WHERE TO SUPERVISE ALGORITHM LEARNING: SAME LABEL BUDGET, DIFFERENT OUTCOMES

Anonymous authors
Paper under double-blind review

ABSTRACT

A label budget specifies how much supervision a learner receives, but not where it can use that supervision effectively. We study this distinction in contextual permutation composition, where the same recurrent attention model can represent an exact algorithm through either hidden-vector or probability-based state feedback. A paired mask intervention retains or removes first-step supervision while giving every example exactly two intermediate labels and an endpoint label. Across eight fresh data blocks and two paired initializations, this substitution changes held-out eight-step accuracy by 87.41 percentage points with probability-based feedback and 55.00 points with hidden-vector feedback at 8,000 updates. Successful full-supervision controls support a 32.41-point difference between these placement effects. Yet training-length success and compositional generalization separate sharply: all 16 probability-based retain models achieve perfect complete-trajectory accuracy on the evaluated 32-step sequences, whereas none of the 48 hidden-vector models reaches 95%. The successful probability-based behavior also survives cutting gradients through the carried state. An earlier 4,000-update comparison and a full-supervision continuation show why training budget changes the interpretation of interface comparisons. Together, these results make supervision placement a measurable design choice and show that label count, state feedback, and optimization budget must be evaluated jointly when studying algorithm learning.

1 INTRODUCTION

Learning an iterative algorithm requires a model to both learn local transitions and carry a usable state between them. Intermediate supervision can help, but an annotation budget alone does not specify which transitions receive direct feedback. If two examples each receive two intermediate labels, should those labels be placed early, uniformly, or at arbitrary positions? And does the answer depend on the representation passed from one iteration to the next? These questions connect neural algorithmic reasoning (Veličković & Blundell, 2021; Veličković et al., 2022) to work on intermediate computation and process supervision (Nye et al., 2021; Uesato et al., 2022; Lightman et al., 2023).

We examine *where to supervise* through a controlled intervention on label placement. The task is contextual permutation composition: an input supplies two permutation tables, an initial state, and a sequence of operations; the model must repeatedly consult the supplied tables to produce the final state. Every sparse condition receives two intermediate state labels and the same endpoint label. We pair the inputs, initialization, training order, and update count, and replace one supervised position on a subset of examples. This design separates a placement contrast from a change in label quantity.

The state interface provides a second experimental axis. A shared recurrent attention block either propagates its hidden vector directly (*native*) or decodes a probability distribution and feeds back its weighted input embeddings (*soft*). The two interfaces have identical trainable tensor shapes and parameter counts. Both admit an explicit solution in their actual forward computation. Thus the comparison concerns how randomly initialized models learn under each interface, while a constructive solution establishes that the algorithm is representable in both. Probability-weighted feedback itself has precedents (Zhang et al., 2025); our contribution is the controlled interaction with sparse supervision and training budget.

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.

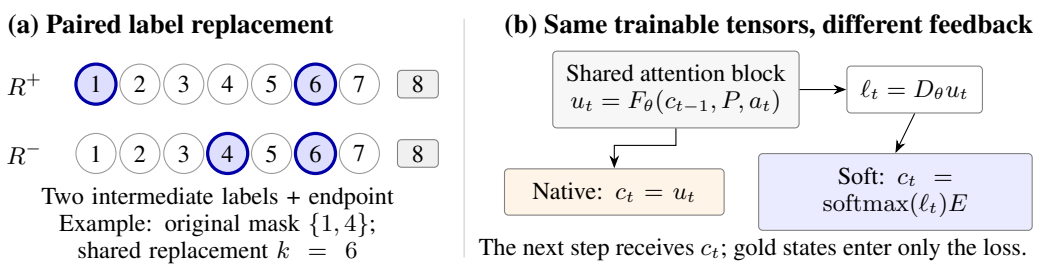


Figure 1: **A placement intervention with a fixed label count.** (a) For an original random mask $\{1, j\}$, a common replacement k gives $R^+ = \{1, k\}$ and $R^- = \{j, k\}$. Examples without step 1 are unchanged. Highlighted circles are paid intermediate labels; step 8 is always supervised. (b) The same recurrent computation uses either native hidden-state feedback or soft predicted-state feedback. Both interfaces have 136,072 trainable parameters.

Three findings organize the paper. First, the paired placement effect reproduces across both interfaces at 8,000 updates, with gains of 87.41 and 55.00 percentage points. Their difference remains substantial after both full-supervision controls meet a predefined success criterion. Second, short-chain success and longer composition separate: the native interface learns eight-step endpoints but fails the 32-step trajectory criterion, while successful soft models execute all evaluated 32-step trajectories correctly. Third, budget changes what an interface comparison can establish. Native full supervision fails the original 4,000-update control and succeeds after continuation; fresh-block confirmation at 8,000 updates then permits the intended interface comparison.

These findings support an empirical account of supervision placement with three distinct measurements: learning at the training length, composition beyond that length, and sensitivity to a specified optimization budget. A gradient-cut intervention adds a useful constraint on interpretation: soft retain models can learn without backpropagation through the carried state, but the comparison does not establish a differential gradient-dependence mechanism because the remove condition is already near chance. The central result is the measured placement–interface relationship; the gradient experiment tests one explanation of it.

2 Related Work

Learning and composing algorithms. Neural execution has been studied through curricula (Zaremba & Sutskever, 2014), Neural GPUs (Kaiser & Sutskever, 2015), and Neural Programmer-Interpreters (Reed & de Freitas, 2015). Recurrent computation with shared Transformer parameters is developed in Universal Transformers (Dehghani et al., 2019). CLRS provides a broad benchmark of algorithmic tasks (Veličković et al., 2022), with subsequent work on generalist processors (Ibarz et al., 2022), causal regularisation (Bevilacqua et al., 2023), and language-based execution (Markeeva et al., 2024). Compositional and formal-language evaluations also distinguish fitting from systematic generalization (Lake & Baroni, 2018; Delétang et al., 2022). Our single task trades breadth for control: supplied tables vary across examples, a construction is available in the tested parameterization, and a sparse-mask intervention can be paired exactly. We use functional transformations as tests of learned behavior, rather than as additional training objectives.

Expressibility, training data, and optimization. ALTA compiles algorithms into Transformer weights and studies learning from execution traces (Shaw et al., 2025). Its trace-supervision experiments train an MLP from encoded execution pairs, while other components are compiled; its discussion also anticipates decoded intermediate supervision with all parameters learned. We study random initialization of all parameters under a fixed sparse label count. Wang et al. (2025) prove learnability results for interleaved input and hidden permutation composition using structured initialization and fixed value matrices. Our supplied-table task and all-parameter AdamW training differ from those assumptions. Delayed generalization in small algorithmic datasets further motivates re-

porting optimization budgets (Power et al., 2022), without implying that the transitions observed here share the grokking mechanism.

What intermediate supervision supplies. Scratchpads expose intermediate computations (Nye et al., 2021), while process-supervised reward models assess intermediate reasoning steps (Uesato et al., 2022; Lightman et al., 2023). Jia et al. (2025) compare process and outcome supervision statistically under reinforcement-learning coverage assumptions; Li (2026) study realizable exact-trace learning. Our outcomes concern optimization with partial state labels in a continuous recurrent model. More broadly, the factorization of training targets can affect what a learner retrieves or computes (Kitouni et al., 2024). We hold the task and sparse label count fixed and estimate the effect of one explicit position substitution.

Continuous reasoning and state supervision. Coconut propagates continuous hidden states (Hao et al., 2025), and Soft Thinking propagates probability-weighted embeddings (Zhang et al., 2025). Recent latent reasoning work analyzes trajectory and representation supervision (Chen et al., 2026); structural process supervision uses prototypes and progressive alignment (Li et al., 2026). These studies motivate examining state feedback and label placement together. We add a controlled paired contrast, successful-learning controls at the chosen budget, and an intervention that preserves the soft forward values while cutting cross-step gradients. We do not transfer our task-level findings into a claim about pretrained language models.

3 Task, state interfaces, and supervision

3.1 Contextual permutation composition

Each example contains a bundle $P = (P_1, P_2)$ of permutations over $n = 8$ symbols, a starting symbol x_0, and an operation sequence $a_{1:H} \in \{1, 2\}^H$. The target trace is

$$x_t = P_{a_t}(x_{t-1}), \qquad t = 1, \ldots, H. \tag{1}$$

The input exposes all $mn = 16$ table rows $(a, s, P_a(s))$ in shuffled order. Public control flow selects the next operation a_t; table lookup and state evolution are learned. Training uses $H = 8$. Evaluation measures held-out eight-step endpoints and 32-step rollouts. Each context supplies its own rules, so a fixed mapping memorized across contexts cannot solve the task.

We distinguish endpoint exact match from complete-trajectory exact match. The latter requires every predicted state in the rollout to equal its oracle state. Eight-state uniform guessing gives a 12.5% endpoint reference; it is not a 12.5% reference for complete trajectories. Ordinary training and evaluation never insert oracle intermediate states into the recurrence.

3.2 Matched parameters with two feedback interfaces

The model is a restricted recurrent cross-attention Transformer (Vaswani et al., 2017; Dehghani et al., 2019) with width 128, four heads, and a ReLU feed-forward layer of width 256. It has no layer normalization, dropout, or temporal positional encoding. Table rows are embedded once as the sum of separate operation, key, and value embeddings. A query combines the current carried state with the current operation embedding. Attention retrieves from the table memory, followed by residual and feed-forward updates. All parameters are shared over steps.

Let $u_t = F_\theta(c_{t-1}, P, a_t)$ denote the block output, $\ell_t = D_\theta(u_t)$ the eight-state logits, and $E \in \mathbb{R}^{8\times 128}$ the learned input embedding matrix. The initial state is $c_0 = E[x_0]$. For nonfinal steps,

$$c_t = \begin{cases} u_t & \text{native,} \\ \mathrm{softmax}(\ell_t)^\top E & \text{soft.} \end{cases} \tag{2}$$

Both interfaces contain exactly 136,072 parameters, with identical initial tensors within each paired replicate. Soft feedback changes the recurrent map and its gradients; equal parameterization does not mean identical hypothesis classes or compute. The soft distribution and the shared embedding matrix both remain trainable.

An explicit assignment of these tensors executes the algorithm correctly in both real-arithmetic forward maps. The construction uses attention to select the operation–state row and a feed-forward update to overwrite the previous state with the retrieved value. Appendix B gives an invariant-set proof and distinguishes it from finite floating-point checks. Constructed parameters are used only to test representability and implementation alignment; every research training run starts from random weights.

3.3 A PAIRED REPLACEMENT AT A FIXED LABEL COUNT

Let M be a uniformly sampled two-element subset of the seven intermediate positions $\{1, \ldots, 7\}$, fixed for each example. If $1 \notin M$, set $M^+ = M^- = M$. Otherwise write $M = \{1, j\}$, draw a common k uniformly from $\{2, \ldots, 7\} \setminus \{j\}$, and set

$$M^+ = \{1, k\}, \qquad M^- = \{j, k\}. \tag{3}$$

We call the resulting conditions retain (R^+) and remove (R^-). Step 1 appears in retain on approximately $2/7$ of examples, rather than on every example; remove has no step-1 labels. The intervention changes one supervised position on affected examples. Both conditions receive exactly two intermediate labels per example and share the replacement position. Marginally, R^+ is uniform over all 21 two-position masks, while R^- is uniform over the 15 masks excluding step 1. The estimand is this specific substitution policy: other position marginals are not all preserved. Appendix D gives the derivation and realized frequencies.

For a supervised set M, the per-example objective is

$$\mathcal{L}_M = \mathrm{CE}(\ell_H, x_H) + \frac{1}{|M|} \sum_{t \in M} \mathrm{CE}(\ell_t, x_t). \tag{4}$$

Full supervision uses $M = \{1, \ldots, 7\}$ with the same mean auxiliary-loss weight. It therefore supplies more labels and serves as a positive learning control. It is not a sparse-budget competitor. Masks are independent of target values and do not enter the model's forward signature. Hidden-label invariance checks confirm that changing unused labels changes neither the loss nor its gradients.

4 EXPERIMENTAL DESIGN

4.1 THREE STAGES WITH DISTINCT EVIDENTIAL ROLES

The study developed through exploration, a fixed-budget comparison, a budget diagnosis, and fresh-block confirmation. Preliminary runs identified early-position supervision as a candidate; the original prefix and random conditions both reached perfect endpoints. A 12-run paired-mask exploration selected the retain–remove contrast before the first locked comparison. Appendix F records this sequence, including the selection rule. We report the final three stages separately:

1. **4k comparison (128 runs).** Eight fresh data/mask blocks, two paired initializations, two interfaces, and four conditions (random, retain, remove, full), each trained for 4,000 updates. All final checkpoints were frozen before locked evaluation.
2. **Budget diagnosis (32 continuations).** Full-supervision checkpoints from that comparison were continued to 8,000 updates. This post-hoc stage used the original training and development sets and no new locked test.
3. **8k confirmation (128 fresh runs).** Eight new blocks and two paired initializations crossed native/soft with retain/remove/full (96 runs), plus soft retain/remove with carried-state gradients cut (32 runs). Every run received 8,000 updates; all 128 completed, with no full reruns or outcome-based exclusions.

Within each block and initialization, every cell shares its starting tensors, sample order, and exposure counts. Each block contains 32,768 training examples and 4,096 development examples. Of 256 binary operation templates at $H = 8$, 48 are reserved for testing; the other 208 are split into 160 training and 48 development templates per block. The original reserved templates had historical evaluation exposure. Fresh confirmation therefore means fresh examples and tables, not previously

Table 1: **All eight cells of the fresh 8k confirmation.** Scores are percentages, averaged over 16 runs. Success counts use a 95% threshold for the indicated metric. Full-control qualification additionally requires training-trajectory success. H32 reports complete-trajectory accuracy.

Interface	Gradient	Labels	H8 end	Success	H32 trace	Success
Native	Standard	R^+	99.31	15/16	0.262	0/16
Native	Standard	R^-	44.31	5/16	0.743	0/16
Native	Standard	Full	99.67	15/16	0.192	0/16
Soft	Standard	R^+	100.00	16/16	100.000	16/16
Soft	Standard	R^-	12.59	0/16	0.093	0/16
Soft	Standard	Full	100.00	16/16	100.000	16/16
Soft	Cut	R^+	100.00	16/16	100.000	16/16
Soft	Cut	R^-	12.69	0/16	0.015	0/16

unknown template identities. Complete table bundles are disjoint across the new blocks and splits and excluded against saved historical tables; symbol-isomorphism classes and all historical subsequences are not excluded.

All final checkpoints and the analysis code were frozen before generating the new locked H8 and H32 sets, each with 4,096 examples per block. Models trained with AdamW (Loshchilov & Hutter, 2019), learning rate 3×10^{-4}, weight decay zero, batch size 64, gradient-norm clipping at 1, and deterministic FP32 with TF32 disabled. The 8k runs each expose 512,000 examples and 1,024,000 intermediate labels in sparse conditions. Hyperparameters and endpoints were fixed before this confirmation. No best-checkpoint selection is used.

4.2 Estimands and successful-learning controls

The primary endpoint is locked H8 endpoint exact match at the fixed update budget. For the 8k confirmation we specify four contrasts:

$$\Delta_j = \mathrm{EM}_{j,R^+} - \mathrm{EM}_{j,R^-}, \quad j \in \{\text{soft}, \text{native}\}, \tag{5}$$

$$I = \Delta_{\text{soft}} - \Delta_{\text{native}}, \tag{6}$$

$$K = (\mathrm{EM}_{\text{soft},R^-} - \mathrm{EM}_{\text{cut},R^-}) - (\mathrm{EM}_{\text{soft},R^+} - \mathrm{EM}_{\text{cut},R^+}). \tag{7}$$

We first average paired differences over the two initializations within each block, then average the eight blocks equally. We use 10,000 common block-bootstrap resamples and Bonferroni-adjusted 98.75% percentile intervals for the four contrasts. The original 4k study had three contrasts and 98.33% intervals. The independent replication unit is the data/mask block, not the individual test example, checkpoint, or initialization. With eight blocks these are approximate intervals; block effects are supplied in Appendix E.

A full-supervision run succeeds only if both training H8 trajectory accuracy and locked H8 endpoint accuracy are at least 95%. An interface's full control is reliable at 14/16 successes. Interpreting I as an interface difference requires both controls to pass, $|I| \geq 10$ percentage points, and an interval excluding zero. Placement replication requires each $\Delta_j \geq 10$ points and a positive interval lower bound. These rules were fixed before the new runs and preserve failed outcomes.

5 Results

5.1 Placement changes learning in both interfaces

At 8,000 updates, retain improves H8 endpoint accuracy by 87.41 percentage points for soft feedback and 55.00 points for native feedback (Table 2). Both placement effects meet the predefined replication rule. The soft retain models reach 100% endpoint accuracy; the soft remove models average 12.59%. Native retain averages 99.31%, while native remove averages 44.31% (Table 1).

The native effect is heterogeneous across blocks (Figure 2a). Five of 16 native remove runs succeed, whereas 15 of 16 retain runs succeed. First-step labels therefore are not necessary for every successful run under this setup. They substantially change the average outcome of the specified training

Table 2: **Predefined paired contrasts at 8k.** Units are percentage points. Each estimate averages eight block differences; intervals adjust for four comparisons. The near-zero K is limited by the standard remove condition's low baseline.

Contrast	Estimate	98.75% interval
Δ_{soft}	+87.41	[+87.17, +87.62]
Δ_{native}	+55.00	[+26.58, +79.10]
$I = \Delta_{\text{soft}} - \Delta_{\text{native}}$	+32.41	[+8.26, +60.90]
K (gradient intervention)	-0.10	[-0.38, +0.16]

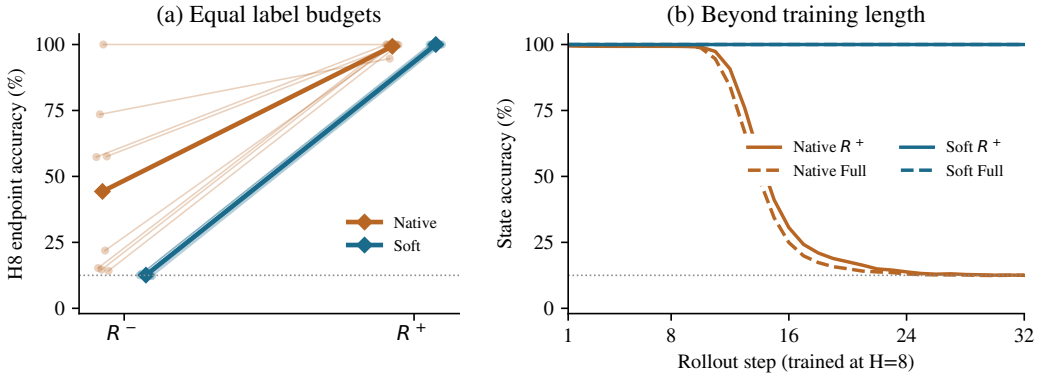


Figure 2: **Learning at H8 and composing beyond H8 give different answers.** (a) Each thin line connects the two-initialization means of one data block; diamonds show overall means. (b) Accuracy at each position of the locked 32-step rollout, averaged over 16 runs per condition. Soft retain and full curves overlap at 100%. Native performance declines along the rollout despite high eight-step endpoint accuracy. Dotted horizontal lines indicate the eight-state endpoint guessing reference.

procedure. The soft effect is more uniform: all 16 retain runs succeed, and no remove run does. Reporting both the average and all-run counts makes this distinction visible.

The full-supervision controls succeed jointly in 15/16 native runs and 16/16 soft runs, exceeding the predefined 14/16 requirement. The one native full-control failure has 93.14% training-trajectory accuracy and 94.92% locked H8 endpoint accuracy and remains a failure. With these controls, the difference between placement effects is $I = 32.41$ points, with interval $[8.26, 60.90]$. Thus the same label substitution has a different magnitude under the two tested feedback interfaces at this budget.

5.2 Training-length success does not ensure longer composition

All 48 native models fail to reach 95% complete-trajectory accuracy on the locked H32 test, including the full-supervision and retain models that mostly succeed at H8. Their mean H32 trajectory accuracies are 0.192% for full and 0.262% for retain. By contrast, all 16 standard soft retain models, all 16 soft full models, and all 16 gradient-cut soft retain models obtain 100% endpoint and complete-trajectory accuracy on the evaluated H32 sets.

The gap is not explained simply by failure to learn legal one-step transitions. A separate diagnostic constructs valid H1 inputs from oracle predecessor states on fixed development examples, without changing training. Native full models average 99.953% on these one-step problems; native retain models average 99.495%. A block that performs accurately from a supplied legal initial state can still compose poorly when its own naturally generated hidden vectors become the next inputs. This observation identifies a local-versus-rollout distinction; it does not uniquely identify an internal failure mechanism.

Functional tests ask whether the successful soft behavior remains correct when the task representation changes. On 2,048 H32 cases per block, we consistently rename symbols, reorder table rows, modify a valid permutation rule and recompute the target, or exchange naturally generated states at

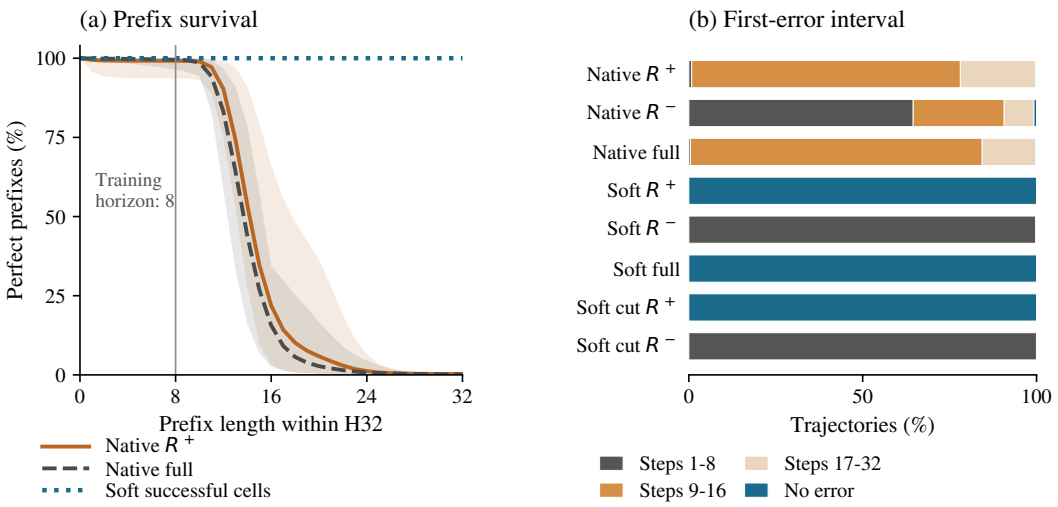


Figure 3: **Native retain/full failures accumulate after the trained horizon.** All quantities use the same locked H32 trajectories. (a) Fraction of complete prefixes with no error; lines average eight block means and shading spans their range, not a confidence interval. The vertical line marks the training horizon. Soft retain, full, and cut retain overlap at 100%. (b) First-error intervals for every cell; "No error" denotes a correct 32-step trajectory. Each cell retains all 16 runs. These descriptive summaries use recorded first-error counts, not products of marginal step accuracies.

step 16 between same-table histories. All models in the three successful soft cells score 100% on every recorded endpoint and trajectory metric. State-transfer scoring applies to the recipient suffix after the swap. These tests support finite compositional behavior on the stated interventions. Native models already fail the unperturbed H32 baseline, so their low transformed scores cannot independently diagnose an additional invariance defect.

5.3 First errors distinguish short-chain learning from longer reuse

To locate the onset of rollout errors, we analyze the first-error counts saved for every locked H32 evaluation. This is a secondary descriptive analysis. Let T be the first incorrect step, with $T = \infty$ for a fully correct trajectory. Prefix survival is

$$S(t) = \Pr(T > t) = \Pr(\hat{x}_1 = x_1, \ldots, \hat{x}_t = x_t), \qquad 1 \le t \le 32. \tag{8}$$

Thus $S(8)$ and $S(32)$ apply the same complete-prefix criterion to the same held-out sequences. We compute them directly from the saved counts, retaining all models and averaging initializations within blocks before averaging the eight blocks. This analysis complements the four predefined contrasts.

Native retain and full models preserve 99.18% and 99.53% of complete prefixes through step 8 (Figure 3a). Survival remains 99.15% and 99.40% at step 9, then falls to 21.91% and 15.62% at step 16. Across all trajectories, 77.27% and 83.91%, respectively, have their first error during steps 9–16. Failure develops over repeated extrapolation beyond the training horizon, with high accuracy extending through the first unseen step. The decline is therefore visible using trajectory correctness at both horizons within a common test set.

The other cells reveal a different failure pattern (Figure 3b). Native remove loses 64.52% of trajectories within the first eight steps, and both soft remove cells lose more than 99.88%. All three successful soft cells preserve every evaluated prefix. Placement and state feedback therefore separate two observable outcomes: acquiring a reliable computation over the trained horizon and sustaining that computation over a longer rollout. Full intermediate supervision largely addresses the former for native feedback, yet leaves the latter unresolved at this budget.

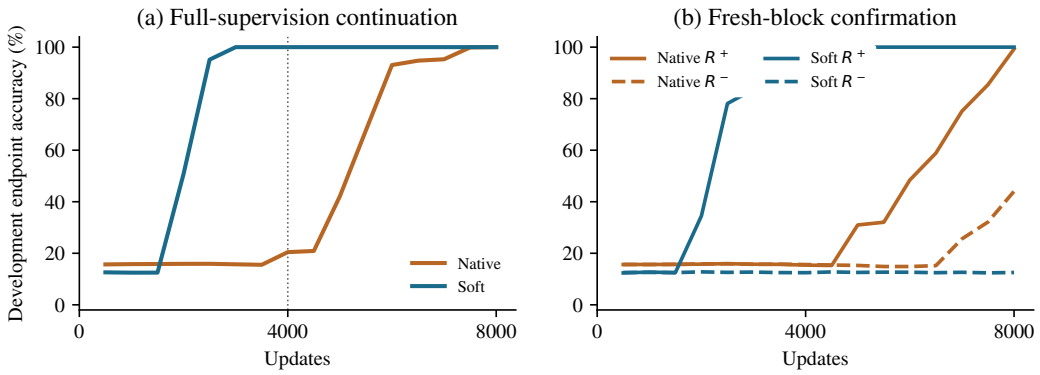


Figure 4: **Budget diagnosis and fresh confirmation are separate stages.** (a) Original full-supervision runs continued from 4k to 8k on their existing data; the dotted vertical line marks continuation. (b) Retain/remove development curves from the new 8k matrix. Curves average 16 runs and serve as secondary learning diagnostics; primary claims use the fixed final locked endpoint.

5.4 Budget determines when the interface contrast is interpretable

The earlier 4k comparison finds a soft placement effect of 87.54 points (98.33% interval $[87.25, 87.80]$), but a native effect of -0.19 points ($[-0.44, 0.04]$). Native full supervision passes in 0/16 runs, compared with 16/16 for soft. The cross-interface gate therefore fails at 4k: this experiment shows a fixed-budget performance difference but does not establish a placement-effect difference under successful learning in both interfaces.

Continuing only the 32 full-supervision runs to 8k changes native joint training/development success from 0/16 to 16/16, while soft remains 16/16 (Figure 4a). Native training-trajectory accuracy reaches 100%, and development endpoint accuracy averages 99.995%. This post-hoc diagnosis motivates the budget of the fresh 8k confirmation, whose sparse conditions and new locked tests establish the main effects above. It does not retroactively repair the failed 4k control. Together, the stages show how a sparse condition can appear ineffective when its interface has not yet learned even under full supervision.

5.5 Successful soft learning survives a cross-step gradient cut

The gradient intervention uses $c_t = \mathrm{stopgrad}(\mathrm{softmax}(\ell_t)^\top E)$ on nonfinal steps. It preserves forward logits and loss values at identical weights, while preventing a later loss from differentiating through a preceding carried-state edge. Local losses, local parameter gradients, shared parameters, and forward state transmission remain. Initialization, masks, and updates are paired with the standard soft conditions.

All 16 cut retain models learn perfectly at H8 and pass all evaluated H32 and functional tests. The cut remove models remain near chance (12.69%). The predefined interaction is $K = -0.10$ points, with interval $[-0.38, 0.16]$. The proposed position-specific gradient-dependence prediction is not supported: standard remove performance is already near chance and cannot provide a useful degradation comparison. The positive observation is nevertheless direct: in this setting, retain supervision supports successful soft learning even when gradients through the carried state are cut. It does not imply that all credit assignment or all forms of gradient propagation are unnecessary.

6 Discussion

Supervision placement is part of the learning problem. The experiment shows that the same number of intermediate labels can lead to sharply different training outcomes. Step 1 is special in this recurrence because it starts from a supplied state embedding; later labels supervise computations that depend on model-generated states. This asymmetry motivates the intervention, but it is not an identification theorem. Indeed, a restricted permutation example in Appendix C can identify a local

rule from complete labels at coprime later positions. The empirical result concerns the specified finite data and optimization process.

Interfaces should be compared after checking what has learned. Successful full supervision gives context to a sparse-supervision comparison. The failed 4k native control and its successful continuation expose a budget-dependent confound, while new 8k controls permit the intended comparison. Even after this check, native and soft differ markedly in long-chain behavior. Endpoint accuracy, local transition accuracy, and rollout accuracy should therefore be reported separately when evaluating algorithm learning.

Diagnose acquisition and reuse separately. The first-error analysis makes this distinction measurable within a single test set. Native retain/full models usually remain correct through the trained horizon and the next step, then accumulate errors as their own states are reused. Extra intermediate labels alone do not remove this pattern. Soft feedback repeatedly maps predictions into the convex hull of shared state embeddings, whereas native feedback carries an unconstrained hidden vector. This difference suggests testing the stability of model-generated states across rollout lengths. The present measurements establish the behavioral separation; a direct account of its origin would require interventions on those states. For evaluation, prefix survival already adds useful information: it locates whether a poor long-chain score reflects failure to acquire short computations or failure to sustain them.

Scope and remaining questions. The study covers one supplied-table task, one matched parameterization, two update budgets, and eight independent blocks per comparison. It establishes large effects within this controlled setting, with approximate small-block inference, rather than a universal ranking of sparse policies or a broad benchmark advantage. The original reserved templates have historical exposure, and exact table exclusion does not exclude symbol-isomorphic instances. Broader tasks, architectures, label budgets, and optimization schedules remain tests of transfer. The gradient-cut result constrains one proposed explanation but leaves the learning mechanism unresolved. Those questions motivate extensions of the present design without changing its supported conclusions.

7 Conclusion

At a fixed sparse label count, a paired change in label placement changes algorithm learning in both tested state interfaces. Full-supervision controls and a budget continuation reveal when that comparison becomes interpretable; long-chain evaluation then separates training-length success from compositional behavior. The resulting lesson is operational: specify where labels occur, what state is carried, and how much optimization is allowed, and evaluate those choices jointly.

AI USE STATEMENT

Generative AI tools (OpenAI Codex) assisted research planning and hypothesis refinement, experiment and analysis implementation, mathematical derivations, interpretation of results, literature retrieval, figure and table preparation, and manuscript drafting and editing. Synthetic examples were generated by seeded permutation samplers and labeled by a deterministic oracle, rather than by language-model responses. Automated checks covered oracle agreement, constructive forward behavior, paired initialization and exposures, checkpoint identity, gradient interventions, source and data hashes, and recomputation of statistical summaries. These checks establish specific computational properties and do not replace author review of the scientific arguments. The authors are responsible for the final text, claims, and artifacts.

REPRODUCIBILITY STATEMENT

Sections 3–4 define the task, interfaces, masks, objective, and inference. Appendices A–I give implementation details, the construction, data provenance, block effects, and artifact instructions. The anonymous supplement contains source code, the fresh-confirmation datasets and masks, run-level results, and a CPU-only verification route for the reported contrasts. Training checkpoints are not required to recompute the reported statistics; the artifact documentation distinguishes this result audit from independent retraining.

ETHICS STATEMENT

The experiments use synthetic finite-state permutation data and involve no human participants or personal data. Reported costs concern a single local GPU campaign. Claims are restricted to the evaluated algorithm-learning setting.

REFERENCES

Beatrice Bevilacqua, Kyriacos Nikiforou, Borja Ibarz, Ioana Bica, Michela Paganini, Charles Blundell, Jovana Mitrovic, and Petar Veličković. Neural algorithmic reasoning with causal regularisation. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 2272–2288. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/bevilacqua23a.html.

Xinghao Chen, Chak Tou Leong, Wenjin Guo, Jian Wang, Wenjie Li, and Xiaoyu Shen. What Makes Effective Supervision in Latent Chain-of-Thought: An Information-Theoretic Analysis. *arXiv preprint arXiv:2606.20075*, 2026. URL https://arxiv.org/abs/2606.20075.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal Transformers. *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1807.03819.

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. *arXiv preprint arXiv:2207.02098*, 2022. URL https://arxiv.org/abs/2207.02098.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training Large Language Models to Reason in a Continuous Latent Space. *Conference on Language Modeling*, 2025. URL https://arxiv.org/abs/2412.06769.

Borja Ibarz, Vitaly Kurin, George Papamakarios, Kyriacos Nikiforou, Mehdi Bennani, Róbert Csordás, Andrew Joseph Dudzik, Matko Bošnjak, Alex Vitvitskyi, Yulia Rubanova, Andreea Deac, Beatrice Bevilacqua, Yaroslav Ganin, Charles Blundell, and Petar Veličković. A generalist neural algorithmic learner. In Bastian Rieck and Razvan Pascanu (eds.), *Proceedings of the First Learning on Graphs Conference*, volume 198 of *Proceedings of Machine Learning Research*, pp.

2:1–2:23. PMLR, 09–12 Dec 2022. URL https://proceedings.mlr.press/v198/ibarz22a.html.

Zeyu Jia, Alexander Rakhlin, and Tengyang Xie. Do we need to verify step by step? Rethinking process supervision from a theoretical perspective. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 27373–27398. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/jia25f.html.

Łukasz Kaiser and Ilya Sutskever. Neural GPUs Learn Algorithms. *arXiv preprint arXiv:1511.08228*, 2015. URL https://arxiv.org/abs/1511.08228.

Ouail Kitouni, Niklas Nolte, Diane Bouchacourt, Adina Williams, Mike Rabbat, and Mark Ibrahim. The Factorization Curse: Which Tokens You Predict Underlie the Reversal Curse and More. *arXiv preprint arXiv:2406.05183*, 2024. URL https://arxiv.org/abs/2406.05183.

Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2873–2882. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/lake18a.html.

Yiqi Li, Xu Chen, Chen Ju, Jiangchao Yao, Zhaoyang Li, Jinsong Lan, Xiaoyong Zhu, Bo Zheng, and Yu Wang. Structural Process Supervision for Latent Chain-of-Thought Reasoning. *arXiv preprint arXiv:2609.09928*, 2026. URL https://arxiv.org/abs/2609.09928.

Zhiyuan Li. The Optimal Sample Complexity of Learning Autoregressive Chain-of-Thought. *arXiv preprint arXiv:2607.07423*, 2026. URL https://arxiv.org/abs/2607.07423.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's Verify Step by Step. *arXiv preprint arXiv:2305.20050*, 2023. URL https://arxiv.org/abs/2305.20050.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. *International Conference on Learning Representations*, 2019. URL https://arxiv.org/abs/1711.05101.

Larisa Markeeva, Sean McLeish, Borja Ibarz, Wilfried Bounsi, Olga Kozlova, Alex Vitvitskyi, Charles Blundell, Tom Goldstein, Avi Schwarzschild, and Petar Veličković. The CLRS-Text Algorithmic Reasoning Language Benchmark. *arXiv preprint arXiv:2406.04229*, 2024. URL https://arxiv.org/abs/2406.04229.

Maxwell Nye, Anders Johan 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 preprint arXiv:2112.00114*, 2021. URL https://arxiv.org/abs/2112.00114.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets. *arXiv preprint arXiv:2201.02177*, 2022. URL https://arxiv.org/abs/2201.02177.

Scott Reed and Nando de Freitas. Neural Programmer-Interpreters. *arXiv preprint arXiv:1511.06279*, 2015. URL https://arxiv.org/abs/1511.06279.

Peter Shaw, James Cohan, Jacob Eisenstein, Kenton Lee, Jonathan Berant, and Kristina Toutanova. ALTA: Compiler-Based Analysis of Transformers. *Transactions on Machine Learning Research*, 2025. URL https://arxiv.org/abs/2410.18077.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process- and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022. URL https://arxiv.org/abs/2211.14275.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. *arXiv preprint arXiv:1706.03762*, 2017. URL https://arxiv.org/abs/1706.03762.

Petar Veličković and Charles Blundell. Neural Algorithmic Reasoning. *arXiv preprint arXiv:2105.02761*, 2021. URL https://arxiv.org/abs/2105.02761.

Petar Veličković, Adrià Puigdomènech Badia, David Budden, Razvan Pascanu, Andrea Banino, Misha Dashevskiy, Raia Hadsell, and Charles Blundell. The CLRS algorithmic reasoning benchmark. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 22084–22102. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/velickovic22a.html.

Zixuan Wang, Eshaan Nichani, Alberto Bietti, Alex Damian, Daniel Hsu, Jason D Lee, and Denny Wu. Learning compositional functions with transformers from easy-to-hard data. In Nika Haghtalab and Ankur Moitra (eds.), *Proceedings of Thirty Eighth Conference on Learning Theory*, volume 291 of *Proceedings of Machine Learning Research*, pp. 5632–5711. PMLR, 30 Jun–04 Jul 2025. URL https://proceedings.mlr.press/v291/wang25a.html.

Wojciech Zaremba and Ilya Sutskever. Learning to Execute. *arXiv preprint arXiv:1410.4615*, 2014. URL https://arxiv.org/abs/1410.4615.

Zhen Zhang, Xuehai He, Weixiang Yan, Ao Shen, Chenyang Zhao, Shuohang Wang, Yelong Shen, and Xin Eric Wang. Soft Thinking: Unlocking the Reasoning Potential of LLMs in Continuous Concept Space. *arXiv preprint arXiv:2505.15778*, 2025. URL https://arxiv.org/abs/2505.15778.

A Implementation and optimization details

Table 3 lists the shared settings. The input memory for row (a, s, v) is $T_a + K_s + V_v$, using separate learned embeddings for each role. Its keys and values are projected once. At each step, a projected query from $c_{t-1} + Q_{a_t}$ attends to all 16 memory rows, using four heads with head dimension 32 and the usual $1/\sqrt{32}$ scaling. If r_t is the projected attention result, the block computes

$$h_t = c_{t-1} + r_t, \qquad u_t = h_t + W_2 \,\mathrm{ReLU}(W_1 h_t + b_1) + b_2. \tag{9}$$

The decoder is a learned affine map to eight logits. Native feedback carries u_t; soft feedback uses Equation 2. There is no argmax in the soft training recurrence and no teacher forcing. The operation sequence is externally supplied and processed in order, so the model does not learn a program counter or infer an operation schedule.

Embedding entries are initialized independently from a normal distribution with standard deviation 0.1. Linear weights use Xavier uniform initialization, with zero biases. The attention-output projection and final feed-forward matrix are scaled by 0.1 after initialization to limit initial residual growth. These initial values are shared across all cells for a block and initialization. Constructed weights are never used for research training.

Table 3: Shared model and optimization settings.

Setting	Value
States / supplied permutation tables	8 / 2
Training horizon / longer test horizon	8 / 32
Width / heads / feed-forward width	128 / 4 / 256
Trainable parameters	136,072
Train / development examples per block	32,768 / 4,096
Locked H8 / H32 examples per block	4,096 / 4,096
Batch size / final updates in confirmation	64 / 8,000
Optimizer	AdamW, learning rate 3×10^{-4}
Adam coefficients / numerical epsilon	$(0.9, 0.999)$ / 10^{-8}
Weight decay / gradient norm limit	0 / 1
Arithmetic	FP32; TF32 disabled
Auxiliary coefficient / endpoint coefficient	1 / 1
Sparse / full intermediate labels per example	2 / 7
Development evaluation interval	500 updates
Model snapshots for diagnostics	0, 1k, 2k, 4k, 6k, 8k

The sampler uses a separate seeded permutation generator, processes all examples before reshuffling, and preserves the cursor and generator state in checkpoints. Each 8k model sees 512,000 examples: 1,024,000 intermediate-label exposures in sparse conditions and 3,584,000 in full conditions. Endpoint exposure is identical. Full supervision averages its seven auxiliary losses, preventing label count from multiplying the auxiliary coefficient. The training code computes all logits, but only selected intermediate labels contribute to the objective. Tests mutate unused labels and verify equality of both loss and parameter gradients.

B Constructive solution in both actual forward maps

This construction establishes representability in the implemented parameterization. It is an existence argument, rather than a theorem about convergence from random initialization. We include it to rule out absence of a representable solution as an explanation of the empirical failures.

Proposition 1. *For $n = 8$ states and $m = 2$ operations, the implemented native and soft interfaces each admit the same fixed parameter assignment whose decoded argmax follows every valid supplied permutation program at every finite step in real arithmetic.*

Proof. Reserve disjoint coordinates S for the n state components, O for the m operation components, and T for an n-component temporary vector. Input state embeddings are one-hot on S. Query

and memory operation embeddings are one-hot on O; table keys are one-hot on S, and table values are one-hot on T. Activate only attention head 0. Set its query/key weights so that the score of row (a, s) for requested operation a_t is

$$A\big(\mathbf{1}[a = a_t] + z_s\big), \qquad A = 20,$$

where z is the current state vector on S. All irrelevant coordinates and other heads are zero. The value and output projections place the attention-weighted next-state distribution q in T. On the nonnegative S, T coordinates, a ReLU feed-forward update subtracts the old S, copies T into S, and clears T. Thus the post-block state is q on S, with all other coordinates zero. The decoder has logits Bq, with $B = 10$.

For true current state s, suppose z is a probability vector satisfying $\|z - e_s\|_\infty \le \varepsilon = 10^{-3}$. The correct row (a_t, s) exceeds every same-operation incorrect-key score by at least $A(1-2\varepsilon)$ and every other-operation score by at least A. Its total off-row attention mass is at most

$$\delta(\varepsilon) = (n-1)e^{-A(1-2\varepsilon)} + (m-1)ne^{-A} < 3.16 \times 10^{-8}. \tag{10}$$

Writing $s' = P_{a_t}(s)$, this gives $q_{s'} \ge 1 - \delta$ and $q_j \le \delta$ for $j \ne s'$. The decoder's argmax is correct. Native feedback carries q, which remains in the invariant set because $\delta < \varepsilon$. Soft feedback carries $\mathrm{softmax}(Bq)$ on the same one-hot embedding codebook; its total mass outside s' is bounded by

$$\rho(\varepsilon) = (n-1)e^{-B(1-2\delta(\varepsilon))} < 3.18 \times 10^{-4} < \varepsilon. \tag{11}$$

Starting from e_{x_0}, induction preserves the invariant and correct argmax at every finite step. The implemented width, head dimension, and feed-forward width suffice for this assignment. □

The residual cancellation above is exact in real arithmetic. It does not by itself establish an unbounded FP32 error guarantee. Separate implementation checks exhaustively enumerate all $24^2 \times 4 \times 2^2 = 9,216$ cases for $n = 4, m = 2, H = 2$ in both interfaces and test finite $n = 8$ probes through $H = 64$. The construction follows the actual recurrence, including soft re-encoding, and uses no alternative inference program.

C Why first-step labels are not an identification requirement

Consider one fixed context and one operation, with candidate transitions restricted to permutations f. Suppose supervision reveals the complete maps f^r and f^s, covering every initial state, where $\gcd(r, s) = 1$. Bézout's identity gives integers u, v with $ur + vs = 1$. Since powers of a permutation commute and are invertible,

$$f = (f^r)^u (f^s)^v. \tag{12}$$

For example, $f = f^3(f^2)^{-1}$ and $f = (f^3)^2(f^5)^{-1}$. Later labels can therefore identify the local rule under these restricted assumptions. Conversely, identity and a nontrivial involution agree at all even powers, so labels only at positions 2, 4, and 8 need not identify step 1. Enumeration over the 24 four-state permutations yields 24 distinct signatures for positions $(2, 3)$ and $(3, 5)$, and 12 for $(2, 4)$.

This observation is not an identifiability result for our mixed-operation, finite, varying-context training set. Without a restricted hypothesis class across contexts, even complete traces on finitely many tables allow functions that agree on observed tables and disagree elsewhere. The empirical placement result thus should be understood as a learning outcome under a particular training procedure, not as an information-theoretic necessity of supervising the first transition.

D Masks, seeds, and data provenance

Table 4 gives the realized training-mask marginals across all eight fresh-confirmation blocks (262,144 examples). Each example has exactly two intermediate labels. The mask replacement is defined before model fitting and independently of labels. The shared replacement position prevents the two arms from receiving entirely unrelated masks, while the later-position frequencies necessarily change when first-step supervision is removed.

The marginal mask distributions also have a simple exact form. Each $\{1, k\}$ in R^+ receives probability $5(1/21)(1/5) = 1/21$ from the five possible original partners of step 1. Every mask without

step 1 is unchanged, also with probability $1/21$. For R^-, each $\{i, j\} \subseteq \{2, \ldots, 7\}$ receives its original mass plus two replacement routes, giving $1/21 + 2(1/21)(1/5) = 1/15$. Thus retain preserves the uniform random-mask marginal, and remove is uniform conditional on excluding step 1.

Table 4: Percentage of examples supervised at each intermediate position. Random is the base mask used to construct the two trained sparse arms; it is not an additional trained arm in the 8k matrix.

Mask	1	2	3	4	5	6	7
Random	28.54	28.65	28.64	28.61	28.42	28.53	28.61
R^+	28.54	28.59	28.59	28.72	28.36	28.60	28.60
R^-	0.00	33.34	33.36	33.48	33.13	33.33	33.36

Fresh-confirmation random streams derive from `SeedSequence([2026091209, phase, block, purpose])`. Initializations are $9000 + 2b + r$, with $b \in \{0, \ldots, 7\}$ and $r \in \{0, 1\}$. Separate purposes control tables, templates, masks, replacements, diagnostic subsets, and functional transformations. The supplement includes the actual data/mask arrays and per-block manifests, so reconstruction need not depend only on a prose seed description.

Each context contains independently sampled permutation tables, with shuffled row order. Complete table bundles are checked against saved historical bundles and against previously generated splits and blocks; collisions are rejected using fixed streams and recorded. This controls literal table reuse. It does not remove all symbol-isomorphic tables, nor all appearances of a held-out eight-operation template as a subsequence in earlier longer programs. The 48 reserved H8 templates are inherited from the earlier study and have already been evaluated historically. All 128 final 8k checkpoints were frozen before generating the fresh locked examples and computing their scores.

E BLOCK-LEVEL INFERENCE AND COMPLETE RESULTS

Let Y_{brjc} be the primary endpoint for block b, initialization r, interface j, and condition c. Each placement block effect is $d_{bj} = \frac{1}{2}\sum_r (Y_{brj,+} - Y_{brj,-})$, and $\widehat{\Delta}_j = \frac{1}{8}\sum_b d_{bj}$. The same block-level averaging applies to I and K. Bootstrap samples draw eight block indices with replacement and retain all within-block pairing across contrasts. For four comparisons, interval quantiles are 0.00625 and 0.99375. These are nominal multiplicity-adjusted percentile intervals with only eight independent blocks; no exact finite-sample coverage claim is made.

Table 5: All block effects at 8k, in percentage points. The two initializations have already been averaged within each row.

Block	Δ_{soft}	Δ_{native}	I	K
0	+87.451	+42.639	+44.812	-0.586
1	+86.890	+84.680	+2.209	+0.195
2	+87.610	+21.155	+66.455	-0.293
3	+87.219	+85.352	+1.868	-0.024
4	+87.488	-0.012	+87.500	-0.012
5	+87.305	+78.113	+9.192	+0.134
6	+87.549	+42.407	+45.142	-0.525
7	+87.781	+85.657	+2.124	+0.317

The gradient-dependence support rule requires $K \geq 10$ points with a positive adjusted lower bound, reliable standard-soft full supervision, and at least 14/16 endpoint successes in both standard-soft remove and cut-soft retain. Standard-soft remove has 0/16 successes, and K is near zero. The specific prediction is therefore unsupported; the floor prevents interpreting this as equivalence of gradient dependence.

F EARLIER STAGES AND SELECTION HISTORY

Early pilot and architecture studies preceded the locked comparisons. The preliminary supervision experiments found that prefix and random labels both reached perfect endpoints while a uniform placement remained near chance. Consequently, an advantage over random placement is not a finding of this paper. A subsequent paired-mask exploration trained two new arms across two existing data draws and three initializations per draw (12 runs). The planned route retained the retain–remove hypothesis only if all retain runs reached 95% and each draw's mean advantage exceeded 10 points. That route passed before the 128-run 4k comparison. Its exploratory outcomes are included in the supplement's records.

The 4k matrix includes random as well as retain, remove, and full conditions. Table 6 retains all its cells. Its full-control gate failed for native, so its numerical interaction was not interpreted as the gated interface result. The full-supervision continuation was then selected post hoc to diagnose budget sensitivity. Only the later 8k matrix provides fresh-block confirmation at the longer budget. Artifact directories preserve the original stage names: `research_evidence` (4k), `budget_diagnostic_v8` (continuation), and `followup_v9` (fresh 8k).

Table 6: All conditions of the original locked 4k comparison. Mean percentages over 16 runs per row.

Interface	Labels	H8 endpoint	H32 complete trajectory
Native	Random	14.827	0.694
Native	R^+	14.835	0.789
Native	R^-	15.021	0.697
Native	Full	19.774	0.801
Soft	Random	100.000	100.000
Soft	R^+	100.000	100.000
Soft	R^-	12.457	0.024
Soft	Full	100.000	100.000

The original comparison required a technical logging correction and a checkpoint resume; these were recorded separately from scientific outcomes. No unfavorable result triggered a replacement seed. The fresh 8k campaign completed exactly 128 from-scratch attempts and did not use its allowance for technical full reruns. All scientific sources, configurations, data, and analysis rules remain frozen in their original manifests.

G PREFIX SURVIVAL AND FIRST-ERROR DIAGNOSTICS

The saved H32 evaluation for each run records counts $c_1, \ldots, c_{32}, c_\infty$, where c_t counts trajectories whose first error is at step t. The counts are nonnegative integers and sum to 4,096. The complete-prefix fraction is $S(t) = (c_\infty + \sum_{j=t+1}^{32} c_j)/4096$. In particular, $S(32)$ equals the originally reported complete-trajectory accuracy. We verify this equality for every run and also check that prefix survival never exceeds the marginal accuracy at the corresponding step. Subsequent recovery of a correct symbol does not restore an already failed prefix.

Table 7 gives the numerical summaries behind Figure 3. All eight cells retain both initializations in all eight blocks. The first-error intervals follow the trained horizon and its successive multiples; no interval was used to select models or redefine the primary endpoint. These summaries were derived during manuscript preparation from the frozen evaluations. We add no hypothesis test or uncertainty interval beyond the original four planned contrasts; the figure's shaded range displays variation across the eight block means.

H FUNCTIONAL AND GRADIENT DIAGNOSTICS

The functional transformations apply to the first 2,048 locked H32 examples in each block, identically across paired models. Consistent symbol renaming changes table keys, table values, and initial

Table 7: Complete-prefix accuracy within the same locked H32 trajectories, and the fraction of all trajectories first failing during steps 9–16. Percentages average two initializations within each block, then eight blocks equally.

Interface	Labels	$S(8)$	$S(16)$	$S(32)$	First error 9–16
Native	R^+	99.18	21.91	0.26	77.27
Native	R^-	35.48	9.30	0.74	26.18
Native	Full	99.53	15.62	0.19	83.91
Soft	R^+	100.00	100.00	100.00	0.00
Soft	R^-	0.11	0.10	0.09	0.02
Soft	Full	100.00	100.00	100.00	0.00
Soft cut	R^+	100.00	100.00	100.00	0.00
Soft cut	R^-	0.02	0.02	0.02	0.00

states with the same bijection. Row reordering preserves the table function. Rule modification changes a valid permutation and recomputes every oracle target; it does not silently retain the old labels. The state-transfer test uses the same table with a donor history that produces a different oracle state at step 16, exchanges the actual naturally generated carried state, and evaluates the recipient's remaining suffix against a recomputed oracle. For native the transferred object is its hidden vector; for soft it is its weighted embedding. No input embedding is substituted for a native internal state. The diagnostic rollout reproduces the ordinary forward output when no intervention is applied.

All final models are tested, including failures. The supplement provides endpoint and trajectory metrics for every transformation and every run. The local H1 probe creates eight legal one-step problems per selected development example using oracle predecessor states; this is an independent inference diagnostic, not an oracle-assisted evaluation of the original rollout. It uses 512 fixed development examples per block.

Gradient diagnostics use 64 fixed development examples and the actual paid-label coefficients. They record each intermediate-position contribution, the auxiliary sum, the endpoint contribution, and the total at six saved snapshots. Unpaid positions contribute zero. Standard and cut gradients are compared at identical weights and inputs, and forward equality is checked independently. The 768 snapshot diagnoses are explanatory measurements and were not used for hyperparameter or checkpoint selection.

I Compute and reproducibility artifacts

Experiments ran on one NVIDIA RTX 3090 Ti using Python 3.11.15 and PyTorch 2.10.0+cu128, with four CPU threads and deterministic FP32 settings. Table 8 reports mean training time and maximum allocated CUDA memory per cell. Equal updates and label exposures do not imply equal FLOPs, especially when cross-step gradients are cut. The complete fresh-confirmation campaign charged 11.678 GPU-worker hours including preflight, evaluation, diagnostics, and packaging. The three final stages together charged 19.631 hours, including the 12-run paired-mask exploration preceding the 4k comparison. These are worker elapsed times rather than GPU kernel-active hours; they exclude the project's 89 earlier pilot and architecture runs. Saved assets for the final stages occupy approximately 1.87 GB before submission packaging.

Table 8: Fresh-confirmation training cost by cell. Peak memory is the maximum allocated CUDA memory among 16 runs; it is not total device reservation.

Interface	Gradient	Labels	Mean seconds	Peak MiB
Native	Standard	R^+	310.4	94.79
Native	Standard	R^-	311.7	94.79
Native	Standard	Full	319.0	94.79
Soft	Standard	R^+	328.0	94.83
Soft	Standard	R^-	320.7	94.83
Soft	Standard	Full	323.1	94.83
Soft	Cut	R^+	320.8	94.82
Soft	Cut	R^-	325.2	94.82

The anonymous ZIP provides the fresh-confirmation datasets and masks, run-level scores from the three reported stages, learning curves and diagnostics, source snapshots, portable verification instructions, and SHA-256 checksums. The default verification route runs on CPU and recomputes the four predefined contrasts from saved per-run scores; it also checks the masks and oracle traces. This validates reported summaries and data integrity, not neural inference from saved model weights. Optimizer states and the full checkpoint collection are excluded from the compact ZIP. A separate opt-in training route supports a matched single-run reproduction using the supplied data; exact GPU replay across hardware and software versions is not assumed.

The paper PDF contains the complete textual appendix, so reading the ZIP is unnecessary to understand the main experimental design or claims.