot both be active at one column. Adding the two middle products yields at most three active addends. Apply the first part with $q = 3, b = 10$. At a switch between active addends, the preceding and current columns can use different sets of three inputs; their union still contains at most six numerical digit slots, plus the preceding output digit. □

Remark 1 (Scope of the carry premise). The condition $q \leq b$ is substantive. In radix ten, $S_{r-1} = 0$ and an incoming carry of ten give $y_{r-1} = 0$ and outgoing carry one, whereas $\lceil (S_{r-1} - y_{r-1})/10 \rceil = 0$. Such an incoming state is outside the theorem's carry range. The local formula also need not recover the true carry when y_{r-1} has already been decoded incorrectly.

Proposition 4 (A bounded carry can be reconstructed from a short window)**.** *For a nonnegative column-sum recurrence of the form equation 17, suppose $\kappa_i \leq C$. If $b^a > C$, then for $r \geq a$,*

$$\kappa_r = \left\lceil \frac{\sum_{j=0}^{a-1} b^j (S_{r-a+j} - y_{r-a+j})}{b^a} \right\rceil. \tag{43}$$

Proof. Multiply $S_i + \kappa_i = y_i + b\kappa_{i+1}$ by $b^{i-(r-a)}$ and sum over $i = r-a, \ldots, r-1$. The carry terms telescope to

$$\sum_{j=0}^{a-1} b^j (S_{r-a+j} - y_{r-a+j}) = b^a \kappa_r - \kappa_{r-a}.$$

The claim follows from $0 \leq \kappa_{r-a} < b^a$. □

For a hypothetical LSD-first flat multiplication, the raw column sum is at most $n(b-1)^2$ and induction gives $\kappa_r \leq n(b-1)$. Thus $a = \lceil \log_b(n(b-1)+1) \rceil$ suffices once the raw sums and preceding correct output digits are available. This is not a Transformer lower bound and does not apply directly to our MSD-first flat baseline. It does correct an overly literal claim that computing one carry necessarily requires direct access to every earlier digit product.

E.2 A ROBUST FINITE ARITHMETIC TABLE

Proof of Lemma 2. For an admissible state s, its distance to its own class is zero. Any state of a different class is a different integer vector, so its maximum-norm distance from s is at least one. This gives the claimed logit margin. Distance to a nonempty set is 1-Lipschitz; taking its negative preserves this property. Hence for $\|\widehat{s} - s\|_\infty \le \eta$, the correct logit is at least $-\eta$ and each incorrect logit is at most $-1 + \eta$. Their difference is at least $1 - 2\eta$, which is positive for $\eta < 1/2$.

Each class is nonempty because $(0, 0, a)$ is admissible for each digit a. Finally, absolute values, finite maxima, and finite minima have exact finite ReLU realizations, using

$$|x| = \mathrm{ReLU}(x) + \mathrm{ReLU}(-x), \quad \max(x, y) = \mathrm{ReLU}(x - y) + y, \quad \min(x, y) = -\max(-x, -y).$$

Apply these identities to the finite minimum of the three-dimensional maximum-norm distances in equation 21. The finite state set depends on b, q only. No claim about a particular six-layer, width-256 realization follows from this generic construction. □

E.3 RETRIEVAL AND SCORE GAPS

Proof of Theorem 4. Let d be a required digit and $\widehat{d}$ its attention-weighted scalar retrieval. The allowed contribution is $(1 - \epsilon)d$. Since all outside values lie in $[0, b - 1]$,

$$|\widehat{d} - d| = \left| \sum_{j \notin J} a_j (v_j - d) \right| \le (b - 1)\epsilon.$$

There are at most q active retrieved digits at either column, so the absolute error in each column sum is at most $q(b - 1)\epsilon$. Missing zero slots contribute no error by premise. The previous output component is exact, or has error at most η in the stated extension. Lemma 2 now proves the one-step claim.

For the first digit, use the exact synthetic preceding state. If all earlier digits are correct, the next step has the correct preceding digit and the one-step argument applies. Induction yields every arithmetic digit; the separate structural and termination premises complete the frame.

For the softmax bound, let u be the largest allowed score. The allowed exponential sum is at least e^u, while the outside sum is at most $me^{u-\Delta}$. The ratio of the outside sum to the allowed sum is at most $me^{-\Delta}$. Since $x/(1 + x)$ is increasing, this proves equation 23. Put $a = 2q(b - 1)$ and $x = me^{-\Delta}$. The strict inequality $x/(1 + x) < 1/a$ is equivalent to $(a - 1)x < 1$, which gives equation 24. If $m = 0$, the outside mass is identically zero. □

Proof of Proposition 1. Use one relevant key with value one and score Δ, and m irrelevant keys with value zero and score zero. The retrieved scalar is

$$z = \frac{e^\Delta}{e^\Delta + m}.$$

Let the two output logits be z for token one and $1 - z$ for token zero. The target is one. Greedy decoding is correct when $z > 1/2$, equivalently $m < e^\Delta$, and incorrect when $m > e^\Delta$. The equality case is a tie and is intentionally excluded. The relevant input set and arithmetic operation do not change with m. □

The bound on leakage is tight for this score pattern. For a prescribed leakage budget ϵ_*, keeping the allowed mass at least $1 - \epsilon_*$ against equal-score distractors requires

$$\Delta \ge \log m + \log \frac{1 - \epsilon_*}{\epsilon_*}.$$

This is a necessity statement for this head and adversarial distractor pattern, not a necessity theorem for every possible neural implementation of arithmetic.

E.4 MULTILAYER PERTURBATION

Proof of Theorem 5. At a fixed input H, denote a head's allowed and outside conditional averages by u_{in} and u_{out}. If its outside mass is ϵ, then its actual average is $(1-\epsilon)u_{\text{in}} + \epsilon u_{\text{out}}$ and its restricted average is u_{in}. Both conditional averages have norm at most $V_{\ell h}$, so their difference has norm at most $2V_{\ell h}\epsilon$. If the outside mass is zero, the difference is zero. After head projections and summation,

$$\left\|A_\ell(H^{\ell-1}) - \bar{A}_\ell(H^{\ell-1})\right\|_{\max,2} \le \delta_\ell. \tag{44}$$

The allowed set is nonempty and softmax scores are finite, so its mass is positive and the restricted conditional average is well-defined.

Let $e_{\ell-1} = \left\|H^{\ell-1} - \bar{H}^{\ell-1}\right\|_{\max,2}$. Insert and subtract $\bar{A}_\ell(H^{\ell-1})$. The triangle inequality, equation 44, and the Lipschitz premise give

$$\begin{aligned}&\left\|[H^{\ell-1} + A_\ell(H^{\ell-1})] - [\bar{H}^{\ell-1} + \bar{A}_\ell(\bar{H}^{\ell-1})]\right\|_{\max,2}\\ &\qquad \le (1 + C_\ell)e_{\ell-1} + \delta_\ell.\end{aligned}$$

Applying F_ℓ multiplies this bound by at most M_ℓ. Since the two executions have identical embeddings, $e_0 = 0$. Induction establishes equation 27; unrolling it yields equation 28. The output-map Lipschitz bound proves equation 29.

For any incorrect token a, the actual logit difference satisfies

$$z_y - z_a \ge \bar{z}_y - \bar{z}_a - 2\left\|z - \bar{z}\right\|_\infty \ge \gamma - 2K_{\text{out}}E_L.$$

It is strictly positive under equation 31. This comparison includes control tokens and end markers as competing tokens even when the target is an arithmetic digit. The score-gap upper bounds on leakage follow from the same softmax calculation as in Theorem 4, at the actual layer inputs. □

An explicit restricted-attention constant. For one head with key dimension a_h, suppose its positionwise maps Q, K, V have Lipschitz constants L_Q, L_K, L_V, and norms bounded by B_Q, B_K, B_V on the relevant domain. Include any pre-normalization in these maps. Scaled dot-product scores, with fixed position-dependent biases allowed, satisfy

$$|s_{tj}(H) - s_{tj}(H')| \le D_h\left\|H - H'\right\|_{\max,2}, \qquad D_h = \frac{L_Q B_K + B_Q L_K}{\sqrt{a_h}}.$$

On any fixed allowed set, softmax satisfies

$$\left\|\text{softmax}(u) - \text{softmax}(v)\right\|_1 \le 2\left\|u - v\right\|_\infty.$$

Indeed, its directional derivative has ith component $p_i(w_i - \sum_j p_j w_j)$, whose absolute sum is at most $2\left\|w\right\|_\infty$; integration along the segment proves the inequality. Separating changes in the weights and values therefore bounds a head's restricted average by

$$(L_V + 2B_V D_h)\left\|H - H'\right\|_{\max,2}.$$

Consequently, a valid block constant is

$$C_\ell \le \sum_h \left\|O_{\ell h}\right\|_{2\to 2}\left(L_{V,\ell h} + 2B_{V,\ell h}\frac{L_{Q,\ell h}B_{K,\ell h} + B_{Q,\ell h}L_{K,\ell h}}{\sqrt{a_h}}\right). \tag{45}$$

For affine query, key, and value maps their Lipschitz constants are spectral norms; their biases must still be included in magnitude bounds. Layer-normalization constants require its positive numerical stabilizer or a proved variance lower bound. No favorable constant should be assumed from architecture names alone. The formula has no explicit key-count factor, but the domain bounds and measured leakage can depend on frame length.

E.5 COMPOSITION AND ARBITRARY PREFIXES

Proof of Corollary 2. Suppose the right side of equation 33 is zero. Every ideal SPLIT and BASE frame is then correct. Consider an ideal COMBINE frame. If its first erroneous generated token were a digit, its preceding prefix would be correct, and Theorem 5 would force that token to be correct,

a contradiction. If its first error were a control or end token, F_{other} would occur. Thus each ideal COMBINE frame is exact. The correct split frames establish the correct recursion tree; induction from the leaves through its combine frames returns AB. Taking the contrapositive gives the pointwise inequality, and expectation gives its probabilistic form. No independence between errors, frames, or repeated uses of shared parameters is used. □

What the certificate does not imply. The successful case is a sufficient condition. Violating equation 31 is not a proof of failure. A reference with negative correct-token margin cannot certify an actual correct prediction through this bound. Product cancellations can also allow a correct root despite erroneous intermediate outputs. Finally, a finite learned place-ID table and finite context range give a finite supported domain; arbitrary-length claims require a separate length-uniform addressing construction.

F RECURSION-TREE SIZES

For operands of width l and cutoff M, let $k_l = \lceil l/2 \rceil$ and $h_l = \lfloor l/2 \rfloor$. Among the four child pairs, three have width at most k_l and one at most h_l. The maximum numbers of BASE leaves and internal (SPLIT plus COMBINE) nodes satisfy $B(l) = 1$, $I(l) = 0$ for $l \le M$, and $B(l) = 3B(k_l) + B(h_l)$, $I(l) = 1+3I(k_l)+I(h_l)$ otherwise; the call count is $B+2I$ and, for $l = M2^D$, equals $(5\cdot 4^D - 2)/3$. Table 4 lists the values for $M = 2$.

Table 4: Worst-case recursion-tree sizes for cutoff $M = 2$.

n	BASE leaves	SPLIT = COMBINE calls	total calls
3, 4	4	1	6
6, 8	16	5	26
9	43	14	71
10	52	17	86
12, 14, 16	64	21	106
18	172	57	286

G COMBINE-ADAPTATION PROTOCOL

The intervention starts from the normally trained system and continues training on COMBINE frames only. Because RAT uses one shared Transformer, "COMBINE-stage adaptation" describes the adaptation data and objective, not a separate module. The $3\times$ and $4\times$ variants use COMBINE examples whose addends are 3 and 4 times as wide as in normal training, and are evaluated on the equal-length grid $n \in \{8, 10, \ldots, 18\}$ with 100 products per cell and exact integer match.

H ADDITIONAL RESULTS

Table 5: Accuracy (%) of the flat model by interaction width w, pooled over $n \in \{2, 3, 4, 5\}$. Columns are split by whether they lie at or below the peak column of their product ("lower half", peak included) or above it; the top column has no digit products, only a carry. In parentheses, the number of columns averaged.

w	lower half	upper half	top column
0	–	–	76 (4)
1	82 (4)	48 (4)	–
2	61 (4)	35 (3)	–
3	47 (3)	24 (2)	–
4	26 (2)	12 (1)	–
5	10 (1)	–	–

Table 6: Per-seed exact match (%) of the diagnostic configuration, 1000 products per length.

n	seed 42	seed 43	seed 44	mean
3	99.3	99.1	99.4	99.3
4	99.6	99.8	99.9	99.8
5	82.8	84.3	83.7	83.6
6	94.5	94.6	95.1	94.7
7	93.7	94.1	94.2	94.0
8	97.1	97.6	96.6	97.1
9	54.3	50.8	51.7	52.3
10	50.6	48.6	52.5	50.6

I COMPLETE FRAME ENCODERS AND INFERENCE SERIALIZATION

This section records the token-level procedures underlying the compact description in Section 4. EMITNUM writes digits LSD-first and assigns decimal place IDs $o + i$. EMITSCALAR is used for control metadata such as k and uses $\pi = 0$. The training origin o is randomized over the legal place-ID range; inference uses $o = 1$.

Algorithm 2 Frame dispatcher and shared emit primitives

function ENCODEFRAME($q, \mathcal{C}$)
  if $q = \text{base}$ **then**
    sample operand lengths in $\{1, \dots, M\}$ and operands A, B; $L \leftarrow \max\{\ell(A), \ell(B)\}$
    $n_{\text{need}} \leftarrow 2L$
  else if $q = \text{split}$ **then**
    sample a legal split width L from the configured split generator; sample A, B of length L
    $n_{\text{need}} \leftarrow L$
  else
    sample k and partial products from the configured combine generator
    $n_{\text{need}} \leftarrow 2k + \ell(P_{hh})$
  end if
  $o \sim \text{Unif}\{1, \dots, \Pi_{\max} - n_{\text{need}}\}$; initialize (x, π, m) empty
  dispatch to BASE, SPLIT, or COMBINE below
  return (x, π, m)
end function

procedure EMIT(t, p, train)
  append token t, place ID p, and mask bit $\mathbf{1}[\text{train}]$
end procedure
procedure EMITNUM(n, o, train; optional width w)
  $d \leftarrow \text{LSD}(n, w)$ if w is supplied, otherwise $\text{LSD}(n)$
  for $i = 0, \dots, |d| - 1$ **do**
    EMIT($d_i, o + i, \text{train}$)
  end for
end procedure
procedure EMITSCALAR(n, train)
  for digits d_i of n in most-significant-first order **do**
    EMIT($d_i, 0, \text{train}$)
  end for
end procedure

J LOGGED HYPERPARAMETERS OF THE DIAGNOSTIC CONFIGURATION

The following values describe the learned-router configuration used in Sections 6.1 and 6.2; fields that were not logged are not inferred. They are not the settings of the adaptation study of Section 6.3.

Algorithm 3 Complete frame encoders

procedure BASE(A, B, L, o)
  emit $\langle\texttt{bos}\rangle$ LSD(A) $*$ LSD(B) $=$ as prompt
  for $j = 0, \ldots, L-1$ **do**
    if $j > 0$ **then**
      EMIT($+, 0, 1$)
    end if
    EMITNUM($Ab_j, o+j, 1;\ w = L+1$)
  end for
  EMIT($=, 0, 1$); EMITNUM($AB, o, 1;\ w = 2L$); EMIT($\langle\texttt{eos}\rangle, 0, 1$)
end procedure

procedure SPLIT(A, B, o)
  $L \leftarrow \max\{\ell(A), \ell(B)\}$; $k \leftarrow \lceil L/2 \rceil$
  $A_{\text{lo}} \leftarrow A \bmod 10^k$; $A_{\text{hi}} \leftarrow \lfloor A/10^k \rfloor$
  $B_{\text{lo}} \leftarrow B \bmod 10^k$; $B_{\text{hi}} \leftarrow \lfloor B/10^k \rfloor$
  emit $\langle\texttt{bos}\rangle$ LSD(A) $*$ LSD(B) $=$ as prompt
  EMIT(SPLIT, $0, r$) ▷ $r = 1$ when the router is learned; $r = 0$ when Python preselects the frame
  EMITSCALAR($k, 1$)
  for (c, U, V) in $(1, A_{\text{lo}}, B_{\text{lo}}), (2, A_{\text{lo}}, B_{\text{hi}}), (3, A_{\text{hi}}, B_{\text{lo}}), (4, A_{\text{hi}}, B_{\text{hi}})$ **do**
    EMIT(CALL, $0, 1$); EMIT($c, 0, 1$)
    EMITNUM($U, o, 1$); EMIT($|, 0, 1$); EMITNUM($V, o, 1$)
  end for
  EMIT($\langle\texttt{eos}\rangle, 0, 1$)
end procedure

procedure COMBINE($P_{\ell\ell}, P_{\ell h}, P_{h\ell}, P_{hh}, k, o$)
  emit $\langle\texttt{bos}\rangle$ COMBINE digits(k) $|$ as prompt
  for (P, s) in $(P_{\ell\ell}, 0), (P_{\ell h}, k), (P_{h\ell}, k), (P_{hh}, 2k)$ **do**
    EMITNUM($P, o+s, 0$); emit delimiter $|$ except after the last partial
  end for
  EMIT($=, 0, 0$)
  $S \leftarrow P_{\ell\ell} + (P_{\ell h} + P_{h\ell})10^k + P_{hh}10^{2k}$
  EMITNUM($S, o, 1$); EMIT($\langle\texttt{eos}\rangle, 0, 1$)
end procedure

The configuration is logged as a single 10,000-update run; stage-specific step counts were not recorded, so we do not describe a multi-stage curriculum.

K WORKED EXAMPLE: 1234 TIMES 5678

Let $A = 1234$, $B = 5678$, $L = 4$, and $k = 2$. Then

$$A_{\text{hi}} = 12,\ A_{\text{lo}} = 34,\ B_{\text{hi}} = 56,\ B_{\text{lo}} = 78,$$

so

$$P_{\ell\ell} = 34 \cdot 78 = 2652, \quad P_{\ell h} = 34 \cdot 56 = 1904, \quad P_{h\ell} = 12 \cdot 78 = 936, \quad P_{hh} = 12 \cdot 56 = 672,$$

and

$$1234 \cdot 5678 = 2652 + (1904 + 936)10^2 + 672\,10^4 = 7{,}006{,}652.$$

One BASE leaf. For 34×78, the scratchpad emits the shifted partial products $34 \cdot 8 = 272$ and $34 \cdot 7 = 238$, represented as $272 + 2380$, followed by 2652. Decimal place IDs shift the second partial by one column.

SPLIT. The learned-router prompt for 1234×5678 emits SPLIT, $k = 2$, and the four calls

$$(34, 78), \quad (34, 56), \quad (12, 78), \quad (12, 56).$$

Algorithm 4 Full learned-router recursive executor

function MULTIPLY(A, B)
  form prompt $x = \langle \texttt{bos} \rangle \, \mathrm{LSD}(A) * \mathrm{LSD}(B) =$ with local origin $o = 1$
  $\tilde{x} \leftarrow$ greedy continuation of f_θ
  if first generated token is `SPLIT` **then**
    parse k and $(U_c, V_c)_{c=1}^{4}$; if malformed, **return** $\perp$
    for $c = 1, \ldots, 4$ **do**
      $P_c \leftarrow$ MULTIPLY(U_c, V_c); if $P_c = \perp$, **return** $\perp$
    end for
    return COMBINEHOP(P_1, P_2, P_3, P_4, k)
  else
    parse the integer after the final $=$ in the BASE continuation; if malformed, **return** $\perp$
    return parsed integer
  end if
end function

function COMBINEHOP($P_{\ell\ell}, P_{\ell h}, P_{h\ell}, P_{hh}, k$)
  serialize the COMBINE prompt with shifts $0, k, k, 2k$ and origin $o = 1$
  greedily decode the sum; if malformed, **return** $\perp$
  return decoded integer
end function

Table 7: Logged settings of the diagnostic configuration.

Item	Value
architecture	6-layer decoder, $d = 256$, 8 heads, $d_{\mathrm{ff}} = 1024$
dropout	0
$\Pi_{\max}$	512
position coupling	true
ALiBi	false
cutoff M	2
learned router	true
frame mix [base, split, combine]	[0.15, 0.10, 0.75]
`split_train_max`	8
`split_copy_max`	32
`combine_k_max`	16
`combine_r_max`	32
updates	10,000
batch size	64
optimizer	AdamW, lr 8×10^{-5}, warmup 200, weight decay 0.01
training seeds	42, 43, 44
inference seed	20260816 (mps)
evaluation size	1000 products per length

Each numerical operand restarts at the same local place origin because π describes the operand's decimal role rather than its token-stream offset.

COMBINE. The four returned products are serialized with shifts $0, 2, 2, 4$. Thus 1904 and 936 share the same middle decimal lane and 672 begins four places above the low product. The network emits the LSD-first digits of $7{,}006{,}652$.

Recursive trace. The root call performs one SPLIT; each of its four child pairs already has width at most two and therefore invokes BASE; the four returned products are then passed to one COMBINE. Hence a 4×4 example uses six neural invocations in total. The 8×8 case repeats the same pattern one split level deeper, producing 16 BASE leaves and five internal split/combine nodes.

L ADDITIONAL TABLES AND FIGURES

Table 8: Digit-level accuracy of the flat model (percent). "Middle half" averages the positions $\lceil T/4 \rceil, \ldots, \lceil 3T/4 \rceil - 1$ of the $T = 2n$ output digits, counted from the most significant (so n positions, off-center by one for odd n).

Operands	Digits	N	Exact	First digit	Last digit	Middle half
2×2	4	100	27	73	97	52
3×3	6	100	19	82	92	67
4×4	8	99	4	74	99	42
5×5	10	94	0	74	38	11

Table 9: The three frames. "Width" describes arithmetic inputs; copying and structural decisions are distinguished. The local carry-recovery rule is given in Theorem 3.

Frame	Writes	Width	Trained on
BASE	scratchpad and product	$\leq M$ products per column	products up to 2×2
SPLIT	`SPLIT`, k, four child pairs	1 per copied digit; metadata separate	operands up to a maximum width
COMBINE	aligned sum of four addends	≤ 3 active legal addends; local carry recovery	real half-products, widths up to $k_{\max}$

Table 10: Exact match (%) for $n \times n$ multiplication, 100 products per length; in parentheses, the gain over normal training. Values are read from the labels of Figure 3.

Digits n	Normal	COMBINE 3×	COMBINE 4×
8	90	92 (+2)	95 (+5)
10	73	77 (+4)	86 (+13)
12	65	69 (+4)	80 (+15)
14	61	64 (+3)	73 (+12)
16	59	62 (+3)	70 (+11)
18	36	41 (+5)	55 (+19)
mean	64.0	67.5 (+3.5)	76.5 (+12.5)

Table 11: RAT exact match (%) by operand length n, diagnostic configuration, 1000 products per length, mean and range over three training seeds.

	n=3	4	5	6	7	8	9	10
mean	99.3	99.8	83.6	94.7	94.0	97.1	52.3	50.6
min–max	99.1–99.4	99.6–99.9	82.8–84.3	94.5–95.1	93.7–94.2	96.6–97.6	50.8–54.3	48.6–52.5

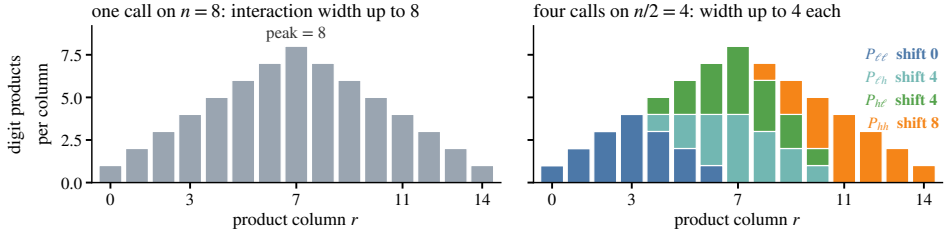

Figure 4: **Recursion cuts the interaction profile into smaller ones.** Left: digit products per column of an 8×8 product. Right: the block identity $AB = P_{\ell\ell} + (P_{\ell h} + P_{h\ell})10^k + P_{hh}10^{2k}$, $k = 4$, partitions these exactly into four 4×4 products at shifts $0, k, k, 2k$; each call sees half the height, combined by addition.