TALLY: LEARNED APPROXIMATELY, PROVED EXACTLY

Anonymous authors
Paper under double-blind review

ABSTRACT

Neural models can learn the correct computation and still fail to execute it reliably. TALLY studies this separation on finite group state tracking. An eight dimensional, input conditioned orthogonal transport makes latent group structure recoverable and raises gated DeltaProduct's pooled median accuracy at $16\times$ training length from 0.014 to 0.989 across twenty matched runs. Its operators encode the task's finite structure, but satisfy the group relations only approximately, so continuous execution eventually drifts. Given frozen operators and the recovered finite orbit, a conditional moment recursion predicts mean squared state deviation with 3.1% mean relative discrepancy across 120 combinations of checkpoint and length. On one frozen model, halving or doubling the operator defect moves the first sampled horizon below 90% transport accuracy from 8,192 to 32,768 or 2,048, while an exact realisation has no observed errors through 32,768. This controlled intervention supports a causal role for approximation error in the transport's failure. TALLY then recovers the latent finite action from checkpoints and unlabelled words, compiles it into discrete execution, and checks every transition and output against the task. The compiled scan makes no errors across 83,886,080 evaluated positions at length 1,048,576. Separate finite checks prove 53 extracted executors and readouts correct on every input at every length. The guarantee covers the frozen discrete executor and readout, not the unchanged neural decoder. Learned approximately, compiled and proved exactly.

1 INTRODUCTION

A neural model can contain the right computation and still be an unreliable executor. Small violations of a learned algebra may be invisible at the training length yet accumulate until the state crosses a decision boundary. Longer tests measure this failure but do not explain it or establish correctness at arbitrary length. We recover the learned finite computation, predict the growth of its continuous execution error, intervene on that error, and compile the recovered structure into a verified discrete executor.

We investigate this route on finite group state tracking: each input is a generator, and the target is the product of the prefix. The word problem for nonsolvable finite groups such as A_5 is complete for NC^1 and provides a demanding test (Barrington, 1989). Recent linear recurrent architectures have sufficient expressive power for these tasks (Grazzi et al., 2025; Siems et al., 2025), but representability does not ensure robust learned execution. Error control dynamics (Chung et al., 2026) and training state coverage (Buitrago Ruiz & Gu, 2025) help explain why accuracy can deteriorate outside the training regime.

TALLY adds an eight dimensional, input conditioned orthogonal transport to gated DeltaProduct. Across twenty matched runs over A_5, S_5, A_6 and A_7, pooled median accuracy at length 2,048 rises from 0.014 to 0.989 after training at length 128, but still decays at longer horizons. The apparent contradiction is the central observation: prefix operators cluster into the task's finite group while violating its relations slightly. Conditional moments predict the resulting state error, and scaling the defect in one frozen model changes its failure horizon. The learned computation is structurally right and operationally unstable.

TALLY separates three jobs and their inputs. *Learning* supplies continuous operators and a state readout. *Structural certification* receives the frozen checkpoint, token alphabet, unlabelled words

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.

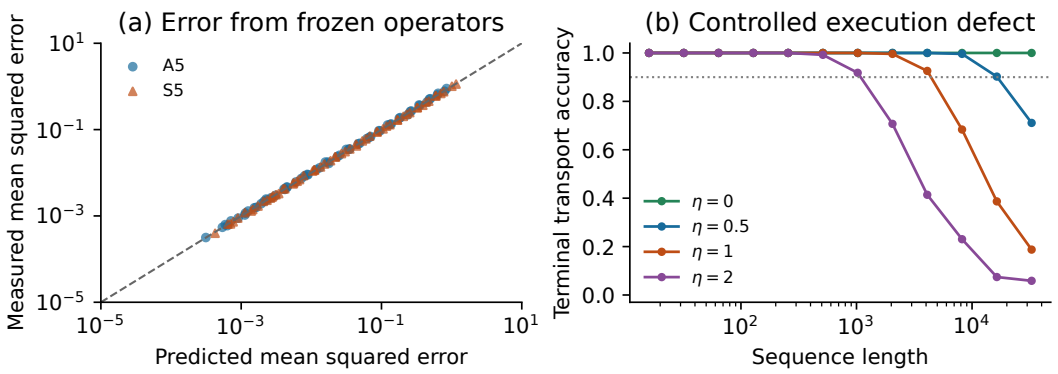

Figure 1: Quantifying and changing continuous failure. (a) Conditional moments versus 256 sampled words per checkpoint at each of twelve lengths (ten checkpoints). (b) On one calibration checkpoint, scale the operator defect by η, holding the start state and readout fixed. First sampled accuracy below 90% occurs at 32,768, 8,192 and 2,048 for $\eta = 0.5, 1, 2$; the exact representation ($\eta = 0$) has no observed errors. These are transport readouts, not full network predictions.

and numerical thresholds; it does not receive state labels or the task transition table. It writes a discrete certificate that is frozen and hashed. Only then does *task verification* receive that certificate and the known task specification, exhaustively checking transitions and readout. Only this final check establishes answer correctness. A structurally valid group may implement the wrong task, and a correct transport may feed a neural decoder that still fails.

The paper makes three contributions:

- A quantitative account of execution failure: finite conditional moments predict deviation from the learned discrete orbit, and a controlled change in operator defect shifts the failure horizon. The analysis separates predictable transport error from downstream decoding failure.
- A route from approximate learned operators to exact execution, through codebook closure or signed permutation maps, with exhaustive checks for 53 group executors and readouts. A matched black box comparison distinguishes transition recovery, output correctness and label free selection.
- Evidence about when structure is recoverable: a prototype loss improves full task state recovery; a GRU backbone supports the same pipeline; and reset/set tasks admit noninvertible finite executors. Changed generators and partial resets expose limits, rather than establishing general automaton learnability.

2 Task, Model and Evaluation

Task. For a finite group G with generators $g_1, \ldots, g_k$, inputs are tokens in $\Sigma = \{\epsilon, g_1, \ldots, g_k\}$, where ϵ is the identity. The target at position t is the class index of $x_t \cdots x_1$. We use A_5, S_5, A_6, A_7 (60–2,520 elements), with A_8, S_8 (20,160 and 40,320) as scale checks. Product, distractor and reset tasks test whether the learned state preserves the full task.

Architecture. The baseline is a two layer gated DeltaProduct (Siems et al., 2025), width 64, four heads and four Householder updates per token. Before each mixer, TALLY adds

$$s_t = Q(x_t)s_{t-1}, \qquad Q(x) = H(u(x))H(v(x)), \qquad H(w) = I - 2\frac{ww^\top}{\|w\|^2}. \tag{1}$$

Here $s_t \in \mathbb{R}^8$, s_0 is a learned normalised vector, and u, v are linear functions of the normalised layer input. A normalised linear readout adds the transport state to the mixer input. The two transports add 3,536 parameters (2.1% on A_5). At layer 0, the operator depends only on the token; later layer operators can depend on the preceding computation. Our finite state verification concerns layer 0.

task	$\|G\|$	DeltaProduct	**+ transport**	+ transport, $64\times$
A_5	60	0.016	**0.993**	0.658
S_5	120	0.009	**0.889**	0.365
A_6	360	0.002	**1.000**	0.726
A_7	2,520	0.284	**0.975**	0.544
pooled		0.014	**0.989**	

control	fits	$16\times$
transport ($\beta = 2$)	20/20	**0.989**
$\beta = 1.99$ (not orthogonal)	20/20	0.024
dense 8×8 transport	0/20	0.004
one reflection	9/10	0.016
LSTM (matched)	6/20	0.070

Table 1: Median accuracy on the final 64 positions at $16\times$ training length; the final left column is at $64\times$. TALLY wins 20/20 seed matched comparisons. The $\beta = 1.99$ control changes the reflection coefficient without changing parameter count; it sharply reduces the extrapolation gain.

Householder products are orthogonal in exact arithmetic for nonzero vectors. The implementation uses floating point normalisation with a small norm floor. Associativity of matrix multiplication permits a parallel prefix scan (Hillis & Steele, 1986); orthogonality preserves norms but neither creates that associativity nor eliminates numerical or learned relation error. The implemented doubling scan has logarithmic depth and $O(T \log T)$ work.

Protocol. Models train on 20,000 words of length 128 for twelve epochs, with five seeds per core group, batch 64, AdamW and learning rate 10^{-3}. S_8 uses batch 32 for memory. Checkpoint selection uses only validation loss at length 128. We distinguish final 64 accuracy, measured on the final 64 positions of each word, from all position accuracy. Pooled medians pool the runs, not the per task medians. Appendix A specifies seeds, sampling, extraction and readout construction.

3 Recovering structure and measuring execution error

Recovering latent structure. On unlabelled training length words, the second moment of layer 0 transport states has 3–6 dominant directions. Let V contain the selected eigenvectors. This is an empirically selected, approximately invariant subspace, not an assumed exactly invariant one; extraction records its leakage. Prefix operators restricted to it, $V^\top Q(x_t) \cdots Q(x_1) V$, form tight clusters. Leader clustering with radius 0.5 finds exactly $|G|$ clusters in all twenty confirmation models.

Labels are used only after this extraction to diagnose the recovered algebra. Majority labels define a bijection that preserves every one of the $|G|^2$ codebook products in 20/20 models. The character matches the standard representation in 18/20; the other cases match the three dimensional irreducible of A_5 and the standard plus trivial representation of A_6. These tests support recovery of task structure, even though the continuous operators do not satisfy its relations exactly.

Measuring the execution defect. Let c_g be the ambient state reconstructed from a certified discrete state g, and let s_t follow the frozen continuous token operators. For independent uniform tokens, conditional first moments and a second moment determine $D_t = \mathbb{E}\|s_t - c_{g_t}\|^2$ without sampling long sequences (Appendix C). This is standard moment propagation applied to the recovered computation, not an assumption of independent accumulated errors. Across ten A_5/S_5 checkpoints and twelve lengths, predicted and measured D_t differ by 3.1% on average (Figure 1). A failure threshold calibrated on two A_5 checkpoints predicts the first sampled horizon below 90% transport accuracy within a factor of two on 7/8 held out checkpoints; a constant horizon achieves 6/8. Thus state error is predictable, while the decoding threshold remains a coarse empirical model.

To test the mechanism, we align an exact orthogonal representation R_a with one learned model and write its polar projected operator as $Q_a(\eta) = R_a \exp(\eta E_a)$. The $\eta = 1$ arm reproduces the polar baseline; halving or doubling η changes the observed failure horizon by a factor of four. Propagation uses float64, and polar projection changes the learned operators by less than 4×10^{-7} in Frobenius norm. This supports learned execution defect as a cause in this transport, beyond arithmetic roundoff alone. It does not establish a universal inverse square law or isolate every source of full network error.

4 COMPILING APPROXIMATE STRUCTURE INTO EXACT EXECUTION

Codebook execution. Average each operator cluster and project it onto the orthogonal group, obtaining distinct matrices $C_1, \ldots, C_n$. Snap token operators to codebook entries and snap each product in the parallel scan. The following criterion concerns the resulting discrete multiplication, not exact equality of the stored matrices under ordinary multiplication.

Proposition 1 (Closure certificate)**.** *Let $C_1, \ldots, C_n$ be distinct orthogonal matrices, let $m = \frac{1}{2}\min_{i \neq j}\|C_i - C_j\|_F$, and let $c = \max_{i,j}\min_k \|C_i C_j - C_k\|_F$. If $c < m/2$, nearest codebook multiplication $i \circ j$ is associative and cancellative. Thus it defines a finite group, and any parenthesisation computes the same discrete prefix product.*

Proof. Orthogonal multiplication preserves Frobenius distance. Each of $C_{(i\circ j)\circ k}$ and $C_{i\circ(j\circ k)}$ is within $2c$ of $C_i C_j C_k$. Their distance is at most $4c < 2m$, so they are the same entry. If $i \circ j = i \circ k$, then $\|C_j - C_k\|_F = \|C_i C_j - C_i C_k\|_F \leq 2c < 2m$, hence $j = k$; right cancellation follows similarly. A finite nonempty associative cancellative set is a group. □

This theorem is over exact arithmetic. The implementation checks distances numerically and uses floating point matrices; the reported residuals are not interval certified bounds. For the twenty confirmation codebooks the closure residual is at least nine times below the threshold. Structural certification uses no task labels; matching the resulting group to the task is a separate check.

All twenty codebooks certify with $|G|$ entries. With codebook snapping, the full network makes no errors at any of 83,886,080 evaluated positions at length 1,048,576. At length 131,072, the unaltered models have pooled median final 64 accuracy 0.0107. The sensitivity and corruption checks are summarised in Appendix G. These are finite length measurements of the full network.

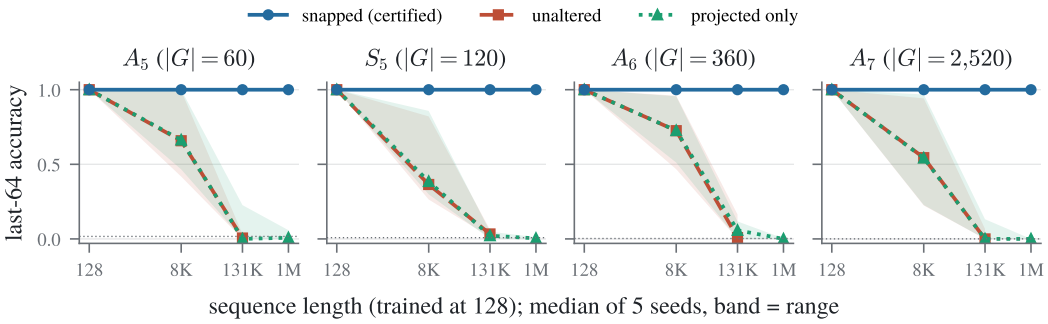

Figure 2: Accuracy on the final 64 positions for twenty confirmation models (five seeds per group), trained at length 128. Bands show the seed range. Codebook snapping ("regenerated") preserves accuracy through the tested lengths; the unaltered and projection only controls deteriorate. Zero errors at finite test lengths and the finite state correctness check are separate results.

Compact signed permutation execution. Closure checking forms $|G|^2$ matrix products and searches the codebook for each, requiring sufficient sample coverage. A smaller representation uses a frame $P = [p_1, \ldots, p_m]$ spanning the selected subspace. We search for axes such that each token maps p_j near $\sigma_{a,j} p_{\pi_a(j)}$, where $\sigma_{a,j} \in \{-1, 1\}$ and π_a is a permutation. Acceptance requires a bijection of axes and residual below half the signed axis separation. The accepted integer maps define an exact finite group of signed permutations.

Execution composes these maps, then applies them to fixed frame coordinates $c_0 = P^\dagger V^\top s_0$. This is an exact discrete computation followed by a floating point reconstruction. For an overcomplete approximate frame, the residual test alone does not prove an exact linear representation in the original subspace; we retain the frame coordinate state throughout composition. Task correctness is checked independently in Section 6.

task	$\lvert G\rvert$	fits at 128	**certified with $\lvert G\rvert$ states**
A_5	60	3/3	**3/3**
S_5	120	2/2	**2/2**
A_6	360	2/2	**2/2**
A_7	2,520	2/2	**2/2**
$A_5 \times A_5$	3,600	10/10	**7/10**
$A_5 \times \mathbb{Z}_7$	420	3/3	**0/3**
A_5 + 200 distractors	60	15/15	**6/15**
S_5 + 200 distractors	120	3/3	**3/3**
A_7 + 200 distractors	2,520	0/3	**0/3**

Table 2: Label based prototype loss, pooling the reported runs. For distractor tasks, success requires the full task state count and every distractor's certified map to be the identity. Standard product training certifies only one 60 state factor in 5/6 fitted runs; the other run does not certify.

Frames with 5–10 axes are found for all twenty confirmation models. Search and checking take 2.3–2.6 CPU seconds, and 2.8–3.6 seconds for the A_8, S_8 checks. These are measured runtimes, not a complexity bound independent of group size. Replacing the transport with frame execution leaves 51 full network errors among 136,478,720 tested positions, all in one A_7 model. In addition, 17/20 nonorthogonal $\beta = 1.99$ controls admit a frame, despite failing codebook certification; their full networks make 41 errors among 116,006,912 positions. Orthogonal training therefore helps but is not necessary for this extraction route.

Installing the executor. A straight through layer uses the certified maps in its forward pass and the continuous transport for gradients (Bengio et al., 2013). After 400 fine tuning steps (11–17 seconds), 20/20 confirmation models retain accuracy 1.000 at length 128 and make no errors in the million token evaluation. This installs discrete execution, rather than eliminating it at inference. From scratch, the same layer learns 2, 2, 120, 2 and 3 transport states across five A_5 seeds; with the auxiliary loss below it learns 1–14. Neither result establishes recovery of the 60 state task.

5 Encouraging the transport to carry the task

Low output cross entropy does not require all prefixes with the same target to have identical transport states. Later layers can also solve a task using information absent from the transport. Standard training illustrates this ambiguity: five of six fitted $A_5 \times A_5$ runs certify only 60 of the 3,600 task states, and the sixth does not certify. Among 200 distractors, none of ten standard or identity initialised A_5 runs certifies the full task.

Prototype loss. For states s_t and labels y_t, let c_k be the mean state of class k in the batch. We add a unit weight loss with no additional learned parameters:

$$\mathcal{L}_{\text{collapse}} = -\frac{1}{N}\sum_t \log \frac{\exp(-\|s_t - c_{y_t}\|^2/\tau)}{\sum_{k\in\mathcal{K}_B}\exp(-\|s_t - c_k\|^2/\tau)}, \qquad \tau = 0.1. \tag{2}$$

$\mathcal{K}_B$ contains classes present in the batch, and gradients propagate through their means. The objective is related to prototypical networks and centre losses (Snell et al., 2017; Wen et al., 2016). It encourages class separation and within class concentration, but does not enforce exact collapse or certify a group. A constant state incurs $\log|\mathcal{K}_B|$ loss; it is not a good separated solution, although it can be a stationary point. Certification remains necessary after training.

Products and distractors. On $A_5 \times A_5$, 7/10 runs certify 3,600 states; their frame based full network executions succeed in the million token tests. On all 15/15 A_5 distractor runs, the largest restricted distractor operator distance from identity is at most 7×10^{-4}, compared with at least 2.8 in the nontrivial standard controls. These operators are approximately identity; the accepted discrete maps are exactly identity. Across A_5, S_5, 9/18 runs certify the task (6/15 and 3/3 respectively). For A_5, rank 3 transports certify in 6/6 cases; rank 4 transports fail frame search in 0/9. The three $A_5 \times \mathbb{Z}_7$ runs also fail axis frame search. Their order seven rotation plane is not covered by the

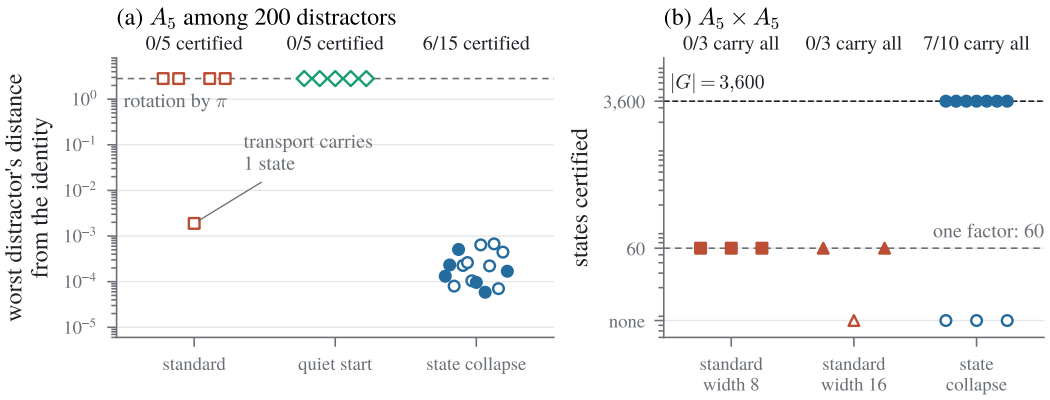

Figure 3: Effect of the prototype loss. Left: worst restricted distractor distance from identity in each A_5 run. Right: certified state count for fitted $A_5 \times A_5$ runs. Filled markers certify; hollow markers do not. Near identity continuous operators and exact identity maps after certification are distinct claims.

candidate axis construction; codebook extraction was not tested on them. Failure of this search is not a proof that no finite structure exists.

Controls and decoder dependence. The loss alone does not make the continuous model exact: pure task accuracies at length 131,072 remain 0.00–0.29. Applied to the baseline's layer 0 output without transport, it improves two of three distractor runs to 0.98 and 0.997 at 16× length, but all are at most 0.029 at 131,072. This control supports a useful state supervision effect; it does not establish that other architectures cannot support extraction. Among distractors, even an exactly executed transport does not reliably repair the full network. A separate transport readout is needed for the verified result.

Prediction based auxiliary targets. The auxiliary loss can group states by the model's own predictions. The ordinary output loss remains supervised. Using all predictions immediately certifies two of five distractor runs and no product runs. Restricting targets to confidence at least 0.9 gives 5/5 product and 1/5 distractor successes. Starting after three epochs gives 3/5 and 2/5. The 11/11 successful gated runs pass verification with prototypes named by model predictions. This removes ground truth grouping from the auxiliary loss and prototype construction, but the verification still uses the known task transition table and output labels.

Text distractors and scale. Replacing random distractors with ordered WikiText-2 bytes while retaining synthetic group tokens and targets gives 4/5 successful runs versus 0/5 without the loss. The alphabet is restricted to the 193 byte values observed in that corpus, plus task symbols. This is a synthetic tracking experiment with text distractors, not natural language understanding. At width 256 and four layers, 1/3 runs certify versus 0/3 without the loss.

6 Finite verification of unbounded correctness

Exactness boundary. Subspace choice, clustering, frame recovery and prototype construction are numerical. After extraction, token maps and accepted transition and readout tables are frozen as integer data; proof records hash the checkpoint and certificate. The separate verifier below operates on this finite object. Its guarantee does not turn the preceding numerical choices into exact real arithmetic and does not cover the unchanged neural decoder.

Let $\mathcal{S}$ be the finite set reached from the start coordinates by certified token maps f_a. A nearest prototype readout r is constructed from exact executor states on training length words. The task is specified independently by a transition table δ, start state e and output map ℓ.

checkpoints	n	reachable states	transitions checked	**proved**
standard training (confirmation)	20	60–2,520	73,500	**20/20**
state collapse	25	60–3,600	302,100	**25/25**
prediction-based collapse	3	60	24,540	**3/3**
A_8, S_8 (standard training)	5	20,160–40,320	927,360	**5/5**
controls: wrong state count	4	1–180	1,117	**0/4**

Table 3: Core verification cohort: certified transport with prototype readout. Wrong state count controls are rejected. These counts concern extracted executors, not the unchanged neural networks.

Proposition 2 (Finite state correctness)**.** *Suppose an exhaustive search from (s_0, e) produces a map $h : \mathcal{S} \to \mathcal{Q}$ such that $h(s_0) = e$ and, for every reachable s and allowed token a, $h(f_a(s)) = \delta(h(s), a)$. If $r(s) = \ell(h(s))$ for every reachable s, then the executor's readout is correct after every prefix of every finite word over the checked alphabet.*

Proof. Induct on prefix length. The start condition supplies the base case. Transition compatibility preserves the pairing after each token; the readout condition supplies the correct output. Every finite word follows checked transitions. □

The checker additionally requires a bijective correspondence with task states in the reported experiments, a stronger condition than the proposition needs and a finite bisimulation in this deterministic setting (Milner, 1989). It checks at most $|\mathcal{S}||\Sigma|$ transitions, followed by the readout on each state. Checking against the known task automaton is verification of an extracted implementation, not discovery of the task specification or a general purpose theorem prover.

Signed permutations only rearrange and negate fixed coordinates, so their execution does not accumulate arithmetic error. The revised checker uses exact stored float64 coordinate tuples, without tolerance based state merging, and rejects malformed token maps. Reconstruction, prototype construction and distances still use floating point arithmetic. The readout checks and geometric extraction are numerical checks with measured margins, not machine checked real arithmetic proofs or guarantees across every hardware/kernel implementation. The checker exports the accepted integer transition and readout tables, with hashes of the checkpoint and certificate. Executing this finite machine uses integer indexing alone and satisfies Proposition 2.

We finitely verify 53 extracted executors and readouts in the core cohort: twenty confirmation runs, 25 label based state loss runs, three ungated prediction based runs and five scale checks. Its records report 188,700 reachable states, 1,327,500 checked transitions and minimum readout margin 0.069. All 4/4 wrong state count controls fail. Two additional corruption checks replace a generator by identity or swap prototype labels. On the 25 state loss checkpoints, direct readout testing gives 78,643,200 correct positions at lengths 131,072 and 1,048,576. The separate gated prediction and text/width cohorts pass in 11/11 and 5/5 cases, with 54,657,024 evaluated positions and no errors. We keep these cohorts separate to make the denominators clear. The guarantee stops at the certified executor and checked readout.

7 Beyond groups: learning transitions that forget

An invertible orthogonal action cannot map distinct states to one reset state. We therefore extend the transport to

$$s_t = g(x_t)Q(x_t)s_{t-1} + (1 - g(x_t))c(x_t), \qquad g(x_t) \in (0, 1), \tag{3}$$

where c is the start vector or a mixture of four learned anchors. Affine maps compose associatively, so a parallel prefix scan still applies. Continuous gates need not reach zero; extraction supplies the exact reset operation.

We cluster transport states and evaluate every token's affine action on each centre. If the largest nearest centre residual is ϵ and the minimum centre separation exceeds 2ϵ, each image has a unique assigned centre. The resulting integer table gives a function on the finite state set for each token.

Functions compose associatively, generating a transformation monoid that may contain noninvertible maps. The test establishes an unambiguous discretisation; task correctness still requires Proposition 2. Labels assign a majority readout to centres, and the checker verifies it exhaustively.

With the prototype loss, 25/25 models pass the finite check across tasks with 60, 120 and 360 states (Appendix F). Their certified executions have no errors in 116,146,176 tested positions; continuous models deteriorate.

Groups and reset like components play a central role in finite automaton decomposition (Krohn & Rhodes, 1965); these experiments do not show that arbitrary automata can be learned. In particular, a counter paired with a bit and a partial reset token is representable by the tested function executor, but 33 training runs across 6 variants and 7 architectures yield 0 certificates. This twelve state task remains a learning and extraction failure; its cause is unresolved.

Another backbone and changed presentations. Replacing gated DeltaProduct with a GRU while retaining TALLY's transport gives a fitted A_5 model whose neural final 64 accuracy falls from 1.000 at length 128 to 0.013 at 32,768, while its recovered 60 state executor passes every transition and readout check. Permuting output labels also succeeds. Changing generators gives mixed results: with labels also permuted, extraction succeeds despite poor neural fit; without that permutation, it fails. These four fixed budget runs support portability of the transport, not arbitrary presentation invariance or extraction from an unmodified GRU (Appendix D). The small language modelling check shows similar predictive quality without comparable discrete structure (Appendix E).

8 Related work

Expressivity and length. Group tracking is a standard probe of recurrent and attention models (Liu et al., 2023; Merrill et al., 2024; Sarrof et al., 2024; Delétang et al., 2023). Complexity theoretic limitations depend on architectural and precision assumptions; we do not claim an unconditional separation of complexity classes. Negative eigenvalues, Householder products and fixed point methods enlarge the capabilities of efficient recurrences (Grazzi et al., 2025; Siems et al., 2025; Movahedi et al., 2025), complementing selective state space and delta rule models (Gu & Dao, 2024; Dao & Gu, 2024; Yang et al., 2024; 2025). Error control analysis (Chung et al., 2026) and state coverage interventions (Buitrago Ruiz & Gu, 2025) address extrapolation failure. Recent preprints also study additive input ablations in Householder recurrences (Howe, 2026) and representation dependent rank in a supervised group composition testbed (Larson, 2026). Our contribution is the link between these stages: recover the latent computation, predict the error of its continuous execution, intervene on that error, then compile and verify an exact executor. Neither low rank structure nor recovery of a hidden automaton is claimed as a new observation by itself.

Discrete execution and extraction. Automaton extraction already converts trained RNN behaviour into finite machines (Weiss et al., 2018; 2019). TALLY uses internal operator geometry to test whether a model contains the correct finite computation even when its continuous execution and output behaviour fail. It then compiles that structure and checks an explicit task specification. Table 4 reports a matched multiclass WL* style comparison. It has zero errors on all six A_5/A_6 checkpoints. On A_7, operator geometry selects the correct transition structure in 3/3 runs, versus 1/3 for the black box rule, but neither has zero deployed errors. The widened black box grid contains the correct structure in 3/3 runs, locating the observed advantage in label free selection and query budget rather than extractability or uniformly better answers. This adapted comparator does not establish superiority over automaton extraction in general; output audits are in Appendix B. Appendix H reports an earlier clustering comparison that found exactly $|G|$ states in 21/24 models and repaired 16/21 of them. We do not claim that finite state verification, automaton extraction or recovery of a hidden automaton is new. Our comparison is a multiclass adaptation rather than the published binary acceptor implementation, and its limits travel with the result.

Geometry and state supervision. Orthogonal recurrences preserve norms and support long range learning (Arjovsky et al., 2016; Mhammedi et al., 2017; Helfrich et al., 2018). Here their geometry enables a closure criterion, while frame extraction can also succeed for nonorthogonal controls. Codebook snapping is related to vector quantisation (van den Oord et al., 2017); the auxiliary objective adapts prototype and centre losses (Snell et al., 2017; Wen et al., 2016), with concentration

reminiscent of neural collapse (Papyan et al., 2020). Its contribution is its effect on extractable recurrent task state, not a new general loss.

9 LIMITATIONS AND CONCLUSION

TALLY makes the separation between learned structure and reliable execution measurable. The trained transport contains an approximate finite computation; conditional moments predict its state error, and controlled defect scaling changes when that error becomes a decoding failure. Compiling the recovered action into discrete composition removes this source of horizon dependence, after which exhaustive finite checks establish correctness of the executor and readout. The result does not certify the unchanged neural decoder. The known task automaton makes verification possible, and successful extraction does not establish unsupervised discovery of semantics.

The experiments use small synthetic tasks and modest seed counts. Extension cohorts are exploratory rather than a single preregistered comparison. Frame search is incomplete and geometric checks are numerical. Failure horizon prediction only slightly improves on a constant baseline, and defect scaling is tested on one model. The GRU and presentation controls use one seed and a short fixed training budget. The black box comparison uses one architecture and nine earlier checkpoints; its multiclass adaptation omits the equivalence query loop because persistent teacher errors make those counterexamples reproduce the teacher's mistakes. The prototype loss improves several extraction rates but does not guarantee learnability; partial resets and training the exact layer from scratch remain failures. Natural data benefits and efficient fused execution remain open. Within this scope, TALLY gives a measured path from approximate learned computation to an exact finite implementation whose correctness can be checked at every length.

REPRODUCIBILITY STATEMENT

The supplement contains code, tests, records, 48 core checkpoints, nine comparison checkpoints, four GRU checkpoints, and scripts to rebuild tables and figures. Five scale checkpoints are omitted for size but can be regenerated. Appendices A–D specify protocols and controls. Frozen and amended comparison records are retained. The independent checker verifies exported integer transitions and outputs; geometric extraction and prototype distances remain numerical. Reproduction notes distinguish new reruns from retained historical results.

AI ASSISTANCE STATEMENT

Generative AI tools assisted with drafting and editing the title, abstract, TLDR, keywords, contribution summary and prose; literature and prior art search; methodological and experimental feedback; critical review and result interpretation; searching for public PD SSM code; debugging numerical and software issues; and code edits and tests in the author designed analysis, extraction, exact execution, auxiliary loss, finite verification and reset pipelines. They also helped check mathematical arguments, proof steps and assumptions; implement finite verification; and audit artifacts and reproducibility. They additionally assisted with designing and executing the execution defect and presentation controls, including generating their synthetic inputs.

The author conceived the problem and central architecture; developed the recovery and discretisation methods; set the certificate and verification strategy; designed core experiments; selected results; and made final scientific interpretations and claims. The author takes responsibility for reviewing and validating AI assisted suggestions, code, mathematical reasoning, analyses and text, and for the complete work. Correctness claims rest on mathematical arguments and released exact finite verifiers, not unverified model output. The original core synthetic tasks were author designed; the additional AI assisted controls are identified above.

REFERENCES

Martin Arjovsky, Amar Shah, and Yoshua Bengio. Unitary evolution recurrent neural networks. In *International Conference on Machine Learning*, 2016.

David A. Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. *Journal of Computer and System Sciences*, 38(1):150–164, 1989.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Ricardo Buitrago Ruiz and Albert Gu. Understanding and improving length generalization in recurrent models. In *International Conference on Machine Learning*, 2025.

Jiwan Chung, Heechan Choi, and Seon Joo Kim. Rethinking state tracking in recurrent models through error control dynamics. *arXiv preprint arXiv:2605.07755*, 2026.

Oswaldo Luiz do Valle Costa, Marcelo Dutra Fragoso, and Ricardo Paulino Marques. *Discrete-Time Markov Jump Linear Systems*. Springer, 2005. doi: 10.1007/b138575.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *International Conference on Machine Learning*, 2024.

Grégoire Delétang, Anian Ruoss, Jordi Grau-Moya, Tim Genewein, Li Kevin Wenliang, Elliot Catt, Chris Cundy, Marcus Hutter, Shane Legg, Joel Veness, and Pedro A. Ortega. Neural networks and the Chomsky hierarchy. In *International Conference on Learning Representations*, 2023.

Riccardo Grazzi, Julien Siems, Arber Zela, Jörg K. H. Franke, Frank Hutter, and Massimiliano Pontil. Unlocking state-tracking in linear RNNs through negative eigenvalues. In *International Conference on Learning Representations*, 2025.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Conference on Language Modeling*, 2024.

Kyle Helfrich, Devin Willmott, and Qiang Ye. Orthogonal recurrent neural networks with scaled Cayley transform. In *International Conference on Machine Learning*, 2018.

W. Daniel Hillis and Guy L. Steele, Jr. Data parallel algorithms. *Communications of the ACM*, 29 (12):1170–1183, 1986.

Gunner Levi Howe. The automaton underneath: The additive input pathway is a parasitic attractor for state tracking in householder linear RNNs. *arXiv preprint arXiv:2609.18966*, 2026.

Kenneth Krohn and John Rhodes. Algebraic theory of machines. I. Prime decomposition theorem for finite semigroups and machines. *Transactions of the American Mathematical Society*, 116: 450–464, 1965.

Samuel Larson. The rank the task demands: A causal rank law for matrix memories trained on group composition. *arXiv preprint arXiv:2609.12259*, 2026.

Bingbin Liu, Jordan T. Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata. In *International Conference on Learning Representations*, 2023.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017.

William Merrill, Jackson Petty, and Ashish Sabharwal. The illusion of state in state-space models. In *International Conference on Machine Learning*, 2024.

Zakaria Mhammedi, Andrew Hellicar, Ashfaqur Rahman, and James Bailey. Efficient orthogonal parametrisation of recurrent neural networks using Householder reflections. In *International Conference on Machine Learning*, 2017.

Robin Milner. *Communication and Concurrency*. Prentice Hall, 1989.

Sajad Movahedi, Felix Sarnthein, Nicola Muca Cirone, and Antonio Orvieto. Fixed-point RNNs: Interpolating from diagonal to dense. In *Advances in Neural Information Processing Systems*, 2025.

Vardan Papyan, X. Y. Han, and David L. Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40): 24652–24663, 2020.

Yash Sarrof, Yana Veitsman, and Michael Hahn. The expressive capacity of state space models: A formal language perspective. In *Advances in Neural Information Processing Systems*, 2024.

Julien Siems, Timur Carstensen, Arber Zela, Frank Hutter, Massimiliano Pontil, and Riccardo Grazzi. DeltaProduct: Improving state-tracking in linear RNNs via Householder products. In *Advances in Neural Information Processing Systems*, 2025.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, 2017.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, 2017.

Gail Weiss, Yoav Goldberg, and Eran Yahav. Extracting automata from recurrent neural networks using queries and counterexamples. In *International Conference on Machine Learning*, 2018.

Gail Weiss, Yoav Goldberg, and Eran Yahav. Learning deterministic weighted automata with queries and counterexamples. In *Advances in Neural Information Processing Systems*, 2019.

Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European Conference on Computer Vision*, 2016.

Songlin Yang, Bailin Wang, Yu Zhang, Yikang Shen, and Yoon Kim. Parallelizing linear transformers with the delta rule over sequence length. In *Advances in Neural Information Processing Systems*, 2024.

Songlin Yang, Jan Kautz, and Ali Hatamizadeh. Gated delta networks: Improving Mamba2 with delta rule. In *International Conference on Learning Representations*, 2025.

A PROTOCOL

Tasks. Generators are 3 cycles: $(0\,1\,2)$ and $(2\,3\,4)$ for A_5, adding $(3\,4\,5)$ for A_6, $(4\,5\,6)$ for A_7 and $(5\,6\,7)$ for A_8; S_n adds the transposition $(0\,1)$. Token 0 is the identity token and each generator has its own token; tokens are drawn uniformly, and the label at each position indexes the prefix product. The vocabulary is the smallest power of two that holds every label and input token (at least 16). *Distractors:* each position is a group token (drawn as above) with probability $1/4$ and otherwise one of 200 distractor symbols, uniformly, which leave the state unchanged. *Products:* $A_5 \times A_5$ has one token per generator of each factor plus the identity token; $A_5 \times \mathbb{Z}_7$ adds a token that advances a counter modulo 7. The label indexes the full product state. *Natural text:* the distractor task with the same random draws for the group tokens, but each other position holds the next byte of WikiText-2 training text, read from a random offset. *Reset and flip flop* (Section 7): one token, or two, drawn at 1% (5% for the flip flop) of positions, each setting the state to a fixed element; the transport's gate is a linear head on the same input, initialised at $\sigma(3)$ so every token starts by keeping the rotation, and the four anchor states of the flip flop arm are learned.

Training. The model of Section 2, trained with AdamW (learning rate 10^{-3}, weight decay 0.1) and a cosine schedule to zero over 12 epochs of 20,000 words of length 128, batch 64 (S_8: batch 32, because batch 64 ran out of GPU memory). The checkpoint kept is the epoch with the lowest loss on 1,000 validation words of length 128, the later one on ties. The larger models use width 256 and 4 layers. One integer seeds both the data and the initialisation. Training one model takes 2–4 minutes on a single 8 GB laptop GPU (NVIDIA RTX 4070), with deterministic kernels (`ZOOLOGY_STRICT_DETERMINISM=1`, `CUBLAS_WORKSPACE_CONFIG=:4096:8`); the runs we repeated were bit identical.

Evaluation. Test words are drawn with a separate seed (training seed + 555). Table 1 uses 64 words at each length. The certified models are evaluated on 64 words at 128 and 8,192, 16 at 131,072 and 4 at 1,048,576 (state collapse runs and the exact layer: 64, 32, 8 and 2). Final 64 accuracy pools the final 64 positions of every evaluated word; where we say "every position" we count all of them. A run is *fitted* when its final 64 accuracy at length 128 is at least 0.99.

Codebook. From 1,024 unlabelled training length words: the subspace V spanned by eigenvectors of the states' second moment with eigenvalue at least 0.05 of the largest; the prefix operators restricted to V, leader clustered at Frobenius radius 0.5 (at most 8,192 clusters); cluster means projected onto the orthogonal group.

Frame search. From 256 unlabelled training length words: V as above, and each token's action on V, orthogonalised. Candidate axes are the one dimensional fixed subspaces of products of up to three token actions (and their transposes), and the intersections of pairs of their fixed subspaces. Each candidate's orbit under the token actions, up to sign (tolerance 0.3), is a set of axes; the frame is one orbit, or a union of orbits, that spans V. If none does, directions fixed by every token are split off and the search is repeated on their complement. The search and the certificate range over the symbols that can occur (for the natural text task, the byte values present in the text). The residual check also requires each token to induce a bijection of axes. Accepted maps are executed as signed permutations in frame coordinates. A first version searched for the vertices of a regular simplex instead (a permutation representation, certified at half the vertex distance); the $\beta = 1.99$ counts use it.

Black box comparison. The nine earlier checkpoints comprise three seeds each for A_5, A_6, A_7. Operator recovery reads the token operators and about 0.23–0.30 million sampled token positions. The black box arms read only model outputs. Their observation table uses 8,192 sampled prefixes of length 1–24, all words of length at most two and 32 random suffixes of length 3–6. The faithful style arm compares every observed label and probability within tolerance 0.1. The tolerant arm pools majority labels, enforces deterministic transitions and scans disagreement tolerances 0.02–0.6; the widened range is the retained R2 amendment. The arms query 20.74–66.23 million token positions. The same relative witness agreement rule selects candidates without labels. Only after selection do we use the known group to score state count, start state, bijection, every transition and outputs separately. Fresh evaluation uses seed offset 999.

Exact layer. The forward pass snaps each token's action to its frame map and composes the maps as integer signed permutations; the backward pass takes the gradient of the continuous transport. A penalty, the mean squared distance between each token's action on the frame and its snapped image (weight 1), keeps the operators near the frame. Fine tuning: 400 steps, batch 32, learning rate 10^{-4}. From scratch: a fixed rank 4 simplex frame, 4,000 steps, learning rate 10^{-3}, five seeds, without and with state collapse (weight 1) on the continuous or the exact state.

Readout and proof. Prototypes are the mean exact transport state per label over 1,024 training length words (8,192 for A_8 and S_8), named by the labels or, for the gated prediction based runs, by the model's own answers. The transport readout returns the label of the nearest prototype. The proof enumerates the states reachable from s_0 under the certified maps and checks the bisimulation and the readout on each (Section 6). The task table is used in this check even when predictions name the prototypes.

Language modelling. Byte level WikiText-2 (raw), windows of 512 bytes, batch 32, learning rate 10^{-3}; bits per byte over the full validation and test sets in nonoverlapping windows.

B MATCHED EXTRACTION DETAILS

The multiclass Moore adaptation of WL* style observation tables uses the same label free relative selection rule as operator geometry. On A_7, geometry selects 3/3 correct transition structures; the faithful style arm selects 0/3. Widening the tolerant grid yields a correct structure in 3/3 runs but selects it in 1/3. None of these A_7 black box candidates has entirely correct outputs: even the

method	A_5/A_6 struct.	A_7 struct.	A_7 grid	A_7 zero errors	query tokens
operator geometry	6/6	3/3	3/3	0/3	0.23–0.30M + weights
faithful WL* style	6/6	0/3	0/3	0/3	20.74–61.47M
tolerant black box	6/6	1/3	3/3	0/3	20.74–66.23M

Table 4: Matched extraction on nine earlier checkpoints. Structural recovery requires the right state count, start state and a bijection preserving every transition; it does not check outputs. "Grid" scores any tried tolerance retrospectively. The zero error column tests deployed answers at length 131,072. Geometry retains the neural decoder; black box machines use extracted outputs. All methods select the six A_5/A_6 transition structures.

selected 2,520 state machine mislabels three states. Neither approach has zero deployed errors on A_7. Conversely, black box execution has zero errors on all six A_5/A_6 checkpoints, versus two for geometry with the neural decoder. The advantage of weight access is structural selection at a smaller query budget, not uniformly better answer correctness. This comparison is a multiclass adaptation rather than the published binary acceptor implementation and omits an equivalence query loop.

C Quantitative execution defect

Conditional moments. Fix continuous operators $Q_a \in \mathbb{R}^{d\times d}$, a finite transition function $\delta(a, g)$ with start g_0, and reconstructed prototypes c_g. Draw tokens independently with probabilities p_a. Define $m_g(t) = \mathbb{E}[s_t \mathbf{1}\{g_t = g\}]$, $\pi_g(t) = \Pr(g_t = g)$, and $S(t) = \mathbb{E}[s_t s_t^\top]$. Then

$$m_h(t+1) = \sum_{a,g:\delta(a,g)=h} p_a Q_a m_g(t), \quad \pi_h(t+1) = \sum_{a,g:\delta(a,g)=h} p_a \pi_g(t), \tag{4}$$

$$S(t+1) = \sum_a p_a Q_a S(t) Q_a^\top, \qquad D_t = \operatorname{tr} S(t) - 2\sum_g c_g^\top m_g(t) + \sum_g \pi_g(t)\|c_g\|^2. \tag{5}$$

Initially $m_{g_0} = s_0$, $\pi_{g_0} = 1$ and $S = s_0 s_0^\top$, with other entries zero. The first two identities condition on the last transition and use independence of the next token. The third conditions $s_{t+1}s_{t+1}^\top$ on that token. Expanding the squared norm proves the last identity. This is a finite conditional moment calculation of the kind used for switching linear systems (Costa et al., 2005); we do not claim moment propagation as a new theorem. It permits correlations between state and accumulated error, and neither requires exact orthogonality nor a diffusion approximation. For separation $m = \frac{1}{2}\min_{g\neq h}\|c_g - c_h\|$, nearest prototype errors obey $\Pr(\text{error}) \le \min(1, D_t/m^2)$ by Markov's inequality. This bound is usually loose; the calibrated failure threshold below is not a theorem about accuracy.

Implementation and prediction protocol. We use the five confirmation seeds 12311–12315 on each of A_5 and S_5. Operators are reconstructed in float32 with bit identical parity to the original layer 0 extraction, then propagated in float64. The signed permutation certificate supplies δ and the orbit prototypes. Dense powers of the lifted first moment, state probability and second moment maps evaluate twelve lengths $16, 32, \ldots, 32768$. These numerical calculations are not interval certificates; dense first moment propagation scales cubically in $|G|d$ per squaring and is not proposed for the largest groups. Exhaustive enumeration of short binary words independently checks the moment implementation. At each length, 256 independent words (seed 20260922, shared across checkpoints) estimate D_t and terminal nearest prototype accuracy. Across 120 cells, the mean of $|\widehat{D}_t - D_t|/D_t$ is 0.0308. A linear fit restricted to predicted errors through length 128 has median relative discrepancy 5.1% but a maximum of 52.2% at lengths at least 1,024; a universal diffusion law is not supported by this comparison.

The first two A_5 seeds calibrate a threshold on D_t/m^2, using the geometric mean of predicted normalised errors at the adjacent sampled horizons bracketing accuracy 0.9. Pooling both models gives 1.2906876, frozen before evaluating the other eight models. Seven predicted first crossings are within a factor of two; A_5 seed 12313 is predicted at 8,192 but crosses at 32,768. Predicting 8,192

for every held out model already attains 6/8, limiting the incremental evidence for useful horizon prediction. The much tighter result concerns mean state error, not precise individual failure.

Controlled scaling. On calibration seed 12311, realise its five axis permutation certificate on the four dimensional simplex subspace, align it by orthogonal Procrustes, and extend by the identity on the orthogonal complement to obtain R_a. Let O_a be the polar projection of Q_a and $E_a = \log(R_a^\top O_a)$, checking its real skew symmetry and reconstruction numerically. Use $Q_a(\eta) = R_a \exp(\eta E_a)$ for the preselected $\eta \in \{0, 0.5, 1, 2\}$, with the same initial vector, exact orbit prototypes, words and length grid. The largest polar correction is 3.76×10^{-7}, and $\eta = 1$ reconstructs O_a within 10^{-10}. Halving/doubling the defect shifts the first sampled 90% crossing from 8,192 to 32,768/2,048; $\eta = 0$ has accuracy one throughout. The full operator distance from R_a to O_a can be large on the unused complement (maximum 0.978); we do not claim the chosen extension is a nearest exact representation. This intervention identifies a transport mechanism on one checkpoint, not a universal scaling law or a causal result for the complete neural decoder. Integer execution provides the separate exact guarantee.

D BACKBONE AND PRESENTATION CONTROLS

Presentation	Validation	Neural at 32,768	States	Proved
Canonical	1.000	0.013	60	yes
Labels permuted	1.000	0.036	60	yes
Generators and labels changed	0.256	0.014	60	yes
Generators changed	0.425	–	–	no

Table 5: A GRU with the same TALLY transport, one seed and fixed budget per arm. Validation is all position accuracy at length 128; long accuracy is over the final 64 positions of 64 words. "Proved" checks the extracted integer transition table and readout, independently of the neural output head. Failed extraction has no long executor comparison and is retained.

Replace only the inner mixer by a width 64 GRU in the two layer model. Train on 20,000 words of length 128 for two epochs (626 steps), batch 64, AdamW at constant learning rate 0.001, gradient norm clipping at one, and no prototype loss. Data and initialisation use seed 23401. Select minimum loss on 1,024 validation words (seed 23402), checked every 100 steps and at completion. The saved outer configuration is inherited from a template; these recorded commands and training logs specify the actual schedule.

The initial registered pair comprises the canonical presentation and changed generators with permuted labels. The changed generators, sampled with seed 20260922 and checked to generate A_5, are a double transposition and a 3 cycle; the same seed fixes a permutation of the 60 output labels. After the changed arm fit poorly, an explicit amendment added labels only and generators only controls, with no budget or hyperparameter tuning. These are diagnostic controls, not four independent initial confirmations.

Frame extraction receives the checkpoint, three token IDs and 256 unlabelled words of length 128. It uses no task transitions or labels. The certificate is hashed before task verification. For each successful certificate, 1,024 short words (seed 23403) name discrete states by majority task labels. Verification then checks the start and all 180 transitions and outputs against the task table. All three successful machines have 60 states; the generators only arm finds no frame. Independent integer checks reproduce the three positive proofs. Evaluation uses 64 words at each of 128, 2,048, 8,192 and 32,768 (seed 23404), reporting both continuous transport and full network predictions. The canonical and labels only networks fit and then fail at long lengths, while their discrete executors are correct. The jointly changed arm recovers a task correct machine despite poor neural fit. This separates latent computation from its use by the neural decoder; it does not demonstrate robustness to arbitrary presentations. In particular, changed generator orders also change the learning problem, so the negative arm cannot be attributed solely to label naming or solely to architecture.

task	transport	$\lvert G\rvert$	certified states	**proved**
A_5, resets at 1%	reset gate + collapse	60	60	**10/10**
A_5, resets at 1%	reset gate	60	60–64	**9/10**
A_5, resets at 1%	no gate (control)	60	1–4096	**0/5**
S_5, resets at 1%	reset gate + collapse	120	120	**5/5**
A_6, resets at 1%	reset gate + collapse	360	360	**5/5**
A_5, flip-flop	set-4 gate + collapse	60	60	**5/5**
A_5, flip-flop	set-4 gate	60	31–269	**0/5**

Table 6: Reset and set operations. Resets occur at 1% of positions, giving a mean interval of approximately 100 tokens. The two set tokens in the flip flop task select different group states. "Proved" refers to the finite executor and its checked readout. The $|G|$ column denotes the underlying group size, not monoid size.

	certified	codebook	rank	closure / margin (max)	unaltered, $1024\times$	**snapped,** $8192\times$ **(min)**
A_5	5/5	60	3–4	0.020	0.006	**1.000**
S_5	5/5	120	4	0.025	0.033	**1.000**
A_6	5/5	360	5–6	0.022	0.009	**1.000**
A_7	5/5	2,520	6	0.056	0.000	**1.000**

Table 7: Codebook extraction from 1,024 unlabelled words of length 128. The closure residual divided by the half separation must be below 0.5.

E Language modelling check

On byte level WikiText-2 (Merity et al., 2017), four layer models of width 128 train for 3,000 steps with three seeds per arm. Mean validation bits per byte are 1.8297 for TALLY and 1.8287 for the baseline; test values are 1.8642 and 1.8620. The differences are approximately 0.0010 and 0.0022 bits per byte. This small study shows similar predictive quality, not statistical equivalence or a language modelling gain. The transport implementation trains $2.1\times$ slower because its matrix scan is unfused.

Removing the learned transport at evaluation costs 0.007–0.009 bits per byte, so it contributes to prediction. However, state second moment eigenvalues span 0.12–0.13, close to an even eight dimensional spread, and no byte operator is near identity. We find no comparable low dimensional symbolic structure here. The affine variant has not been evaluated for language modelling.

F Reset and set results

With the prototype loss, 25/25 models pass, covering 60, 120 and 360 task states; their certified executions have zero errors across 116,146,176 tested positions. Without the loss, 9/15 gated runs pass; the ungated controls pass in 0/5 cases. Across 34 measured continuous models, final 64 accuracy is at most 0.26 at length 2,048 and 0.22 at 131,072. Certified centre separation is at least 0.41, with residual at most 0.030.

G Additional results

H Earlier designs

TALLY is the end point of a sequence of designs. The supplementary `history/` folder holds their code and result files, and every number below is computed from them.

A standalone orthogonal recurrence. The transport of Equation (1) on its own tracked permutation groups past its training length, but it has no writable memory and could not store and recall. Placing it in front of gated DeltaProduct kept the baseline's memory and gave the model of Section 2.

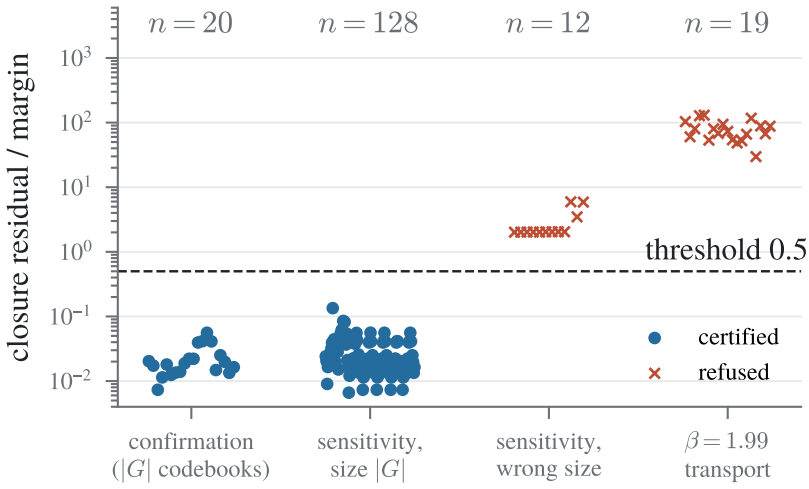

Figure 4: Closure residual divided by the acceptance margin for every codebook. Values below 0.5 certify. All 20 confirmation codebooks and 128 sensitivity codebooks of size $|G|$ pass; 12 deliberately incorrect sizes and 19 codebooks from the $\beta = 1.99$ control fail.

The horizon is finite. We first expected exact orthogonality to give an unbounded horizon. Trained at length 128, that version's final 64 accuracy on A_5 was 0.93–1.00 at 2,048, 0.08–0.28 at 32,768 and at most 0.01 at 1,048,576 (three seeds). This is the problem the paper addresses.

A near group, repaired by extraction. Extracting a state machine from each trained model's input and output behaviour found exactly $|G|$ states in 21/24 models (A_5 to S_7, up to 5,040 states). The learned operators satisfied the group's multiplication only up to a defect of 0.02–0.32 (median 0.04), and replacing them with the exact group made 16/21 achieve at least 99% accuracy in the last 64 positions at 131,072 tokens, where the unaltered models were at most 0.12. This is where "learned approximately" comes from. But the repair needed the extracted table, and a certificate based on cluster purity did not predict which models it would fix; the certificates of Section 4 test closure instead.

Exactness by training. Three attempts failed. A discrete executor with learned 0/1 transitions runs a correct machine exactly, but gradient descent does not find one. A penalty pulling each reflection onto the 64 directions whose operators are signed permutations left the snapped models at chance (final 64 0.009 and 0.016 at length 128; directions 0.15–0.25 from the nearest allowed one): a learned representation does not move to a signed permutation basis. A Fisher ratio consistency loss on the transport state lengthened the horizon on A_5 (0.28 against 0.04 at 32,768, one seed) but did not remove it: the deviation from the class centre grew from 0.034 at positions 0–127 to 0.175 at 1,024–2,047 (margin 0.308), approximately as $\sqrt{n}$ in this diagnostic; this is not a universal error law. These led to making the model exact after training (Section 4), to signed permutation frames that are found rather than trained, and to state collapse (Section 5).

Restoration. Snapping every transport state to the nearest class centre, learned with labels, every 64 tokens made the 20 transport models correct at all 115,343,360 evaluated positions (0 wrong). The regenerative scan removes labels from codebook construction and permits parallel prefix composition.

Frames. A first frame search looked for permutation frames and certified 18/20 models; the other two had learned monomial representations, which the axis frame of Section 4 covers.